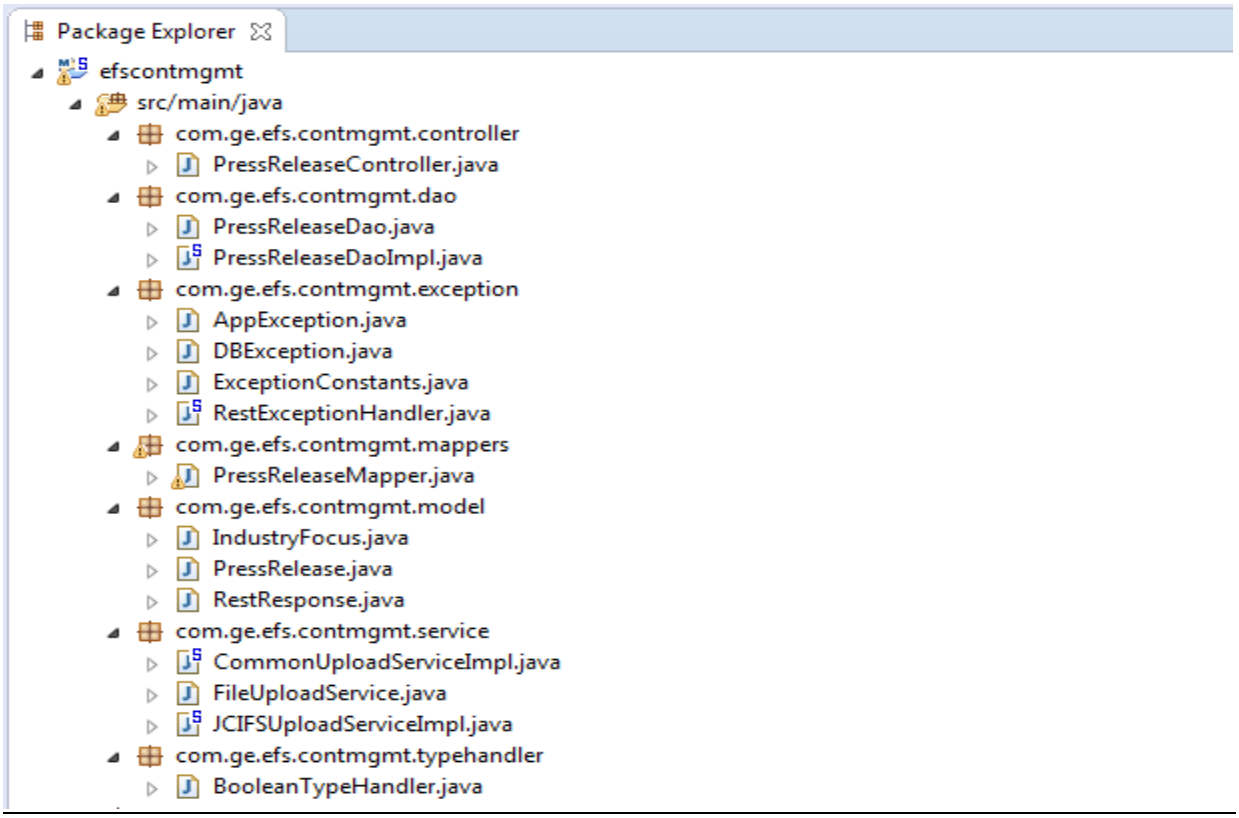
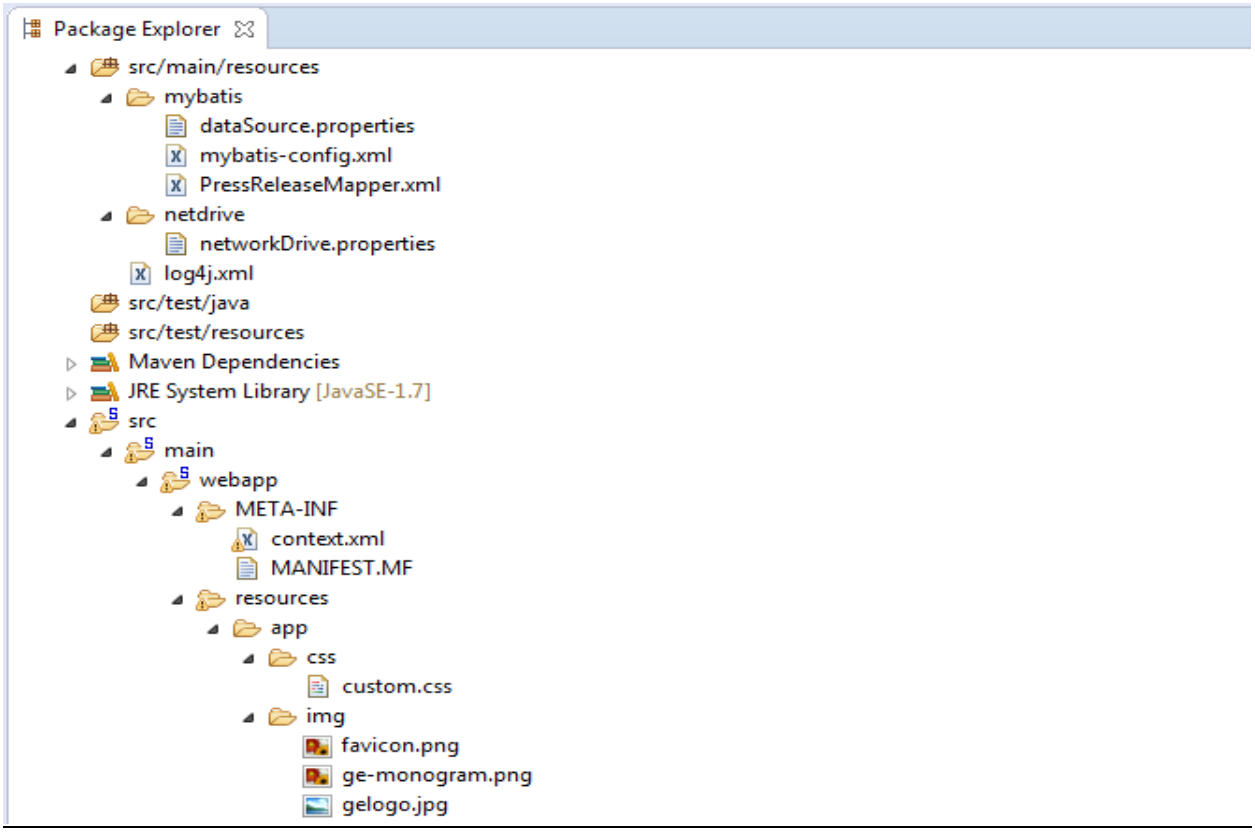
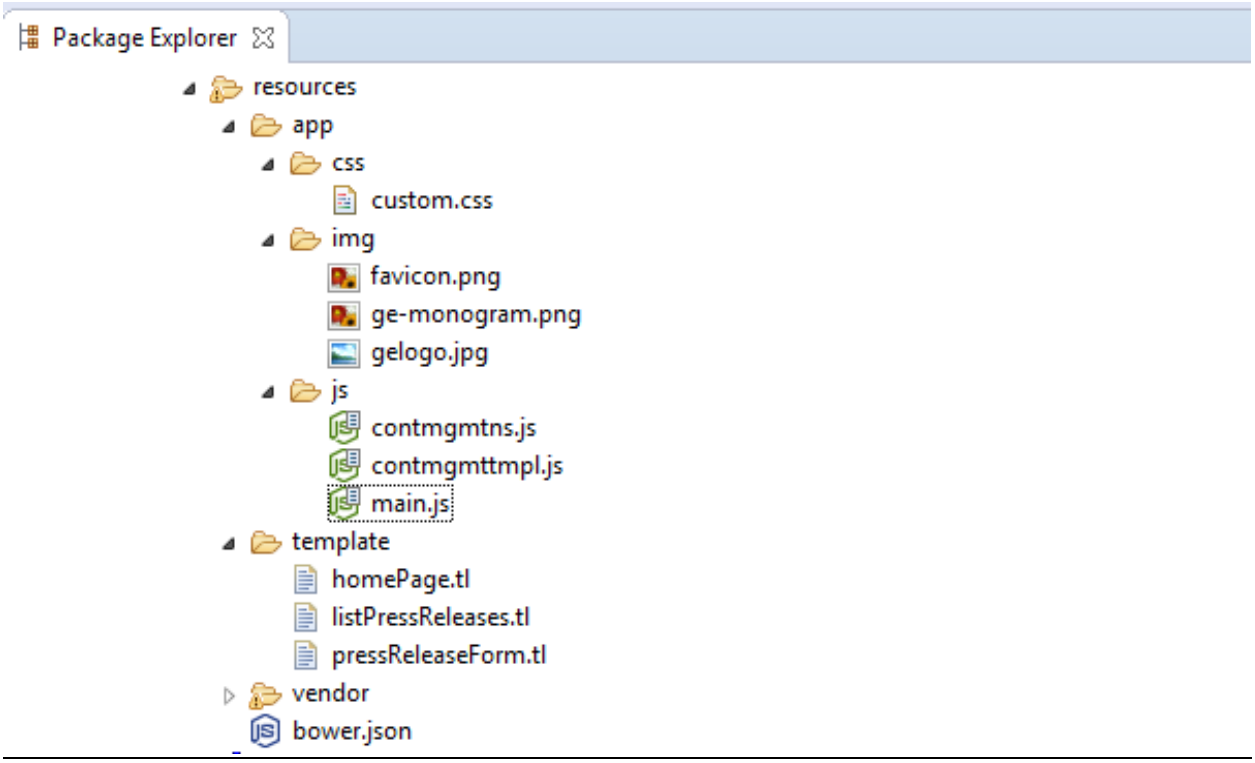
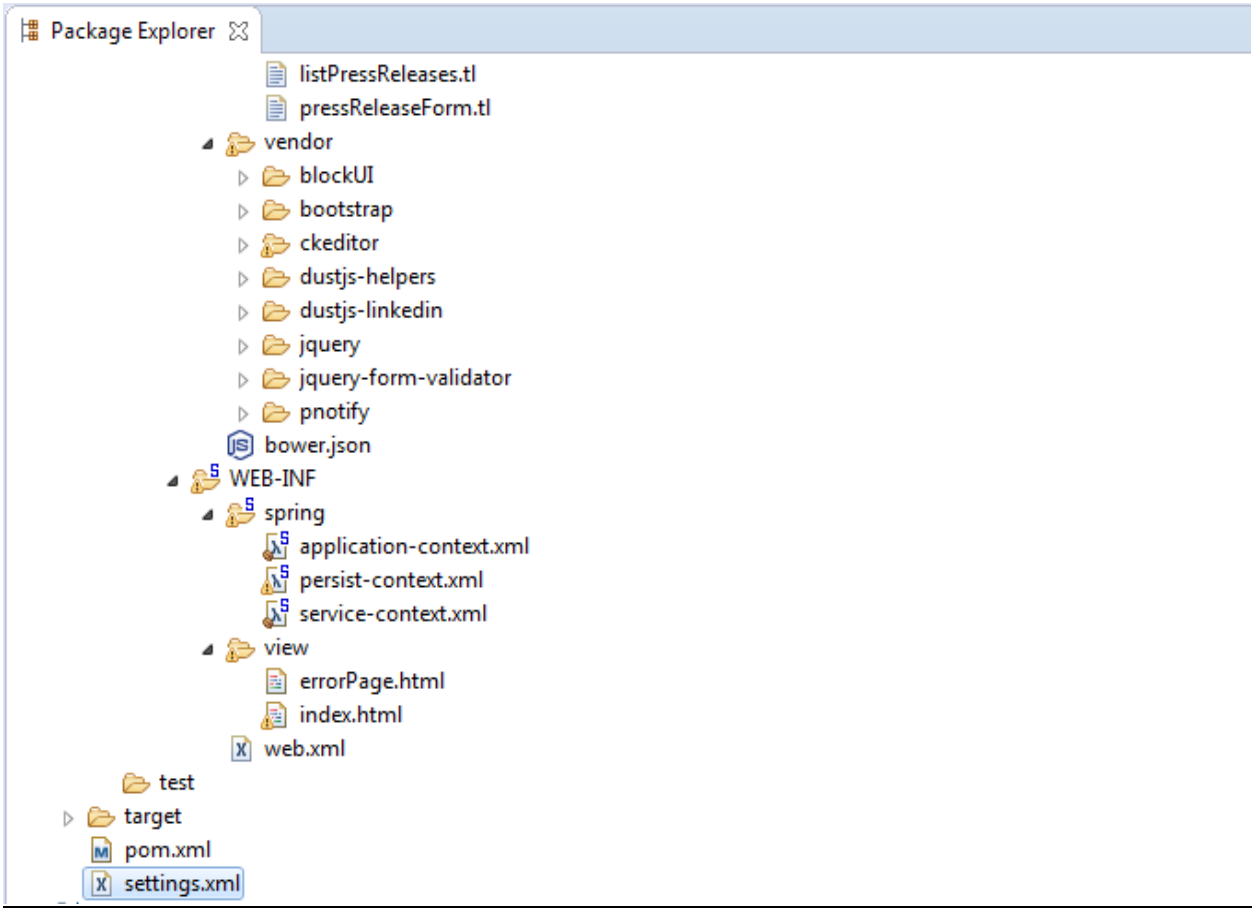
Project Structure






Package Explorer
resources
app
css
custom.css
img
favicon.png
ge-monogram.png
gelogo.jpg
js
contmgmtns.js
contmgmttmpl.js
main.js
template
homePage.tl
listPressReleases.tl
pressReleaseForm.tl
vendor
bower.json


Package Explorer
listPressReleases.tl
pressReleaseForm.tl
vendor
blockUI
bootstrap
ckeditor
dustjs-helpers
dustjs-linkedin
jquery
jquery-form-validator
pnotify
bower.json
WEB-INF
spring
application-context.xml
persist-context.xml
service-context.xml
view
errorPage.html
index.html
web.xml
test
target
pom.xml
settings.xml

The HTML

index.html

```
<!DOCTYPE html>
<html>
<head>
	<title>EFS Content Management</title>
	<meta charset="utf-8">
	<meta http-equiv="X-UA-Compatible" content="IE=edge">
	<meta name="viewport" content="width=device-width,initial-scale=1">
	<!-- The following code will write a base tag with dynamic base Url -->
	<!-- <base href="http://localhost:8080/efscontmgmt/"> -->
	<!--
	<script type="text/javascript">
		if (typeof window.location.origin === 'undefined')
			window.location.origin = window.location.protocol + '//' +
window.location.host;
		var baseUrl = window.location.origin + "/" +
window.location.pathname.split("/")[1] + "/";
		document.write('<base href="' + baseUrl + '">');
	</script>
	-->
	<!-- At the time of deploying the app in server, we need to do URL Rewriting
at the Apache Server
		level to route the requests as below:
		http://localhost:8080 -> http://localhost:8080/efscontmgmt/
		http://localhost:8080/efscontmgmt -> http://localhost:8080/efscontmgmt/
	-->
	<script src="resources/vendor/jquery/dist/jquery.min.js"></script>
    <script src="resources/vendor/blockUI/jquery.blockUI.js"></script>
	<script src="resources/vendor/bootstrap/dist/js/bootstrap.min.js"></script>
	<script src="resources/vendor/dustjs-linkedin/dist/dust-core.min.js"></script>
    <script src="resources/vendor/dustjs-helpers/dist/dust-helpers.min.js"></script>
    <script src="resources/vendor/jquery-form-validator/form-validator/jquery.form-
validator.min.js"></script>
    <script src="resources/vendor/pnotify/pnotify.custom.min.js"></script>
    <script src="resources/vendor/ckeditor/ckeditor.js"></script>
    <script src="resources/vendor/ckeditor/adapters/jquery.js"></script>
	<script src="resources/app/js/contmgmttmpl.js"></script>
    <script src="resources/app/js/contmgmtns.js"></script>
	<link href="resources/vendor/bootstrap/dist/css/bootstrap.css"
rel="stylesheet">
    <link href="resources/vendor/pnotify/pnotify.custom.min.css" rel="stylesheet">
	<link href="resources/app/css/custom.css" rel="stylesheet">
	<link rel="icon" href="resources/app/img/favicon.png">
</head>
<body>
	<nav class="navbar navbar-inverse navbar-fixed-top">
	  <div class="container-fluid">
	    <!-- Brand and toggle get grouped for better mobile display -->
	    <div class="navbar-header">

	      <a class="navbar-brand" href="#">
```

```
                    <span><img class="main-logo" src="resources/app/img/ge-
monogram.png" style="display: inline-block;"></span>
                    <span style="display: inline-block;">EFS Content
Management</span>
                </a>

              <button type="button" class="navbar-toggle collapsed" data-
toggle="collapse" data-target="#bs-example-navbar-collapse-1" aria-expanded="false">
                <span class="sr-only">Toggle navigation</span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
              </button>
            </div>

            <!-- Collect the nav links, forms, and other content for toggling -->
            <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
              <ul class="nav navbar-nav navbar-right">
                <li><a href="#">Home <span class="sr-only">(current)</span></a></li> <!-
-class="active"-->
                    <li class="dropdown">
                      <a href="#PostContent" class="dropdown-toggle" data-
toggle="dropdown" role="button" aria-haspopup="true" aria-expanded="false">Post
Content <span class="caret"></span></a>
                    <ul class="dropdown-menu">
                    <li class="dropdown-header">Post External</li>
                      <li><a href="#PressReleases">Press Release</a></li>
                       <li><a href="#Infographic">Infographic</a></li>
                    <li class="dropdown-header">Post Internal</li>
                    <li><a target="_blank" href="http://gecms.ge.com/ige/">Weekly
Coverage</a></li>
                    <li><a target="_blank" href="http://gecms.ge.com/ige/">Press
Release</a></li>
                    </ul>
                  </li>
                  <li class="dropdown">
                    <a href="#ReviewContent" class="dropdown-toggle" data-
toggle="dropdown" role="button" aria-haspopup="true" aria-expanded="false">Review
Content <span class="caret"></span></a>
                    <ul class="dropdown-menu">
                      <li><a target="_blank"
href="http://geenergyfinancialservices.com/press_releases">EFS External</a></li>
                      <li><a target="_blank" href="https://my.ge.com/card/ge-news">EFS
Internal</a></li>
                      <!-- <li><a target="_blank" href="http://sc.ge.com/*efshub">EFS
Hub</a></li> -->
                      <!-- <li role="separator" class="divider"></li> -->
                    </ul>
                  </li>
                       <li><a href="#Logout">Logout</a></li>
                  <li><a href="#Feedback">Feedback</a></li>
            </ul>
              </div><!-- /.navbar-collapse -->
            </div><!-- /.container-fluid -->
          </nav>
```

```
       <div id="mainContent" class="container">
      <!-- Main component for a primary marketing message or call to action -->
        <!-- This div will be replaced using a template -->

    </div> <!-- /container -->

    <!--Bootstrap modal windows-->
    <div id="modalContent">
        <!--TODO: Move this to a template -->
        <div id="deleteModal" class="modal fade" tabindex="-1" role="dialog">
          <div class="modal-dialog" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close"><span aria-hidden="true">&times;</span></button>
                <h3 class="modal-title">Confirmation Page</h3>
              </div>
              <div class="modal-body">
                <p>Do you want to delete this record&quest;</p>
                <input id="pressRelDelId" type="text" hidden>
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-default" data-
dismiss="modal">Close</button>
                <button id="deletePr" type="button" class="btn btn-
primary">Delete</button>
              </div>
            </div><!-- /.modal-content -->
          </div><!-- /.modal-dialog -->
        </div><!-- /.modal -->

        <div id="copyModal" class="modal fade bs-example-modal-sm" tabindex="-1"
role="dialog" aria-labelledby="mySmallModalLabel">
          <div class="modal-dialog modal-sm" role="document">
            <div class="modal-content">
              <div class="modal-header">
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close"><span aria-hidden="true">&times;</span></button>
                <h4 class="modal-title" id="myModalLabel">Copy URL/Link</h4>
              </div>
              <div class="modal-body">
                <input id="cpyText" type="text" class="form-control">
              </div>
            </div>
          </div>
        </div>

    </div> <!-- End of modals -->

    <script src="resources/app/js/main.js"></script>
</body>
</html>
```

errorPage.html

```
<!DOCTYPE html>
<html>
<head>
	<title>Error Page</title>
</head>
<body>
	<h2>This is an error page</h2>
</body>
</html>
```

The Templates

homePage.tl

```
<!-- We don't need a script tag here as we are compiling it externally through dustc
-->
<!-- <script type="text/template" id="homePage"> -->
	<div class="jumbotron" id="homePage">
	    <h2>EFS Content Management</h2>
	    <br>
	    <p>This is a one stop shop to manage all contents of EFS specific
websites.</p>
	    <p>Powerful and flexible to post edit and delete content to external as
well as internal content websites.</p>
	    <p>Reduces manual effort and dependecy of IT operation team.</p>
	    <br>
	    <p>
	      <a class="btn btn-lg btn-primary" href="#PressReleases"
role="button">Get Started &raquo;</a>
	    </p>
	</div>
<!-- </script> -->
```

listPressReleases.tl

```
<div class=jumbotron id="listPressRelease">
<h2>Latest 5 Press Releases</h2>
<br>
<div class="table-responsive">
    <table class="table table-bordered table-striped table-hover">
        <thead>
            <tr>
                <th class="col-sm-2">Published Date</th>
                <th class="col-sm-3">Press Release Name</th>
                <th class="col-sm-2">Industry Focus</th>
                <th class="col-sm-2">Location</th>
                <th class="col-sm-3">Action</th>
            </tr>
        </thead>
        <tbody>
        {#pressReleases}
            <tr>
                <td>{publishDate}</td>
                <td>{name}</td>
                <td>{industryFocus}</td>
                <td>{location}</td>
```

```
                <td>
                  <div class="btn-group btn-group-sm" role="group" aria-label="...">
                    <button id="btnView{pressReleaseId}" type="button" class="btn
btn-primary">View <span class="glyphicon glyphicon-list-alt" aria-
hidden="true"></span></button>
                    <button id="btnEdit{pressReleaseId}" type="button" class="btn
btn-primary">Edit <span class="glyphicon glyphicon-pencil" aria-
hidden="true"></span></button>
                    <button id="btnDelete{pressReleaseId}" type="button" class="btn
btn-primary">Delete <span class="glyphicon glyphicon-trash" aria-
hidden="true"></span></button>
                  </div>
                </td>
            </tr>
        {/pressReleases}
        </tbody>
    </table>
</div>
<br>
    <button id="createNewPressRelease" class="btn btn-primary page-action"
type="button"><span class="glyphicon glyphicon-plus" aria-hidden="true"></span>
Create New</button>
    <button id="cancelPressReleasesView" class="btn btn-primary page-action"
type="button"><span class="glyphicon glyphicon-arrow-left" aria-hidden="true"></span>
Go Back</button>
</div>
```

pressReleaseForm.tl

```
<div class=jumbotron id="pressReleaseForm">
<!-- dustc --pwd=template/ template/*.tl -o app/js/contmgmttmpl.js -->
  {@select key=mode}
    {@eq value="view"}<h2 id="formHeader">View Press Release</h2>{/eq}
    {@eq value="create"}<h2 id="formHeader">Create Press Release</h2>{/eq}
    {@eq value="edit"}<h2 id="formHeader">Edit Press Release</h2>{/eq}
  {/select}
  <br>
  <form id="pressRlsForm" class="form-horizontal">
    <div class="form-group publishDate {@ne key=mode value="view"}hidden{/ne}">
      <label for="publishDate" class="col-sm-2 control-label">Publish Date</label>
      <div class="col-sm-10">
        <input type="text" class="form-control" id="publishDate"
value="{pressRelease.publishDate}" {@eq key=mode value="view"}disabled{/eq}>
      </div>
    </div>
    <div class="form-group">
      <label for="pressReleaseName" class="col-sm-2 control-label">Press Release
Name</label>
      <div class="col-sm-10">
        <input type="text" class="form-control" id="pressReleaseName"
value="{pressRelease.name}" placeholder="Press Release Name" {@eq key=mode
value="view"}disabled{/eq} data-validation="required">
      </div>
    </div>
    <div class="form-group">
```

```
      <label for="industryFocus" class="col-sm-2 control-label">Industry
Focus</label>
      <div class="col-sm-10">
        {@eq key=mode value="view"}
        <input type="text" class="form-control" id="pressReleaseName"
value="{pressRelease.industryFocus}" disabled>
        {:else}
        <select class="form-control" id="industryFocus" data-validation="required">
            <option value="">Select One</option>
            {#industryFoci}
            <option value={id} {@eq key=id value=pressRelease.industryFocusId}
selected{/eq}>{name}</option>
            {/industryFoci}
        </select>
        {/eq}
      </div>
    </div>
    <div class="form-group">
      <label for="location" class="col-sm-2 control-label">location</label>
      <div class="col-sm-10">
        <input type="text" class="form-control" id="location"
value="{pressRelease.location}" placeholder="Location" {@eq key=mode
value="view"}disabled{/eq} data-validation="length" data-validation-length="max100">
      </div>
    </div>
    <div class="form-group">
      <label for="pressBody" class="col-sm-2 control-label">Press Body</label>
      <div class="col-sm-10">
        <textarea class="form-control" rows="5" id="pressBody" placeholder="Press
Body" {@eq key=mode value="view"}disabled{/eq}>{pressRelease.body}</textarea>
      </div>
    </div>
    <div class="form-group">
      <label for="contactInfo" class="col-sm-2 control-label">Contact Info</label>
      <div class="col-sm-10">
        <textarea class="form-control" rows="3" id="contactInfo" placeholder="Contact
Info" {@eq key=mode value="view"}disabled{/eq}>{pressRelease.contactInfo}</textarea>
      </div>
    </div>
    <div class="form-group">
      <label for="quote" class="col-sm-2 control-label">Quote</label>
      <div class="col-sm-10">
        <textarea class="form-control" rows="3" id="quote" placeholder="Quote" {@eq
key=mode value="view"}disabled{/eq}>{pressRelease.quote}</textarea>
      </div>
    </div>
    <div class="form-group">
      <label for="quoteAuthor" class="col-sm-2 control-label">Quote Author</label>
      <div class="col-sm-10">
        <input type="text" class="form-control" id="quoteAuthor"
value="{pressRelease.quoteAuthor}" placeholder="Quote Author" {@eq key=mode
value="view"}disabled{/eq} data-validation="length" data-validation-length="max255">
      </div>
    </div>
    <div class="form-group">
```

```
      <label for="quoteAuthorTitle" class="col-sm-2 control-label">Quote Author
Title</label>
      <div class="col-sm-10">
        <input type="text" class="form-control" id="quoteAuthorTitle"
value="{pressRelease.quoteAuthorTitle}" placeholder="Quote Author Title" {@eq
key=mode value="view"}disabled{/eq} data-validation="length" data-validation-
length="max255">
      </div>
    </div>
    {@eq key=mode value="view"}
    <div class="form-group">
      <label class="col-sm-2 control-label" for="inputFile">Press Release PDF</label>
      <p class="fileInput col-sm-10" id="inputFile">{pressRelease.pdfUrl}</p>
    </div>
    <div class="form-group">
      <label class="col-sm-2 control-label" for="inputImg">Press Release
Image</label>
      <p class="fileInput col-sm-10" id="inputImg">{pressRelease.imageUrl}</p>
    </div>
    {:else}
    <div class="form-group">
      <label class="col-sm-2 control-label" for="inputFile">Press Release PDF</label>
      <div class="col-sm-1">
        <button id="cpyFile" type="button" class="btn btn-default btn-xs"><span
class="glyphicon glyphicon-copy" aria-hidden="true"></span> Copy Link</button>
      </div>
      <input class="fileInput col-sm-9" type="file" id="inputFile"
             data-validation="mime size"
             data-validation-allowing="pdf"
             data-validation-max-size="25M">
    </div>
    <div class="form-group">
      <label class="col-sm-2 control-label" for="inputImg">Preess Release
Image</label>
      <div class="col-sm-1">
        <button id="cpyImg" type="button" class="btn btn-default btn-xs"><span
class="glyphicon glyphicon-copy" aria-hidden="true"></span> Copy Link</button>
      </div>
      <input class="fileInput col-sm-9" type="file" id="inputImg"
             data-validation="mime size"
             data-validation-allowing="jpg, png, gif"
             data-validation-max-size="25M">
    </div>
    {/eq}
    <div class="form-group">
      <label class="col-sm-2 control-label" for="activeFlag">Active</label>
        <div class="checkbox col-sm-10">
          <label>
            <input id="activeFlag" type="checkbox" {@eq key=mode
value="view"}disabled{/eq} {@eq key=pressRelease.active value=1}checked{/eq}>
          </label>
        </div>
    </div>
    <br>
    {@eq key=mode value="view"}
```

```
    <button id="pageActionEdit" class="btn btn-primary page-action"
type="button"><span class="glyphicon glyphicon glyphicon-pencil" aria-
hidden="true"></span> Edit</button>
    {:else}
    <button id="pageActionSubmit" class="btn btn-primary page-action"
type="button"><span class="glyphicon glyphicon-floppy-disk" aria-
hidden="true"></span> Submit</button>
    {/eq}
    <button id="pageActionCancel" class="btn btn-primary page-action"
type="button"><span class="glyphicon glyphicon-remove" aria-hidden="true"></span>
Cancel</button>
  </form>
</div>
```

The JavaScript

contmgmtns.js

```
var Cmgt = Cmgt || {};
Cmgt.namespace = function(ns_string) {
    var parts = ns_string.split('.'), parent = Cmgt, i;
    if (parts[0] == "Cmgt") {
        parts = parts.slice(1);
    }
    pl = parts.length;
    for (i = 0; i < pl; i++) {
        //create a property if it doesnt exist
        if (typeof parent[parts[i]] == 'undefined') {
            parent[parts[i]] = {};
        }
        parent = parent[parts[i]];
    }
    return parent;
};

if (typeof window.location.origin === 'undefined')
            window.location.origin = window.location.protocol + '//' +
window.location.host;

Cmgt.contextRoot = window.location.origin + "/" +
window.location.pathname.split("/")[1] + "/";
```

contmgmttmpl.js

This should be generated by compiling the template source present in the /efscontmgmt/src/main/webapp/resources/template folder

```
dustc --pwd=template/ template/*.tl -o app/js/contmgmttmpl.js
```

main.js

```
//Press Release Scripts
//Author: Prabhu Purohit

Cmgt.namespace("Cmgt.Prl");
Cmgt.Prl = {
```

```
    //TODO: Move this to the Cmgt.Util namespace later
    util: {
        getDate: function() {
            var d = new Date();
            var date = d.getFullYear() + '-' +
              ('0' + (d.getMonth() + 1)).slice(-2) + '-' +
              ('0' + d.getDate()).slice(-2);
            return date;
        },

        convertNullToStr: function(pressReleases) {
            pressReleases.forEach(function(elem,indx,arr){
                //industry focus id is nullable, so replace it with empty string
                elem.industryFocusId = elem.industryFocusId ? elem.industryFocusId :
5;
                //The following fields will be replaced with emty string if null
                elem.location = elem.location ? elem.location : "";
                elem.body = elem.body ? elem.body : "";
                elem.contactInfo = elem.contactInfo ? elem.contactInfo : "";
                elem.quote = elem.quote ? elem.quote : "";
                elem.quoteAuthor = elem.quoteAuthor ? elem.quoteAuthor : "";
                elem.quoteAuthorTitle = elem.quoteAuthorTitle ? elem.quoteAuthorTitle
: "";
                //For image and pdf url we will set it to null so that it doesn't get
updated
                //in DB with empty string
                elem.imageUrl = elem.imageUrl ? elem.imageUrl : null;
                elem.pdfUrl = elem.pdfUrl ? elem.pdfUrl : null;
                elem.industryFocus = elem.industryFocus ? elem.industryFocus : "";
            });
            return pressReleases;
        },

        compareForChanges: function(prFormData) {
            var compareTo = Cmgt.Prl.model.currentPressRelease;
            var changed = false;
            if ((prFormData.industryFocusId !== compareTo.industryFocusId) ||
                (prFormData.name !== compareTo.name) ||
                (prFormData.location !== compareTo.location) ||
                (prFormData.body !== compareTo.body) ||
                (prFormData.contactInfo !== compareTo.contactInfo) ||
                (prFormData.quote !== compareTo.quote) ||
                (prFormData.quoteAuthor !== compareTo.quoteAuthor) ||
                (prFormData.quoteAuthorTitle !== compareTo.quoteAuthorTitle) ||
                (prFormData.active !== compareTo.active)) {
                    changed = true;
                }
            //If the image and pdf url are not null, user has uploaded the file
            if (prFormData.imageUrl || prFormData.pdfUrl) {
                changed = true;
            }
            return changed;
        }
    },
```

```
    model: {

        //File upload paths
        imgUplPath: "/img/press-releases/",
        pdfUplPath: "/files/press-releases/",

        //Latest press releases
        pressReleaseData : {},

        //Current press release id
        currentPressReleaseId : 0,

        //This will be used for comparison against form data before sending edit
request
        currentPressRelease : {},

        //Form data to be exchanged
        formData : {
            mode: '',
            pressRelease: {},
            industryFoci: []
        },

        getLatestPressReleases : function(){
            var contextRoot = "http://localhost:8080/efscontmgmt";
            var url = contextRoot + "/pressreleases/latest5";
            //var url = Cmgt.contextRoot + "/pressreleases/latest5";

            $.ajax({
                type: "GET",
                url: url,

                beforeSend: function(){
                    $.blockUI();
                },
                success: function (pressRlsData) {
                    if (pressRlsData.message !== "0000") {
                        Cmgt.Prl.exHandler.handleError(pressRlsData,null,null,null);
                    } else {
                        Cmgt.Prl.model.pressReleaseData.pressReleases =
                        Cmgt.Prl.util.convertNullToStr(pressRlsData.payload);
                        Cmgt.Prl.view.renderLatestPressReleasespage();
                        Cmgt.Prl.view.registerEvents();
                        Cmgt.Prl.model.getIndustryFoci();
                        console.log("SUCCESSFULLY GET PRESS RELEASES");
                        Cmgt.Prl.view.sendNotification("success", "Success: GET PRESS
RLS", pressRlsData.msgDesc);
                    }
                },
                error: function (jqXHR, textStatus, errorThrown) {
                    Cmgt.Prl.exHandler.handleError(null, jqXHR, textStatus,
errorThrown);
                },
                complete: function() {
```

```
                    console.log("GET PRESS RELEASE COMPLETE");
                    $.unblockUI();
                }

            });
        },

        getIndustryFoci: function() {
            var contextRoot = "http://localhost:8080/efscontmgmt";
            var url = contextRoot + "/industryfoci/all";
            $.ajax({
                type: "GET",
                url: url,
                success: function (indFocData) {
                    if (indFocData.message !== "0000") {
                        Cmgt.Prl.exHandler.handleError(indFocData,null,null,null);
                    } else {
                        Cmgt.Prl.model.formData.industryFoci = indFocData.payload;
                        console.log("SUCCESSFULLY GET INDUSTRY FOCI");
                        //Cmgt.Prl.view.sendNotification("success", "Success: GET IND
FOCI", indFocData.msgDesc);
                    }
                },
                error: function (jqXHR, textStatus, errorThrown) {
                    Cmgt.Prl.exHandler.handleError(null, jqXHR, textStatus,
errorThrown);
                },
                complete: function() {
                    console.log("GET INDUSTRY FOCI COMPLETE");
                }
            });
        }

    },

    view: {

        renderHomePage: function(){
            dust.render('homePage', {}, function(err, out) {
                        $("#mainContent").html(out);
            });

            //Have pNotify adopt Bootstrap styling
            PNotify.prototype.options.styling = "bootstrap3";
        },

        renderPressReleaseFormPage: function(formData) {
            //Compiled with dustc compiler
            //dustc --pwd=template/ template/*.tl -o app/js/contmgmttmpl.js
            dust.render('pressReleaseForm', formData, function(err, out) {
                        $("#mainContent").html(out);
            });
            //Remove placeholder attributes from the view page
            if (formData.mode === "view") {
                $("form input, form textarea").removeAttr('placeholder')
```

```
            }
            //The follwing code is not required as we are setting
            //the selected value dynamically in the dust template

            // if (formData.mode === "edit") {
            //
$("#industryFocus").val(formData.pressRelease.industryFocusId).prop('selected',true);
            // }

            $('#pressBody').ckeditor();
            $('#contactInfo').ckeditor();

        },

        renderLatestPressReleasespage: function() {

            dust.render('listPressReleases', Cmgt.Prl.model.pressReleaseData,
function(err, out) {
                $("#mainContent").html(out);
            });
        },

        registerEvents: function() {
            var btnViewId, btnEditId, btnDeleteId;

Cmgt.Prl.model.pressReleaseData.pressReleases.forEach(function(elem,indx,arr){
                btnViewId = "#btnView"+elem.pressReleaseId;
                $(btnViewId).click(function(e){
                    e.preventDefault();
                    window.location.hash =
'#'+'PressReleases/PressRelease/'+elem.pressReleaseId;
                });
                btnEditId = "#btnEdit"+elem.pressReleaseId;
                $(btnEditId).click(function(e){
                    e.preventDefault();
                    window.location.hash =
'#'+'PressReleases/PressRelease/'+elem.pressReleaseId+'?action=edit';
                });
                btnDeleteId = "#btnDelete"+elem.pressReleaseId;
                $(btnDeleteId).click(function(e){
                    e.preventDefault();
                    Cmgt.Prl.model.currentPressReleaseId = elem.pressReleaseId;
                    $("#pressRelDelId").val(elem.pressReleaseId);
                    $("#deleteModal").modal('show');
                })

            });

            $("#createNewPressRelease").click(function(e){
                e.preventDefault();
                window.location.hash =
'#'+'PressReleases/PressRelease?action=create';
            });
            $("#cancelPressReleasesView").click(function(e){
                e.preventDefault();
```

```
                window.location.hash = '#';
            });

            $('#deletePr').click(function(e) {
                e.preventDefault();
                window.location.hash = '#'+'PressReleases/PressRelease/'+
$("#pressRelDelId").val() +'?action=delete';
                $('#deleteModal').modal('hide');
            })
        },

        registerFormEvents: function(pressReleaseId, mode) {
            switch(mode) {

            case "view":

                $("#pageActionEdit").click(function(e){
                e.preventDefault();
                window.location.hash =
'#'+'PressReleases/PressRelease/'+pressReleaseId+'?action=edit';
                });
                break;

            case "create":

                $("#pageActionSubmit").click(function(e){
                    e.preventDefault();
                    //Ajax POST request
                    if (Cmgt.Prl.controller.isValidForm()) {
                        Cmgt.Prl.controller.submitCreateForm();
                    }
                });
                break;

            case "edit":

                $("#pageActionSubmit").click(function(e){
                    e.preventDefault();
                    if (Cmgt.Prl.controller.isValidForm()) {
                        Cmgt.Prl.controller.submitEditForm(pressReleaseId);
                    }
                });
                break;
            }

            $("#cpyFile").click(function(e){
                e.preventDefault();
                var pdfUrl;
                $("#copyModal").modal('show');
                 if($("#inputFile")[0].files[0]) {
                    pdfUrl = Cmgt.Prl.model.pdfUplPath +
$("#inputFile")[0].files[0].name;
                } else {
                    pdfUrl = Cmgt.Prl.model.pdfUplPath;
                }
```

```
                $("#cpyText").val(pdfUrl);
            });

            $("#cpyImg").click(function(e){
                e.preventDefault();
                var imgUrl;
                $("#copyModal").modal('show');
                if($("#inputImg")[0].files[0]) {
                    imgUrl = Cmgt.Prl.model.imgUplPath +
$("#inputImg")[0].files[0].name;
                } else {
                    imgUrl = Cmgt.Prl.model.imgUplPath;
                }
                $("#cpyText").val(imgUrl);
            });

            $("#cpyText").click(function(e){
                e.preventDefault();
                $("#cpyText").focus();
                $("#cpyText").select();
            });

            $("#pageActionCancel").click(function(e){
                e.preventDefault();
                window.location.hash = '#'+'PressReleases';
            });
        },

        registerFormValidator: function() {
            $.validate({
                form : '#pressRlsForm',
                modules: 'file',
                validateOnBlur : true,
                validateOnEvent : true,
                onModulesLoaded: function() {
                    console.log('All modules loaded!');
                },
                onSuccess: function() {
                    console.log('Success!');
                },
                onValidate: function() {
                    console.log('Validation starts!');
                },
                onError: function() {
                    console.log('Error!');
                }
            });
        },

        sendNotification: function(notiType,notiTitle,message) {
            new PNotify({
                title: notiTitle,
                width: "500px",
                delay: 3000,
                text: message,
```

```
                type: notiType
            });
        }
    },

    exHandler: {

        //For the error message
        errCode: "",
        errSev: "",
        errMsg: "",

        //Message codes
        msgSuccess: "0000",
        msgFatal: "9999",

        handleError: function(respData, jqXHR, textStatus, errorThrown) {

            if (respData) {
                console.log("FAILURE: "+respData.msgDesc);
                Cmgt.Prl.view.sendNotification("error", "Error: "+respData.message,
"Message: "+respData.severity+": "+respData.msgDesc);
            }
            else if (jqXHR.readyState === 0) {
                console.log("FAILURE: Network");
                Cmgt.Prl.view.sendNotification("error", "Error: NETWORK", "Message:
NOT ABLE TO MAKE A NETWORK CONNECTION");
            }
            else if (jqXHR.responseText) {
                var errorData = $.parseJSON(jqXHR.responseText);
                console.log("FAILURE: "+errorData.msgDesc);
                Cmgt.Prl.view.sendNotification("error", "Error: "+errorData.message,
"Message: "+errorData.severity+": "+errorData.msgDesc);
            }
            else {
                console.log("FAILURE: "+errorThrown);
                Cmgt.Prl.view.sendNotification("error", "Error: HTTP", "Message:
"+errorThrown);
            }
        }

    },

    controller: {

        uploadFiles: function() {
            var contextRoot = "http://localhost:8080/efscontmgmt";
            var url = contextRoot + "/upload";
            var params = new FormData();
            if ($("#inputImg")[0].files[0]) {
                params.append("files", $("#inputImg")[0].files[0]);
            }
            if ($("#inputFile")[0].files[0]) {
                params.append("files", $("#inputFile")[0].files[0]);
            }
```

```
            $.ajax({
                url: url,
                type: "POST",
                data: params,
                enctype: 'multipart/form-data',
                processData: false,
                contentType: false,
                success: function (data) {
                    if (data.message !== "0000") {
                        Cmgt.Prl.exHandler.handleError(data,null,null,null);
                    } else {
                        console.log("SUCCESSFULLY UPLOADED");
                        Cmgt.Prl.view.sendNotification("success", "Success: UPLOAD",
data.msgDesc);
                    }
                },
                error: function (jqXHR, textStatus, errorThrown) {
                    Cmgt.Prl.exHandler.handleError(null, jqXHR, textStatus,
errorThrown);
                },
                complete: function() {
                    console.log("UPLOAD FUNCTION COMPLETE");
                }
            });
        },

        isValidForm: function() {
            //Invoke the jQuery Form Validator
            return $("#pressRlsForm").isValid();
        },

        getLatestFormValues: function() {
            return {
                "pressReleaseId": Cmgt.Prl.model.currentPressReleaseId,
                "industryFocusId": parseInt($("#industryFocus").val().trim()),
                "name": $("#pressReleaseName").val().trim(),
                "location": $("#location").val().trim(),
                "publishDate": Cmgt.Prl.util.getDate(),
                "body": $("#pressBody").val().trim(),
                "contactInfo": $("#contactInfo").val().trim(),
                "quote": $("#quote").val().trim(),
                "quoteAuthor": $("#quoteAuthor").val().trim(),
                "quoteAuthorTitle": $("#quoteAuthorTitle").val().trim(),
                "imageUrl": (function(){
                    var imgUrl;
                    if($("#inputImg")[0].files[0]) {
                        imgUrl = Cmgt.Prl.model.imgUplPath +
$("#inputImg")[0].files[0].name;
                    } else {
                        imgUrl = null;
                    }
                    return imgUrl;
                })(),
```

```
                "pdfUrl": (function(){
                    var pdfUrl;
                    if($("#inputFile")[0].files[0]) {
                        pdfUrl = Cmgt.Prl.model.pdfUplPath +
$("#inputFile")[0].files[0].name;
                    } else {
                        pdfUrl = null;
                    }
                    return pdfUrl;
                })(),

                "active": $("#activeFlag").prop('checked')? 1:0,
                "industryFocus": $("#industryFocus option:selected").text().trim()
            };
        },

        submitCreateForm : function() {
            var contextRoot = "http://localhost:8080/efscontmgmt";
            var url = contextRoot + "/pressreleases/pressrelease?action=create";
            var params = Cmgt.Prl.controller.getLatestFormValues();
            Cmgt.Prl.model.formData.pressRelease = params;

            $.ajax({
                type: "POST",
                url: url,
                data: JSON.stringify(params),
                contentType: "application/json",
                //Handling CORS POST requests
                // xhrFields: {
                //     withCredentials: true
                // },
                beforeSend: function(){
                    //Make the background screen fade away
                    $.blockUI();
                },
                success: function (data) {
                    if (data.message !== "0000") {
                        Cmgt.Prl.exHandler.handleError(data,null,null,null);
                    } else {
                        if (params.imageUrl || params.pdfUrl) {
                            Cmgt.Prl.controller.uploadFiles();
                        }
                        console.log("SUCCESSFULLY CREATED");
                        Cmgt.Prl.view.sendNotification("success", "Success: CREATE",
data.msgDesc);
                        //Show the List page
                        window.location.hash = '#PressReleases';
                    }
                },
                error: function (jqXHR, textStatus, errorThrown) {
                    Cmgt.Prl.exHandler.handleError(null, jqXHR, textStatus,
errorThrown);
                },
                complete: function() {
                    console.log("CREATE FUNCTION COMPLETE");
```

```
                    $.unblockUI();
                }

            });
        },

        submitEditForm : function(pressReleaseId) {
            var contextRoot = "http://localhost:8080/efscontmgmt";
            var url = contextRoot +
"/pressreleases/pressrelease/"+pressReleaseId+"?action=edit";
            var params = Cmgt.Prl.controller.getLatestFormValues();
            Cmgt.Prl.model.formData.pressRelease = params;

            if
(!Cmgt.Prl.util.compareForChanges(Cmgt.Prl.model.formData.pressRelease)) {
                console.log("Nothing Changed");
                //Image is tricky as if nothing gets uploaded, it should be fine
                Cmgt.Prl.view.sendNotification("error", "Error!", "Nothing Changed");
            } else {

                $.ajax({
                    type: "POST",
                    url: url,
                    data: JSON.stringify(params),
                    contentType: "application/json",
                    //Handling CORS POST requests
                    // xhrFields: {
                    //     withCredentials: true
                    // },
                    beforeSend: function(){
                        $.blockUI();
                    },
                    success: function (data) {
                        if (data.message !== "0000") {
                            Cmgt.Prl.exHandler.handleError(data,null,null,null);
                        } else {
                            if (params.imageUrl || params.pdfUrl) {
                                Cmgt.Prl.controller.uploadFiles();
                            }
                            console.log("SUCCESSFULLY EDITED");
                            Cmgt.Prl.view.sendNotification("success", "Success:
EDIT", data.msgDesc);
                            //Show the List page
                            window.location.hash = '#PressReleases';
                        }
                    },
                    error: function (jqXHR, textStatus, errorThrown) {
                        Cmgt.Prl.exHandler.handleError(null, jqXHR, textStatus,
errorThrown);
                    },
                    complete: function() {
                        console.log("EDIT FUNCTION COMPLETE");
                        $.unblockUI();
                    }
```

```
                });
            }

        },

        deletePressRelease : function(pressReleaseId) {
            var contextRoot = "http://localhost:8080/efscontmgmt";
            var url = contextRoot +
"/pressreleases/pressrelease/"+pressReleaseId+"?action=delete";
            $.ajax({
                type: "POST",
                url: url,
                data: JSON.stringify({}),
                contentType: "application/json",
                //Handling CORS POST requests
                // xhrFields: {
                //     withCredentials: true
                // },
                beforeSend: function(){
                    $.blockUI();
                },
                success: function (data) {
                    if (data.message !== "0000") {
                        Cmgt.Prl.exHandler.handleError(data,null,null,null);
                    } else {
                        console.log("SUCCESSFULLY DELETED");
                        Cmgt.Prl.view.sendNotification("success", "Success: DELETE",
data.msgDesc);
                        window.location.hash = '#PressReleases';
                    }
                },
                error: function (jqXHR, textStatus, errorThrown) {
                    Cmgt.Prl.exHandler.handleError(null, jqXHR, textStatus,
errorThrown);
                },
                complete: function() {
                    console.log("DELETE FUNCTION COMPLETE");
                    $.unblockUI();
                }

            });
        },

        route: function (url) {
            // Get the keyword from the url.
            var temp = url.split('/')[0];
            // Hide whatever page is currently shown.
            //$('.mainContent').removeClass('visible');
            $('#mainContent').empty();

            var map = {

                // The Homepage.
                '': function() {
                    Cmgt.Prl.view.renderHomePage();
```

```
                },

                // The Homepage. This is for IE compatibility.
                '#': function() {
                    Cmgt.Prl.view.renderHomePage();
                },

                // Press Release page.
                '#PressReleases': function() {

                    // Get the id of which press release we want to show and call the
appropriate function.
                    var parts = url.split('/');
                    if (parts.length > 2) {
                        var action, pressReleaseId;
                        if (parts[2].indexOf('=') > -1) {
                            action = (parts[2].split('=')[1]).trim();
                            pressReleaseId = (parts[2].split('=')[0]).split('?')[0];
                        } else {
                            action = "";
                            pressReleaseId = parts[2].trim();
                        }

                        switch (action) {
                            case "": Cmgt.Prl.model.formData.mode = "view"; break;
                            case "edit": Cmgt.Prl.model.formData.mode = "edit";
break;
                            case "delete": Cmgt.Prl.model.formData.mode = "delete";
break;
                        };

                        if (action === "delete") {
                            //Delete Logic
                            console.log("This is the delete action");
                            Cmgt.Prl.controller.deletePressRelease(pressReleaseId);

                        } else {
                            //View and Edit logic
                            Cmgt.Prl.model.currentPressReleaseId =
parseInt(pressReleaseId);
                            Cmgt.Prl.model.currentPressRelease =
Cmgt.Prl.controller.getCurrentPressRelease(pressReleaseId);
                            //Initially set the form data to current press release
data to display the form
                            //Later check the difference w.r.t. the current(base)
data only
                            Cmgt.Prl.model.formData.pressRelease =
Cmgt.Prl.model.currentPressRelease;

Cmgt.Prl.view.renderPressReleaseFormPage(Cmgt.Prl.model.formData);
                            Cmgt.Prl.view.registerFormEvents(pressReleaseId,
Cmgt.Prl.model.formData.mode);
                            Cmgt.Prl.view.registerFormValidator();
                        }
```

```
                    } else if (parts.length === 2) {
                        if (parts[1].trim() === "PressRelease?action=create") {
                            //Create Logic
                            Cmgt.Prl.model.formData.mode = "create";
                            Cmgt.Prl.model.formData.pressRelease = {};

Cmgt.Prl.view.renderPressReleaseFormPage(Cmgt.Prl.model.formData);
                            Cmgt.Prl.view.registerFormEvents(pressReleaseId,
Cmgt.Prl.model.formData.mode);
                            Cmgt.Prl.view.registerFormValidator();
                        }

                    } else if (parts.length === 1) {
                        //View Latest 5 Press Release logic
                        Cmgt.Prl.model.getLatestPressReleases();
                    }
                },
            };

            // Execute the needed function depending on the url keyword (stored in
temp).
            if(map[temp]){
                map[temp]();
            }
            // If the keyword isn't listed in the above - render the error page.
            else {
                console.log("URL mapping not found");
                Cmgt.Prl.view.sendNotification("error", "Error!", "URL mapping not
found");
            }
        },

        getCurrentPressRelease: function(pressReleaseId) {
            var pressRelease;

Cmgt.Prl.model.pressReleaseData.pressReleases.forEach(function(elem,indx,arr){
                if (elem.pressReleaseId.toString() === pressReleaseId) {
                    pressRelease=elem;
                }
            });
            return pressRelease;
        }
    }
};

$(function() {
    // An event handler with calls the render function on every hashchange.
    // The render function will show the appropriate content of out page.
    $(window).on('hashchange', function(){
        Cmgt.Prl.controller.route(decodeURI(window.location.hash));
    });

    // Manually trigger a hashchange to start the app and display the Home page.
    window.location.hash = '#';
    $(window).trigger('hashchange');
```

```
})
```

The CSS

custom.css

```
* {
	font-family: "GE Inspira", "ge-inspira", "Helvetica Neue", Helvetica, Arial,
sans-serif;
}

.navbar-brand {
	padding-top: 8px;
}
.main-logo {
	height: auto;
	width: auto;
	margin-right: 5px;
	background: transparent;
}

.dropdown-header {
    font-weight: bold;
    font-size: 14px;
    background-color: #e6e6e6;
}

body { padding-top: 80px; }

.page-action {
    margin-right: 5px;
}

.table > thead > tr > th {
    text-align: center;
    vertical-align: middle;
}

.table > tbody > tr > td {
    text-align: center;
    vertical-align: middle;
}

.jumbotron p.fileInput {
    font-size: 14px;
}

.modal-header {
    background-color: #e6e6e6;
}

div.jGrowl-notification {
    font-size: 14px;
    float: right;
```

```
    margin-left: 6px;
    background-color: blue;
}
```

bower.json

```
{
  "name": "efsextcontent",
  "authors": [
    "Prabhu Purohit"
  ],
  "description": "This website is designed to post content to efs external website",
  "main": "index.html",
  "keywords": [
    "efs",
    "external",
    "website",
    "press",
    "release",
    "content",
    "posting"
  ],
  "license": "PCP Innovation",
  "homepage": "http://extcontent.pcp.innovation.com/",
  "private": true,
  "ignore": [
    "**/.*",
    "node_modules",
    "bower_components",
    "test",
    "tests"
  ],
  "dependencies": {
    "jquery": "2",
    "bootstrap": "3",
    "dustjs-linkedin": "^2.7.5",
    "dustjs-helpers": "^1.7.3",
    "blockUI": "blockui#*",
    "ckeditor": "^4.7.1",
    "jquery-form-validator": "^2.3.74"
  }
}
```

ckeditor/config.js

```
/**
 * @license Copyright (c) 2003-2017, CKSource - Frederico Knabben. All rights
reserved.
 * For licensing, see LICENSE.md or http://ckeditor.com/license
 */

CKEDITOR.editorConfig = function( config ) {
	// Define changes to default configuration here.
```

```
	// For complete reference see:
	// http://docs.ckeditor.com/#!/api/CKEDITOR.config

	// The toolbar groups arrangement, optimized for two toolbar rows.
	config.toolbarGroups = [
		{ name: 'clipboard',   groups: [ 'clipboard', 'undo' ] },
		{ name: 'editing',     groups: [ 'find', 'selection', 'spellchecker' ]
},
		{ name: 'links' },
		{ name: 'insert' },
		{ name: 'forms' },
		{ name: 'tools' },
		{ name: 'document',    groups: [ 'mode', 'document', 'doctools' ] },
		{ name: 'others' },
		'/',
		{ name: 'basicstyles', groups: [ 'basicstyles', 'cleanup' ] },
		{ name: 'paragraph',   groups: [ 'list', 'indent', 'blocks', 'align',
'bidi' ] },
		{ name: 'styles' },
		{ name: 'colors' },
		{ name: 'about' }
	];

	// Remove some buttons provided by the standard plugins, which are
	// not needed in the Standard(s) toolbar.

	//Prabhu Purohit: Keeping Source is a good option to check the generated HTML
	//in the development phase
	config.removeButtons =
'Subscript,Superscript,About,Anchor,SpecialChar,Blockquote,HorizontalRule';

	// Set the most common block elements.
	config.format_tags = 'p;h1;h2;h3;pre';

	// Simplify the dialog windows.
	config.removeDialogTabs = 'image:advanced;link:advanced';
};
```

The Rest Service

pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.ge.capital.efs</groupId>
  <artifactId>efscontmgmt</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>
  <properties>
	<spring.version>4.0.6.RELEASE</spring.version>
	<jackson.version>2.5.0</jackson.version>
```

```
    <log4j.version>1.2.15</log4j.version>
    <java.compiler.version>1.7</java.compiler.version>
</properties>
<build>
    <finalName>efscontmgmt</finalName>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>3.2</version>
            <configuration>
                <source>${java.compiler.version}</source>
                <target>${java.compiler.version}</target>
            </configuration>
        </plugin>
    </plugins>
</build>
<dependencies>
    <!-- Spring -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-jdbc</artifactId>
        <version>${spring.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-tx</artifactId>
        <version>${spring.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-aop</artifactId>
        <version>${spring.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-core</artifactId>
        <version>${spring.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-beans</artifactId>
        <version>${spring.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>${spring.version}</version>
    </dependency>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context-support</artifactId>
        <version>${spring.version}</version>
    </dependency>
    <dependency>
```

```
        <groupId>org.springframework</groupId>
        <artifactId>spring-web</artifactId>
        <version>${spring.version}</version>
</dependency>
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-webmvc</artifactId>
        <version>${spring.version}</version>
</dependency>
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-test</artifactId>
        <version>${spring.version}</version>
</dependency>
<dependency>
        <groupId>cglib</groupId>
        <artifactId>cglib</artifactId>
        <version>2.2.2</version>
</dependency>

<!-- MyBatis configuration -->
<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis</artifactId>
    <version>3.0.6</version>
</dependency>
<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis-spring</artifactId>
    <version>1.0.1</version>
</dependency>

<!-- MySQL configuration -->
<dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>5.1.6</version>
</dependency>

<dependency>
        <groupId>c3p0</groupId>
        <artifactId>c3p0</artifactId>
        <version>0.9.1.2</version>
</dependency>

<!-- javax.servlet dependency -->
<dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>javax.servlet-api</artifactId>
        <version>3.0.1</version>
        <scope>provided</scope>
</dependency>
<dependency>
        <groupId>javax.servlet.jsp</groupId>
        <artifactId>javax.servlet.jsp-api</artifactId>
```

```
            <version>2.2.1</version>
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>javax.el</groupId>
            <artifactId>javax.el-api</artifactId>
            <version>2.2.2</version>
            <scope>provided</scope>
        </dependency>

        <!-- Jackson Dependency -->

        <dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-core</artifactId>
        <version>${jackson.version}</version>
        </dependency>

        <dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-databind</artifactId>
        <version>${jackson.version}</version>
        </dependency>

        <!-- Multi-part file support -->
        <dependency>
          <groupId>commons-fileupload</groupId>
          <artifactId>commons-fileupload</artifactId>
          <version>1.3.1</version>
     </dependency>

     <!-- https://mvnrepository.com/artifact/jcifs/jcifs -->
        <dependency>
            <groupId>jcifs</groupId>
            <artifactId>jcifs</artifactId>
            <version>1.3.17</version>
        </dependency>

     <!-- Log4j -->
        <dependency>
            <groupId>log4j</groupId>
            <artifactId>log4j</artifactId>
            <version>${log4j.version}</version>
        </dependency>
   </dependencies>
</project>
```

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.0" xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd">
```

```
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>
              /WEB-INF/spring/application-context.xml
              /WEB-INF/spring/service-context.xml
             /WEB-INF/spring/persist-context.xml
        </param-value>
  </context-param>

  <listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-
class>
  </listener>
  <servlet>
    <servlet-name>EfsContMgmtDispatcher</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    <init-param>
      <param-name>contextConfigLocation</param-name>
      <param-value></param-value>
    </init-param>
    <load-on-startup>2</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>EfsContMgmtDispatcher</servlet-name>
    <!-- Putting /* will force every resource request to the DispatcherServlet
including static resources -->
    <url-pattern>/</url-pattern>
  </servlet-mapping>

  <resource-ref>
    <description>DB Connection for Application</description>
    <res-ref-name>jdbc/efsContMgmtDS</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
  </resource-ref>

  <!-- Filters -->
  <!-- This filter is an implementation of W3C's CORS (Cross-Origin Resource Sharing)
specification,
  which is a mechanism that enables cross-origin requests.
  Link: https://tomcat.apache.org/tomcat-7.0-doc/config/filter.html#CORS_Filter
  This is required only for testing local client to interact with the server. Remove
it
  from the actual production source -->

  <!-- <filter>
       <filter-name>CorsFilter</filter-name>
       <filter-class>org.apache.catalina.filters.CorsFilter</filter-class>
  </filter>
  <filter-mapping>
       <filter-name>CorsFilter</filter-name>
       <url-pattern>/*</url-pattern>
  </filter-mapping> -->

  <!-- Exception Handling -->
```

```
  <!-- The following will be going through DispatcherServlet and
SimpleUrlHandlerMapping -->
  <error-page>
    <exception-type>java.lang.Throwable</exception-type>
    <location>/view/errorPage.html</location>
  </error-page>
  <error-page>
    <error-code>403</error-code>
    <location>/view/errorPage.html</location>
  </error-page>
  <error-page>
    <error-code>404</error-code>
    <location>/view/errorPage.html</location>
  </error-page>
  <error-page>
    <error-code>500</error-code>
    <location>/view/errorPage.html</location>
  </error-page>
  <error-page>
    <error-code>503</error-code>
    <location>/view/errorPage.html</location>
  </error-page>
</web-app>
```

application-context.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
       xmlns:util="http://www.springframework.org/schema/util"
xmlns:mvc="http://www.springframework.org/schema/mvc"
       xsi:schemaLocation="http://www.springframework.org/schema/mvc
http://www.springframework.org/schema/mvc/spring-mvc-4.0.xsd
            http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
            http://www.springframework.org/schema/util
http://www.springframework.org/schema/util/spring-util-4.0.xsd
            http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.0.xsd">

       <!-- Enables the Spring MVC @Controller programming model -->
       <mvc:annotation-driven />

       <!-- Add caching after the project is deployed in Production -->
       <!-- <mvc:interceptors>
              <mvc:interceptor>
                     <mvc:mapping path="/resources/**/*.js" />
                     <mvc:mapping path="/resources/**/*.css" />
                     <mvc:mapping path="/resources/**/*.png" />
                     <mvc:mapping path="/resources/**/*.gif" />
                     <mvc:mapping path="/resources/**/*.jpg" />
                     <bean id="webContentInterceptor"

       class="org.springframework.web.servlet.mvc.WebContentInterceptor">
```

```
                        <property name="cacheSeconds" value="28800" />
                        <property name="useExpiresHeader" value="true" />
                        <property name="useCacheControlHeader" value="true" />
                        <property name="useCacheControlNoStore" value="false" />
                    </bean>
            </mvc:interceptor>
        </mvc:interceptors> -->

        <!-- @Controller stereotypes will be detected. -->
        <context:component-scan
              base-package="com.ge.efs.contmgmt.controller,
              com.ge.efs.contmgmt.dao,
              com.ge.efs.contmgmt.exception">
        </context:component-scan>

        <!-- Static resources mapping -->
        <!-- Goes through SimpleUrlHandlerMapping -->
        <mvc:resources mapping="/resources/**" location="/resources/" />
        <mvc:resources mapping="/view/**" location="/WEB-INF/view/" />

        <!-- Resolves view selected for rendering by @Controllers to .jsp resources
              in the /WEB-INF/view directory -->
        <!-- <bean

        class="org.springframework.web.servlet.view.InternalResourceViewResolver">
              <property name="prefix" value="/WEB-INF/view/" />
              <property name="suffix" value=".jsp" />
              <property name="order" value="1" />
        </bean> -->

        <!-- Multiple property placeholder not working, so clubbed into one -->
        <bean
class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">
            <!-- <property name="ignoreResourceNotFound" value="true"/> -->
            <property name="locations">
                <list>
                     <value>classpath:mybatis/dataSource.properties</value>
                    <value>classpath:netdrive/networkDrive.properties</value>
                </list>
            </property>
        </bean>

</beans>
```

persist-context.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:jdbc="http://www.springframework.org/schema/jdbc"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xmlns:jee="http://www.springframework.org/schema/jee"
```

```
      xsi:schemaLocation="http://www.springframework.org/schema/jdbc
http://www.springframework.org/schema/jdbc/spring-jdbc.xsd
            http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
            http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-4.0.xsd
            http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-3.0.xsd
            http://www.springframework.org/schema/jee
http://www.springframework.org/schema/jee/spring-jee-4.0.xsd">

    <context:component-scan base-package="com.ge.contmgmt.util" />

      <!-- Transaction Manager -->
      <bean id="transactionManager"
class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
            <property name="dataSource" ref="dataSource" />
      </bean>

      <tx:annotation-driven />

      <!-- MapperScannerConfigurer -->
      <bean id="mapperScannerConfigurer"
class="org.mybatis.spring.mapper.MapperScannerConfigurer">
            <property name="basePackage" value="com.ge.efs.contmgmt.mappers" />
            <property name="sqlSessionFactory" ref="sqlSessionFactory" />
            <!-- <property name="sqlSessionFactoryBeanName"
value="sqlSessionFactory1"/> -->
      </bean>

      <!-- SessionFactory -->
      <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
            <property name="configLocation" value="classpath:mybatis/mybatis-
config.xml" />
            <property name="dataSource" ref="dataSource" />
      </bean>

      <!-- DataSource -->
      <beans profile="local">
            <bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"
                  destroy-method="close">
                  <property name="user" value="${jdbc.username}" />
                  <property name="password" value="${jdbc.password}" />
                  <property name="driverClass" value="${jdbc.driver}" />
                  <property name="jdbcUrl" value="${jdbc.url}" />
                  <property name="maxIdleTime" value="1800" />
                  <property name="maxIdleTimeExcessConnections" value="300" />
                  <property name="initialPoolSize" value="1" />
                  <property name="maxPoolSize" value="2" />
                  <property name="minPoolSize" value="1" />
                  <property name="acquireIncrement" value="1" />
                  <property name="acquireRetryAttempts" value="0" />
            </bean>
      </beans>
```

```
	<beans profile="efs_acceptance,efs_union,efs_production">
		<jee:jndi-lookup id="dataSource" jndi-
name="java:/comp/env/jdbc/efsContMgmtDS" />
	</beans>

</beans>
```

service-context.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
	xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
		http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.0.xsd">


	<!-- Configuration for Multipart file resolver -->
	<bean id="multipartResolver"
	  class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
	  <!-- 25MB*1024*1024 -->
	  <property name="maxUploadSize" value="26214400"/>
    </bean>

    <beans profile="local,efs_acceptance">

		<bean id="jcifsUpload"
class="com.ge.efs.contmgmt.service.JCIFSUploadServiceImpl">
			<property name="uploadUser" value="${upload.user.stage}" />
			<property name="uploadPassword" value="${upload.password.stage}"
/>
			<property name="uploadPathImg" value="${jcifs.path.image.stage}"
/>
			<property name="uploadPathPdf" value="${jcifs.path.pdf.stage}" />
		</bean>

		<bean id="commonUpload"
class="com.ge.efs.contmgmt.service.CommonUploadServiceImpl">
			<property name="uploadPathImg" value="${common.path.image.stage}"
/>
			<property name="uploadPathPdf" value="${common.path.pdf.stage}"
/>
		</bean>
	</beans>

	<beans profile="efs_production">

		<bean id="jcifsUpload"
class="com.ge.efs.contmgmt.service.JCIFSUploadServiceImpl">
			<property name="uploadUser" value="${upload.user.prod}" />
			<property name="uploadPassword" value="${upload.password.prod}"
/>
```

```
                    <property name="uploadPathImg" value="${jcifs.path.image.prod}"
/>
                    <property name="uploadPathPdf" value="${jcifs.path.pdf.prod}" />
            </bean>

            <bean id="commonUpload"
class="com.ge.efs.contmgmt.service.CommonUploadServiceImpl">
                    <property name="uploadPathImg" value="${common.path.image.prod}"
/>
                    <property name="uploadPathPdf" value="${common.path.pdf.prod}" />
            </bean>
      </beans>

</beans>
```

context.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<Context path="/efscontmgmt">
    <!-- JNDI contexts used by the application -->
    <ResourceLink global="jdbc/efsContMgmtDS" name="efsContMgmtDS"
type="javax.sql.DataSource"/>
</Context>
```

MANIFEST.MF

```
Manifest-Version: 1.0
```

log4j.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE log4j:configuration PUBLIC "-//APACHE//DTD LOG4J 1.2//EN" "log4j.dtd">
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">

    <!-- Standard Console Appender -->
    <appender name="CONSOLE" class="org.apache.log4j.ConsoleAppender">
        <param name="Target" value="System.out" />
        <layout class="org.apache.log4j.PatternLayout">
            <param name="ConversionPattern" value="%d{dd MMM yyyy HH:mm:ss,SSS} %-5p:
%c - %m%n" />
        </layout>
    </appender>

    <!-- Application Loggers -->
    <logger name="com.ge.services">
        <level value="debug" />
    </logger>

    <logger name="org.springframework">
        <level value="debug" />
    </logger>

      <logger name="com.fasterxml.jackson">
        <level value="debug" />
```

```
    </logger>

    <!-- Root Logger -->
    <root>
        <priority value="debug" />
        <appender-ref ref="CONSOLE" />
    </root>

</log4j:configuration>
```

networkDrive.properties

```
#Stage Configuration
upload.user.stage=stageUser
upload.password.stage=XXXXXX
jcifs.path.image.stage=smb://share.pcp.innovation.com/Net_Drive_Stage$/root/img/
jcifs.path.pdf.stage=smb://share.pcp.innovation.com/Net_Drive_Stage$/root/file/
common.path.image.stage=\\\\share.pcp.innovation.com\\Net_Drive_Stage$\\root\\img\\
common.path.pdf.stage=\\\\share.pcp.innovation.com\\Net_Drive_Stage$\\root\\file\\

#Production Configuration
upload.user.prod=prodUser
upload.password.prod=XXXXXX
jcifs.path.image.prod= smb://share.pcp.innovation.com/Net_Drive_Prod$/root/img/
jcifs.path.pdf.prod= smb://share.pcp.innovation.com/Net_Drive_Prod$/root/file/
common.path.image.prod=\\\\share.pcp.innovation.com\\Net_Drive_Prod$\\root\\img\\
common.path.pdf.prod=\\\\share.pcp.innovation.com\\Net_Drive_Prod$\\root\\file\\
```

PressReleaseMapper.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd" >
<mapper namespace="com.ge.efs.contmgmt.mappers.PressReleaseMapper">

        <resultMap id="PressReleaseResultMap" type="PressRelease" >
            <result column="ID" property="pressReleaseId" jdbcType="NUMERIC" />
            <result column="INDUSTRY_FOCUS_ID" property="industryFocusId"
jdbcType="NUMERIC" />
            <result column="NAME" property="name" jdbcType="VARCHAR" />
            <result column="LOCATION" property="location" jdbcType="VARCHAR" />
            <result column="PUBLISH_DATE" property="publishDate" jdbcType="DATE" />
            <result column="BODY" property="body" jdbcType="VARCHAR" />
            <result column="CONTACT_INFO" property="contactInfo" jdbcType="VARCHAR"
/>
            <result column="QUOTE" property="quote" jdbcType="VARCHAR" />
            <result column="QUOTE_AUTHOR" property="quoteAuthor" jdbcType="VARCHAR"
/>
            <result column="QUOTE_AUTHOR_TITLE" property="quoteAuthorTitle"
jdbcType="VARCHAR" />
            <result column="IMAGE_URL" property="imageUrl" jdbcType="VARCHAR" />
            <result column="PDF_URL" property="pdfUrl" jdbcType="VARCHAR" />
            <result column="ACTIVE" property="active" jdbcType="NUMERIC" />
```

```
          <result column="INDUSTRY_FOCI_NAME" property="industryFocus"
jdbcType="VARCHAR" />
        </resultMap>

        <resultMap id="IndustryFocusResultMap" type="IndustryFocus" >
            <result column="ID" property="id" jdbcType="NUMERIC" />
            <result column="NAME" property="name" jdbcType="VARCHAR" />
        </resultMap>

      <select id="getLatestPressReleases" parameterType="Integer"
resultMap="PressReleaseResultMap" >
            SELECT
                  A.ID,
                  A.INDUSTRY_FOCUS_ID,
                  A.NAME,
                  A.LOCATION,
                  A.PUBLISH_DATE,
                  A.BODY,
                  A.CONTACT_INFO,
                  A.QUOTE,
                  A.QUOTE_AUTHOR,
                  A.QUOTE_AUTHOR_TITLE,
                  A.IMAGE_URL,
                  A.PDF_URL,
                  A.ACTIVE,
                  B.NAME AS 'INDUSTRY_FOCI_NAME'
            FROM PCP1000.PRESS_RELEASES A
            LEFT JOIN PCP1000.INDUSTRY_FOCI B
                  ON A.INDUSTRY_FOCUS_ID = B.ID
            ORDER BY A.ID DESC
            LIMIT #{count};
      </select>

      <select id="getPressReleaseById" parameterType="Integer"
resultMap="PressReleaseResultMap" >
            SELECT
                  A.ID,
                  A.INDUSTRY_FOCUS_ID,
                  A.NAME,
                  A.LOCATION,
                  A.PUBLISH_DATE,
                  A.BODY,
                  A.CONTACT_INFO,
                  A.QUOTE,
                  A.QUOTE_AUTHOR,
                  A.QUOTE_AUTHOR_TITLE,
                  A.IMAGE_URL,
                  A.PDF_URL,
                  A.ACTIVE,
                  B.NAME AS 'INDUSTRY_FOCI_NAME'
            FROM PCP1000.PRESS_RELEASES A
            LEFT JOIN PCP1000.INDUSTRY_FOCI B
                  ON A.INDUSTRY_FOCUS_ID = B.ID
            WHERE A.ID = #{pressReleaseId, jdbcType=NUMERIC}
      </select>
```

```
<select id="getAllIndustryFoci" resultMap="IndustryFocusResultMap" >
	SELECT
		ID, NAME
	FROM PCP1000.INDUSTRY_FOCI
</select>

<select id="getCurrentPressReleaseId" resultType="Integer" >
	SELECT ID
	  FROM PCP1000.PRESS_RELEASES A
	ORDER BY A.ID DESC
	LIMIT 1;
</select>

<!-- foreach construct can also be used for the column list and value list -->
<insert id="createPressRelease" parameterType="PressRelease">
	INSERT INTO
		PCP1000.PRESS_RELEASES

		(ID,
		<if test="industryFocusId != null">
		INDUSTRY_FOCUS_ID,
		</if>
		<if test="name != null">
		NAME,
		</if>
		<if test="location != null">
		LOCATION,
		</if>
		<if test="publishDate != null">
		PUBLISH_DATE,
		</if>
		<if test="body != null">
		BODY,
		</if>
		<if test="contactInfo != null">
		CONTACT_INFO,
		</if>
		<if test="quote != null">
		QUOTE,
		</if>
		<if test="quoteAuthor != null">
		QUOTE_AUTHOR,
		</if>
		<if test="quoteAuthorTitle != null">
		QUOTE_AUTHOR_TITLE,
		</if>
		<if test="imageUrl != null">
		IMAGE_URL,
		</if>
		<if test="pdfUrl != null">
		PDF_URL,
		</if>
		ACTIVE
		)
```

```
                VALUES(
                       #{pressReleaseId,jdbcType=NUMERIC},
                <if test="industryFocusId != null">
                       #{industryFocusId,jdbcType=NUMERIC},
                </if>
                <if test="name != null">
                       #{name,jdbcType=VARCHAR},
                </if>
                <if test="location != null">
                       #{location,jdbcType=VARCHAR},
                </if>
                <if test="publishDate != null">
                       #{publishDate,jdbcType=DATE},
                </if>
                <if test="body != null">
                       #{body,jdbcType=VARCHAR},
                </if>
                <if test="contactInfo != null">
                       #{contactInfo,jdbcType=VARCHAR},
                </if>
                <if test="quote != null">
                       #{quote,jdbcType=VARCHAR},
                </if>
                <if test="quoteAuthor != null">
                       #{quoteAuthor,jdbcType=VARCHAR},
                </if>
                <if test="quoteAuthorTitle != null">
                       #{quoteAuthorTitle,jdbcType=VARCHAR},
                </if>
                <if test="imageUrl != null">
                       #{imageUrl,jdbcType=VARCHAR},
                </if>
                <if test="pdfUrl != null">
                       #{pdfUrl,jdbcType=VARCHAR},
                </if>
                       #{active,jdbcType=NUMERIC}
                )
</insert>

<update id="editPressRelease" parameterType="PressRelease">
       UPDATE PCP1000.PRESS_RELEASES
         <set>
                <if test="industryFocusId != null">
                       INDUSTRY_FOCUS_ID = #{industryFocusId,jdbcType=NUMERIC},
                </if>
                <if test="name != null">
                       NAME = #{name,jdbcType=VARCHAR},
                </if>
                <if test="location != null">
                       LOCATION = #{location,jdbcType=VARCHAR},
                </if>
                <if test="publishDate != null">
                       PUBLISH_DATE = #{publishDate,jdbcType=DATE},
                </if>
```

```
                    <if test="body != null">
                          BODY = #{body,jdbcType=VARCHAR},
                    </if>
                    <if test="contactInfo != null">
                          CONTACT_INFO = #{contactInfo,jdbcType=VARCHAR},
                    </if>
                    <if test="quote != null">
                          QUOTE = #{quote,jdbcType=VARCHAR},
                    </if>
                    <if test="quoteAuthor != null">
                          QUOTE_AUTHOR = #{quoteAuthor,jdbcType=VARCHAR},
                    </if>
                    <if test="quoteAuthorTitle != null">
                          QUOTE_AUTHOR_TITLE = #{quoteAuthorTitle,jdbcType=VARCHAR},
                    </if>
                    <if test="imageUrl != null">
                          IMAGE_URL = #{imageUrl,jdbcType=VARCHAR},
                    </if>
                    <if test="pdfUrl != null">
                          PDF_URL = #{pdfUrl,jdbcType=VARCHAR},
                    </if>
                    <if test="active != null">
                          ACTIVE = #{active,jdbcType=NUMERIC}
                    </if>
               </set>
                    WHERE
                          ID = #{pressReleaseId,jdbcType=NUMERIC};
      </update>

      <delete id="deletePressRelease" parameterType="Integer">
            DELETE
              FROM PCP1000.PRESS_RELEASES
            WHERE
              ID = #{pressReleaseId,jdbcType=NUMERIC}
      </delete>
</mapper>
```

mybatis-config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration
    PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
    "http://mybatis.org/dtd/mybatis-3-config.dtd">

<configuration>

  <settings>
    <setting name="lazyLoadingEnabled" value="true" />
    <setting name="aggressiveLazyLoading" value="false" />
    <setting name="mapUnderscoreToCamelCase" value="true" />
  </settings>

  <typeAliases>
    <!-- Type aliases to be used in the Mapper.xml -->
    <typeAlias type="com.ge.efs.contmgmt.model.PressRelease" alias="PressRelease" />
```

```
    <typeAlias type="com.ge.efs.contmgmt.model.IndustryFocus" alias="IndustryFocus"
/>
  </typeAliases>

  <!-- If a single TypeHandler is registered to handle a Java type,
  it will be used by default in ResultMaps using this Java type -->
  <!-- You can create a custom TypeHandler and may not choose to register it in the
mybatis-config.xml
  and optionally use it in the ResultMaps using typeHandler=CustomHandler.class -->
  <typeHandlers>
    <typeHandler handler="com.ge.efs.contmgmt.typehandler.BooleanTypeHandler"
                    javaType="java.lang.Boolean" jdbcType="VARCHAR" />
  </typeHandlers>

  <mappers>
      <mapper resource="mybatis/PressReleaseMapper.xml" />
  </mappers>

</configuration>
```

dataSource.properties

```
jdbc.driver=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3360
jdbc.username=root
jdbc.password=rootPassword
```

Java

PressReleaseController.java

```
package com.ge.efs.contmgmt.controller;

import java.io.IOException;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import javax.servlet.http.HttpServletRequest;

import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.util.CollectionUtils;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.multipart.MultipartFile;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;
```

```
import com.ge.efs.contmgmt.dao.PressReleaseDao;
import com.ge.efs.contmgmt.exception.AppException;
import com.ge.efs.contmgmt.exception.DBException;
import com.ge.efs.contmgmt.exception.ExceptionConstants;
import com.ge.efs.contmgmt.exception.RestExceptionHandler;
import com.ge.efs.contmgmt.model.IndustryFocus;
import com.ge.efs.contmgmt.model.PressRelease;
import com.ge.efs.contmgmt.model.RestResponse;
import com.ge.efs.contmgmt.service.FileUploadService;

//CrossOrigin filter only after Spring 4.2
//@CrossOrigin(maxAge = 3600)
@Controller
@RequestMapping("/")
public class PressReleaseController {

	public static final Logger LOGGER =
Logger.getLogger(PressReleaseController.class);

	@Autowired
	@Qualifier("commonUpload")
	FileUploadService fileUploadService;

	@Autowired
	PressReleaseDao pressReleaseDao;

	@Autowired
	RestExceptionHandler restExceptionHandler;

	@RequestMapping(method = RequestMethod.GET)
	public String dislayHome(Model model, HttpServletRequest request){

		return "/view/index.html";

	}

	@RequestMapping(value = "/pressreleases/pressrelease/{pressReleaseId}", method
= RequestMethod.GET)
	public @ResponseBody RestResponse getPressReleasebyId(Model model,
			@PathVariable("pressReleaseId") Integer pressReleaseId,
HttpServletRequest request)
			throws DBException, AppException {
		try {
			RestResponse restResp = new RestResponse();
			PressRelease pressRelease =
pressReleaseDao.getOnePressRelease(pressReleaseId);
			if (null == pressRelease) {
				restResp.setMessage(ExceptionConstants.NO_RESULT);
				restResp.setSeverity(ExceptionConstants.MEDIUM);

	restResp.setMsgDesc(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.NO_R
ESULT));
			}
			restResp.setPayload(pressRelease);
			return restResp;
```

```
		}
		catch (DBException e) {
			throw new DBException("PRESS RELEASE " +
ExceptionConstants.NOT_FOUND, e);
		}
		catch (Exception e) {
			throw new AppException("Exception occured in Application", e);
		}
	}

	@RequestMapping(value = "/pressreleases/latest5", method = RequestMethod.GET)
	public @ResponseBody RestResponse getLatest5PressReleases(Model model,
HttpServletRequest request)
			throws DBException, AppException {
		try {
			RestResponse restResp = new RestResponse();
			List<PressRelease> pressRelease =
pressReleaseDao.getMultiplePressReleases(5);
			if (CollectionUtils.isEmpty(pressRelease)) {
				restResp.setMessage(ExceptionConstants.NO_RESULT);
				restResp.setSeverity(ExceptionConstants.MEDIUM);

	restResp.setMsgDesc(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.NO_R
ESULT));
			}
			restResp.setPayload(pressRelease);
			return restResp;
		}
		catch (DBException e) {
			throw new DBException("PRESS RELEASE " +
ExceptionConstants.NOT_FOUND, e);
		}
		catch (Exception e) {
			throw new AppException("Exception occured in Application", e);
		}
	}

	@RequestMapping(value = "/industryfoci/all", method = RequestMethod.GET)
	public @ResponseBody RestResponse getAllIndustryFoci(Model model,
HttpServletRequest request)
			throws DBException, AppException {
		try {
			RestResponse restResp = new RestResponse();
			List<IndustryFocus> industryFoci =
pressReleaseDao.getIndustryFoci();
			if (CollectionUtils.isEmpty(industryFoci)) {
				restResp.setMessage(ExceptionConstants.NO_RESULT);
				restResp.setSeverity(ExceptionConstants.MEDIUM);

	restResp.setMsgDesc(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.NO_R
ESULT));
			}
			restResp.setPayload(industryFoci);
			return restResp;
		}
```

```
		catch (DBException e) {
			throw new DBException("INDUSTRY FOCI " +
ExceptionConstants.NOT_FOUND, e);
		}
		catch (Exception e) {
			throw new AppException("Exception occured in Application", e);
		}
	}

	@RequestMapping(value = "/pressreleases/pressrelease", method =
RequestMethod.POST)
	public @ResponseBody RestResponse createPressRelease(@RequestBody PressRelease
pressRelease,
			@RequestParam(value="action") String action,
			HttpServletRequest request,
			RedirectAttributes redirectAttributes) throws DBException,
AppException {

		try {
			RestResponse restResp = new RestResponse();
			Integer insertCount = 0;
			Integer newPressReleaseId = 0;
			if (action.equals("create")) {
				Integer currentPressReleaseId =
pressReleaseDao.getCurrPressReleaseId();
				if (null != currentPressReleaseId) {
					newPressReleaseId = currentPressReleaseId + 1;
					pressRelease.setPressReleaseId(newPressReleaseId);
					insertCount =
pressReleaseDao.insPressRelease(pressRelease);
					if (insertCount == 1) {
						LOGGER.info("Press Release Inserted
Successfully");
					} else {

	restResp.setMessage(ExceptionConstants.TXN_FAILED);

	restResp.setSeverity(ExceptionConstants.MEDIUM);

	restResp.setMsgDesc(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.TXN_
FAILED));
					}
				}
				else {
					restResp.setMessage(ExceptionConstants.NO_RESULT);
					restResp.setSeverity(ExceptionConstants.MEDIUM);

	restResp.setMsgDesc(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.NO_R
ESULT));
				}
			}
			restResp.setPayload(newPressReleaseId);
			return restResp;

		} catch (DBException e) {
```

```
                  throw new DBException("PRESS RELEASE " +
ExceptionConstants.INSERT_FAILED, e);
            }

            catch (Exception e) {
                  throw new AppException("Exception occured in Application", e);
            }
      }

      @RequestMapping(value = "/pressreleases/pressrelease/{pressReleaseId}", method
= RequestMethod.POST)
      public @ResponseBody RestResponse editPressRelease(@RequestBody PressRelease
pressRelease,
                  @RequestParam(value = "action") String action,
                  @PathVariable("pressReleaseId") Integer pressReleaseId)
                              throws DBException, AppException {

            try {

                  RestResponse restResp = new RestResponse();
                  if (action.equals("delete")) {
                        Integer deleteCount = 0;
                        deleteCount =
pressReleaseDao.delPressRelease(pressReleaseId);
                        if (deleteCount == 1) {
                              LOGGER.info("Press Release deleted Successfully");
                        } else {
                              restResp.setMessage(ExceptionConstants.TXN_FAILED);
                              restResp.setSeverity(ExceptionConstants.MEDIUM);

      restResp.setMsgDesc(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.TXN_
FAILED));
                        }

                        restResp.setPayload(deleteCount);
                        return restResp;

                  } else {

                        Integer editCount = 0;
                        editCount = pressReleaseDao.updPressRelease(pressRelease);

                        if (editCount == 1) {
                              LOGGER.info("Press Release updated Successfully");
                        } else {
                              restResp.setMessage(ExceptionConstants.TXN_FAILED);
                              restResp.setSeverity(ExceptionConstants.MEDIUM);

      restResp.setMsgDesc(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.TXN_
FAILED));
                        }

                        restResp.setPayload(editCount);
                        return restResp;
                  }
```

```
            } catch (DBException e) {
                  if (action.equals("delete")) {
                        throw new DBException("PRESS RELEASE " +
ExceptionConstants.DELETE_FAILED, e);
                  } else {
                        throw new DBException("PRESS RELEASE " +
ExceptionConstants.UPDATE_FAILED, e);
                  }
            }
            catch (Exception e) {
                  throw new AppException("Exception occured in Application", e);
            }
      }

      @RequestMapping(value = "/upload", method = RequestMethod.POST)
    public @ResponseBody RestResponse uploadMultiFiles(@RequestParam("files")
MultipartFile[] files)
            throws IOException, AppException {

            RestResponse restResp = new RestResponse();
            Map<String, String> resp = new HashMap<String, String>();
            int fileSentCnt = files.length;
            int imgUploadCnt = 0;
            int pdfUploadCnt = 0;

            try {

                  if (fileSentCnt == 0) {
                        throw new
IOException(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.READ_ERROR));
                  }

                  for (MultipartFile file : files) {

                        String filename = file.getOriginalFilename();

                        if (filename.length() < 5) {
                              throw new
IOException(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.INVALID_FILE));
                        }

                        String filetype = filename.substring(filename.length()-3);
                        if (!filetype.equalsIgnoreCase("jpg") &&
!filetype.equalsIgnoreCase("png") &&
                                    !filetype.equalsIgnoreCase("gif") &&
!filetype.equalsIgnoreCase("pdf")) {
                              throw new
IOException(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.INVALID_FILE));
                        }

                        if (file.isEmpty()) {
                              throw new
IOException(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.NO_INPUT));
                         }
```

```
                        if (filetype.equalsIgnoreCase("jpg") ||
                                    filetype.equalsIgnoreCase("png") ||
                                    filetype.equalsIgnoreCase("gif")) {
                              imgUploadCnt = fileUploadService.uploadFile(file,
"IMAGE");
                              if (imgUploadCnt == 1) {
                                    LOGGER.info("Image File Upload Successful");
                                    resp.put(filename,
ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.SUCCESSFUL));
                              }
                        }

                        if (filetype.equalsIgnoreCase("pdf")) {
                              pdfUploadCnt = fileUploadService.uploadFile(file,
"PDF");
                              if (pdfUploadCnt == 1) {
                                    LOGGER.info("PDF File Upload Successful");
                                    resp.put(filename,
ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.SUCCESSFUL));
                              }
                        }

                }

                   restResp.setMessage(ExceptionConstants.SUCCESSFUL);
                   restResp.setSeverity(ExceptionConstants.LOW);

      restResp.setMsgDesc(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.SUCC
ESSFUL));
                   restResp.setPayload(resp);

                   return restResp;

        }

             catch (IOException e) {
                   throw new IOException(e);
        }

        catch (Exception e) {
                   throw new AppException("Exception occured in Application", e);
             }

    }
}
```

PressReleaseDao.java

```
package com.ge.efs.contmgmt.dao;

import java.util.List;

import com.ge.efs.contmgmt.exception.DBException;
import com.ge.efs.contmgmt.model.IndustryFocus;
```

```
import com.ge.efs.contmgmt.model.PressRelease;

public interface PressReleaseDao {

	public PressRelease getOnePressRelease(Integer pressReleaseId) throws
DBException;
	public List<PressRelease> getMultiplePressReleases(Integer count) throws
DBException;
	public List<IndustryFocus> getIndustryFoci() throws DBException;
	public Integer insPressRelease(PressRelease pressRelease) throws DBException;
	public Integer delPressRelease(Integer pressReleaseId) throws DBException;
	public Integer updPressRelease(PressRelease pressRelease) throws DBException;
	public Integer getCurrPressReleaseId() throws DBException;

}
```

PressReleaseDaoImpl.java

```
package com.ge.efs.contmgmt.dao;

import java.util.List;

import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;
import org.springframework.transaction.annotation.Transactional;

import com.ge.efs.contmgmt.exception.DBException;
import com.ge.efs.contmgmt.mappers.PressReleaseMapper;
import com.ge.efs.contmgmt.model.IndustryFocus;
import com.ge.efs.contmgmt.model.PressRelease;

@Repository
public class PressReleaseDaoImpl implements PressReleaseDao {

	public static final Logger LOGGER =
Logger.getLogger(PressReleaseDaoImpl.class);

	@Autowired
	PressReleaseMapper pressReleaseMapper;

	@Override
	public PressRelease getOnePressRelease(Integer pressReleaseId) throws
DBException {

		try {
			PressRelease pressRelease =
pressReleaseMapper.getPressReleaseById(pressReleaseId);
			return pressRelease;
		}
		catch (Exception e) {
			throw new DBException(e);
		}
	}
```

```
	@Override
	public List<PressRelease> getMultiplePressReleases(Integer count) throws
DBException {
		try {
			List<PressRelease> pressRelease =
pressReleaseMapper.getLatestPressReleases(5);
			return pressRelease;
		}
		catch (Exception e) {
			throw new DBException(e);
		}
	}

	@Override
	public List<IndustryFocus> getIndustryFoci() throws DBException {
		try {
			List<IndustryFocus> industryFoci =
pressReleaseMapper.getAllIndustryFoci();
			return industryFoci;
		}
		catch (Exception e) {
			throw new DBException(e);
		}
	}

	@Override
	@Transactional
	public Integer insPressRelease(PressRelease pressRelease) throws DBException {
		try {
			Integer insertCount = 0;
			insertCount =
pressReleaseMapper.createPressRelease(pressRelease);
			return insertCount;
		} catch (Exception e) {
			throw new DBException(e);
		}
	}

	@Override
	@Transactional
	public Integer delPressRelease(Integer pressReleaseId) throws DBException {
		try {
			Integer deleteCount = 0;
			deleteCount =
pressReleaseMapper.deletePressRelease(pressReleaseId);
			return deleteCount;
		} catch (Exception e) {
			throw new DBException(e);
		}
	}

	@Override
	@Transactional
	public Integer updPressRelease(PressRelease pressRelease) throws DBException {
		try {
```

```
                        Integer editCount = 0;
                        editCount = pressReleaseMapper.editPressRelease(pressRelease);
                        return editCount;
                } catch (Exception e) {
                        throw new DBException(e);
                }
        }

        @Override
        public Integer getCurrPressReleaseId() throws DBException {
                try {
                        Integer currPressReleaseId =
pressReleaseMapper.getCurrentPressReleaseId();
                        return currPressReleaseId;
                } catch (Exception e) {
                        throw new DBException(e);
                }
        };

}
```

AppException.java

```
package com.ge.efs.contmgmt.exception;

/**
 * @author Prabhu Purohit
 *
 */
public class AppException extends Exception {

        private static final long serialVersionUID = 1L;

        /**
         * Constructor of DBException class.
         */
        public AppException() {
                super();
        }

        /**
         *
         * @param message
         * @param cause
         */
        public AppException(final String message, final Throwable cause) {
                super(message, cause);
        }

        /**
         *
         * @param message
         */
        public AppException(final String message) {
                super(message);
```

```
	}

	/**
	 *
	 * @param cause
	 */
	public AppException(final Throwable cause) {
		super(cause);

	}

}
```

DBException.java

```
package com.ge.efs.contmgmt.exception;

/**
 * @author Prabhu Purohit
 *
 */
public class DBException extends Exception {

	private static final long serialVersionUID = 1L;

	/**
	 * Constructor of DBException class.
	 */
	public DBException() {
		super();
	}

	/**
	 *
	 * @param message
	 * @param cause
	 */
	public DBException(final String message, final Throwable cause) {
		super(message, cause);
	}

	/**
	 *
	 * @param message
	 */
	public DBException(final String message) {
		super(message);
	}

	/**
	 *
	 * @param cause
	 */
	public DBException(final Throwable cause) {
		super(cause);
```

```
	}

}
```

ExceptionConstants.java

```
package com.ge.efs.contmgmt.exception;

import java.util.HashMap;
import java.util.Map;

public class ExceptionConstants {

	//message codes

	public static final String SUCCESSFUL = "0000";
	public static final String NO_RESULT = "1001";
	public static final String TXN_FAILED = "2001";
	public static final String NO_INPUT = "3001";
	public static final String INVALID_FILE = "3002";
	public static final String READ_ERROR = "3003";
	public static final String UPLOAD_FAILED = "3004";
	public static final String PARTIAL_UPLOAD = "3005";
	public static final String FATAL = "9999";

	//message description

	public static final Map<String, String> MESSAGE_MAP = new HashMap<String,
String>();

	static {
		MESSAGE_MAP.put(SUCCESSFUL, "SUCCESSFUL");
		MESSAGE_MAP.put(NO_RESULT, "RESULT NOT FOUND");
		MESSAGE_MAP.put(TXN_FAILED, "TRANSACTION FAILED");
		MESSAGE_MAP.put(NO_INPUT, "NO INPUT PROVIDED");
		MESSAGE_MAP.put(INVALID_FILE, "INVALID FILE NAME OR TYPE");
		MESSAGE_MAP.put(READ_ERROR, "FILE COULD NOT BE READ");
		MESSAGE_MAP.put(UPLOAD_FAILED, "FILE UPLOAD FAILED");
		MESSAGE_MAP.put(PARTIAL_UPLOAD, "PARTIAL FILE UPLOAD");
	}

	//custom message description

	public static final String NOT_FOUND = "NOT FOUND";
	public static final String INSERT_FAILED = "INSERT FAILED";
	public static final String UPDATE_FAILED = "UPDATE FAILED";
	public static final String DELETE_FAILED = "DELETE FAILED";

	//message severity

	public static final String HIGH = "HIGH";
	public static final String MEDIUM = "MEDIUM";
	public static final String LOW = "LOW";
```

```
}
```

RestExceptionHandler.java

```
package com.ge.efs.contmgmt.exception;

import java.io.IOException;

import org.apache.log4j.Logger;
import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.ControllerAdvice;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.ResponseStatus;

import com.ge.efs.contmgmt.model.RestResponse;

@ControllerAdvice
public class RestExceptionHandler {

	public static final Logger LOGGER =
Logger.getLogger(RestExceptionHandler.class);

	@ExceptionHandler(DBException.class)
	@ResponseStatus(value=HttpStatus.INTERNAL_SERVER_ERROR)
	public @ResponseBody RestResponse sendDBErrorResponse(DBException ex) {
		LOGGER.error(ex.getMessage(), ex);
		RestResponse errRes = new RestResponse(ExceptionConstants.FATAL,
				ExceptionConstants.HIGH, ex.getMessage(), null);
		return errRes;

	}

	@ExceptionHandler(AppException.class)
	@ResponseStatus(value=HttpStatus.INTERNAL_SERVER_ERROR)
	public @ResponseBody RestResponse sendAppErrorResponse(AppException ex) {
		LOGGER.error(ex.getMessage(), ex);
		RestResponse errRes = new RestResponse(ExceptionConstants.FATAL,
				ExceptionConstants.HIGH, ex.getMessage(), null);
		return errRes;

	}

	@ExceptionHandler(IOException.class)
	@ResponseStatus(value=HttpStatus.INTERNAL_SERVER_ERROR)
	public @ResponseBody RestResponse sendIOErrorResponse(IOException ex) {
		LOGGER.error(ex.getMessage(), ex);
		RestResponse errRes = new RestResponse(ExceptionConstants.FATAL,
				ExceptionConstants.HIGH, ex.getMessage(), null);
		return errRes;

	}
}
```

PressReleaseMapper.java

```
package com.ge.efs.contmgmt.mappers;

import java.util.List;

import org.apache.ibatis.annotations.Param;
import org.springframework.stereotype.Repository;

import com.ge.efs.contmgmt.model.IndustryFocus;
import com.ge.efs.contmgmt.model.PressRelease;

//@Repository
public interface PressReleaseMapper {

	public PressRelease getPressReleaseById(@Param("pressReleaseId") Integer
pressReleaseId);
	public List<PressRelease> getLatestPressReleases(@Param("count") Integer
count);
	public List<IndustryFocus> getAllIndustryFoci();
	public Integer getCurrentPressReleaseId();
	public Integer createPressRelease(PressRelease pressRelease);
	public Integer editPressRelease(PressRelease pressRelease);
	public Integer deletePressRelease(@Param("pressReleaseId") Integer
pressReleaseId);

}
```

IndustryFocus.java

```
package com.ge.efs.contmgmt.model;

public class IndustryFocus {
	private Integer id;
	private String name;

	public Integer getId() {
		return id;
	}
	public void setId(Integer id) {
		this.id = id;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}


}
```

PressRelease.java

```java
package com.ge.efs.contmgmt.model;

public class PressRelease {

	private Integer pressReleaseId;
	private Integer industryFocusId;
	private String name;
	private String location;
	private String publishDate;
	private String body;
	private String contactInfo;
	private String quote;
	private String quoteAuthor;
	private String quoteAuthorTitle;
	private String imageUrl;
	private String pdfUrl;
	private Integer active;
	private String industryFocus;

	public Integer getPressReleaseId() {
		return pressReleaseId;
	}
	public void setPressReleaseId(Integer pressReleaseId) {
		this.pressReleaseId = pressReleaseId;
	}
	public Integer getIndustryFocusId() {
		return industryFocusId;
	}
	public void setIndustryFocusId(Integer industryFocusId) {
		this.industryFocusId = industryFocusId;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getLocation() {
		return location;
	}
	public void setLocation(String location) {
		this.location = location;
	}
	public String getPublishDate() {
		return publishDate;
	}
	public void setPublishDate(String publishDate) {
		this.publishDate = publishDate;
	}
	public String getBody() {
		return body;
	}
	public void setBody(String body) {
		this.body = body;
	}
```

```
	public String getContactInfo() {
		return contactInfo;
	}
	public void setContactInfo(String contactInfo) {
		this.contactInfo = contactInfo;
	}
	public String getQuote() {
		return quote;
	}
	public void setQuote(String quote) {
		this.quote = quote;
	}
	public String getQuoteAuthor() {
		return quoteAuthor;
	}
	public void setQuoteAuthor(String quoteAuthor) {
		this.quoteAuthor = quoteAuthor;
	}
	public String getQuoteAuthorTitle() {
		return quoteAuthorTitle;
	}
	public void setQuoteAuthorTitle(String quoteAuthorTitle) {
		this.quoteAuthorTitle = quoteAuthorTitle;
	}
	public String getImageUrl() {
		return imageUrl;
	}
	public void setImageUrl(String imageUrl) {
		this.imageUrl = imageUrl;
	}
	public String getPdfUrl() {
		return pdfUrl;
	}
	public void setPdfUrl(String pdfUrl) {
		this.pdfUrl = pdfUrl;
	}
	public Integer getActive() {
		return active;
	}
	public void setActive(Integer active) {
		this.active = active;
	}
	public String getIndustryFocus() {
		return industryFocus;
	}
	public void setIndustryFocus(String industryFocus) {
		this.industryFocus = industryFocus;
	}

}
```

RestResponse.java

```
package com.ge.efs.contmgmt.model;
```

```
import com.ge.efs.contmgmt.exception.ExceptionConstants;

public class RestResponse {
	private String message;
	private String severity;
	private String msgDesc;
	private Object payload;

	public RestResponse() {
		this.message = ExceptionConstants.SUCCESSFUL;
		this.severity = ExceptionConstants.LOW;
		this.msgDesc =
ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.SUCCESSFUL);
		this.payload = null;
	}

	public RestResponse(String message, String severity, String msgDesc, Object
payload) {
		this.message = message;
		this.severity = severity;
		this.msgDesc = msgDesc;
		this.payload = payload;
	}

	public String getMessage() {
		return message;
	}
	public void setMessage(String message) {
		this.message = message;
	}
	public String getSeverity() {
		return severity;
	}
	public Object getPayload() {
		return payload;
	}

	public void setSeverity(String severity) {
		this.severity = severity;
	}
	public String getMsgDesc() {
		return msgDesc;
	}
	public void setMsgDesc(String msgDesc) {
		this.msgDesc = msgDesc;
	}
	public void setPayload(Object payload) {
		this.payload = payload;
	}
}
```

CommonUploadServiceImpl.java

```
package com.ge.efs.contmgmt.service;
```

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;

import org.apache.log4j.Logger;
import org.springframework.web.multipart.MultipartFile;

import com.ge.efs.contmgmt.exception.ExceptionConstants;

//@Service("commonUpload")
//This is configured as a bean in the service-context.xml
public class CommonUploadServiceImpl implements FileUploadService {

	public static final Logger LOGGER =
Logger.getLogger(CommonUploadServiceImpl.class);

	private String uploadPathImg;
	private String uploadPathPdf;

	@Override
	public Integer uploadFile(MultipartFile file, String mediaType) throws
IOException {

		int uploadStatus = 0;
		//PATH for normal file upload
		String uploadPath = (mediaType.equals("IMAGE")) ? uploadPathImg :
uploadPathPdf;

		try {
		// Following is a simple upload of file using Spring Multipart
implementation
	       // Get the file and save it to the Path. This works for local drives or
		// drives which are mounted in the local PC.

		byte[] bytes = file.getBytes();
		Path path = Paths.get(uploadPath + file.getOriginalFilename());
		Files.write(path, bytes);
		uploadStatus = 1;

	  } catch (Exception e) {
		throw new IOException(
				ExceptionConstants.MESSAGE_MAP.get(

	ExceptionConstants.UPLOAD_FAILED)+file.getOriginalFilename(), e);
	  }

	  return uploadStatus;
	}

	public String getUploadPathImg() {
		return uploadPathImg;
	}

	public void setUploadPathImg(String uploadPathImg) {
```

```
		this.uploadPathImg = uploadPathImg;
	}

	public String getUploadPathPdf() {
		return uploadPathPdf;
	}

	public void setUploadPathPdf(String uploadPathPdf) {
		this.uploadPathPdf = uploadPathPdf;
	}
}
```

FileUploadService.java

```
package com.ge.efs.contmgmt.service;

import java.io.IOException;

import org.springframework.web.multipart.MultipartFile;

public interface FileUploadService {

	public Integer uploadFile(MultipartFile file, String mediaType) throws
IOException;

}
```

JCIFSUploadServiceImpl.java

```
package com.ge.efs.contmgmt.service;

import java.io.IOException;

import jcifs.smb.NtlmPasswordAuthentication;
import jcifs.smb.SmbFile;
import jcifs.smb.SmbFileOutputStream;

import org.apache.log4j.Logger;
import org.springframework.web.multipart.MultipartFile;

import com.ge.efs.contmgmt.exception.ExceptionConstants;

//@Service("jcifsUpload")
//This is configured as a bean in the service-context.xml
public class JCIFSUploadServiceImpl implements FileUploadService {

	private String uploadUser;
	private String uploadPassword;
	private String uploadPathImg;
	private String uploadPathPdf;

	public static final Logger LOGGER =
Logger.getLogger(JCIFSUploadServiceImpl.class);

	@Override
```

```
	public Integer uploadFile(MultipartFile file, String mediaType) throws
IOException {

		//The following is used to upload the file to a network location using
ID and password
		//This uses the JCIFS library
		// In a Production environment, use Functional SSO
		int uploadStatus = 0;
		//PATH for JCISF usage
		String uploadPath = (mediaType.equals("IMAGE")) ? uploadPathImg :
uploadPathPdf;

		try {
			byte[] bytes = file.getBytes();
			//Mention the user in terms of "DOMAIN;UserName:Password"
			String user = "LOGON;"+uploadUser+":"+"uploadPassword";
			NtlmPasswordAuthentication auth = new
NtlmPasswordAuthentication(user);
			String path = uploadPath + file.getOriginalFilename();

			LOGGER.info("File Path: " + uploadPath +
file.getOriginalFilename());

			SmbFile sFile = new SmbFile(path, auth);
			SmbFileOutputStream sfos = new SmbFileOutputStream(sFile);
			sfos.write(bytes);
			sfos.close();
			uploadStatus = 1;
		}
		catch (Exception e) {
		throw new IOException(
				ExceptionConstants.MESSAGE_MAP.get(

	ExceptionConstants.UPLOAD_FAILED)+file.getOriginalFilename(), e);
		}

		return uploadStatus;
	}

	public String getUploadUser() {
		return uploadUser;
	}

	public void setUploadUser(String uploadUser) {
		this.uploadUser = uploadUser;
	}

	public String getUploadPassword() {
		return uploadPassword;
	}

	public void setUploadPassword(String uploadPassword) {
		this.uploadPassword = uploadPassword;
	}
```

```
	public String getUploadPathImg() {
		return uploadPathImg;
	}

	public void setUploadPathImg(String uploadPathImg) {
		this.uploadPathImg = uploadPathImg;
	}

	public String getUploadPathPdf() {
		return uploadPathPdf;
	}

	public void setUploadPathPdf(String uploadPathPdf) {
		this.uploadPathPdf = uploadPathPdf;
	}
}
```

BooleanTypeHandler.java

```
package com.ge.efs.contmgmt.typehandler;

import java.sql.CallableStatement;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

import org.apache.ibatis.type.BaseTypeHandler;
import org.apache.ibatis.type.JdbcType;
import org.apache.ibatis.type.MappedJdbcTypes;
import org.apache.ibatis.type.MappedTypes;

@MappedJdbcTypes(JdbcType.VARCHAR)
@MappedTypes(Boolean.class)
public class BooleanTypeHandler extends BaseTypeHandler<Boolean> {

  private static final String YES = "Y";
  private static final String NO = "N";

  @Override
  public void setNonNullParameter(PreparedStatement ps, int i, Boolean parameter,
JdbcType jdbcType) throws SQLException {
    boolean b = parameter.booleanValue();
    ps.setString(i, b ? YES : NO);
  }

  @Override
  public Boolean getNullableResult(ResultSet rs, String columnName) throws
SQLException {
    return convertStringToBooelan(rs.getString(columnName));
  }

  @Override
  public Boolean getNullableResult(CallableStatement cs, int columnIndex) throws
SQLException {
    return convertStringToBooelan(cs.getString(columnIndex));
```

```
  }

  private Boolean convertStringToBooelan(String strValue) throws SQLException {
    if (YES.equalsIgnoreCase(strValue)) {
      return new Boolean(true);
    } else if (NO.equalsIgnoreCase(strValue)) {
      return new Boolean(false);
    } else {
      throw new SQLException("Unexpected value " + strValue + " found where " + YES +
" or " + NO + " was expected.");
    }
  }

}
```