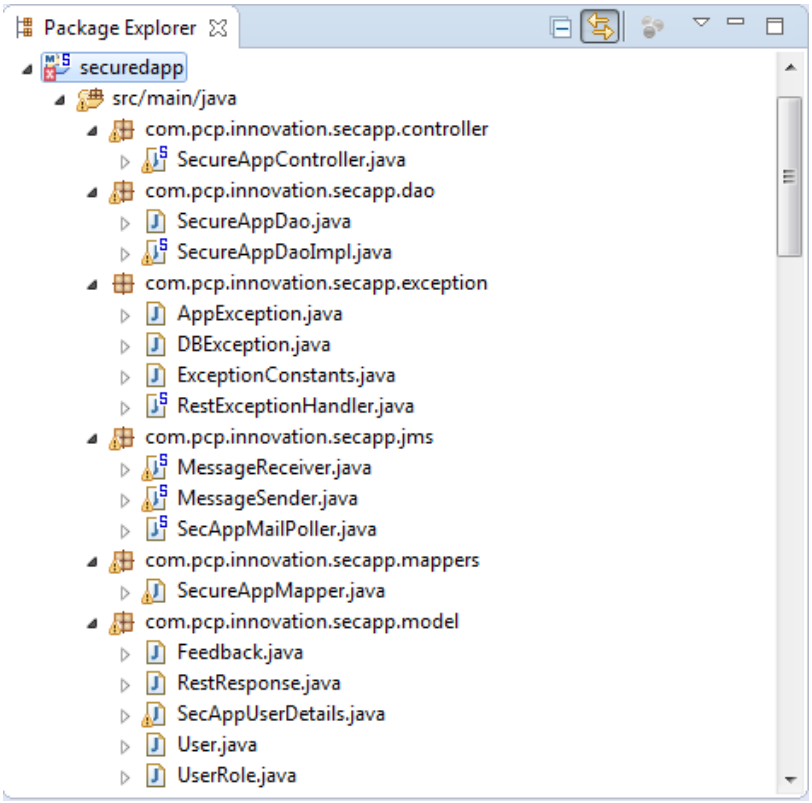
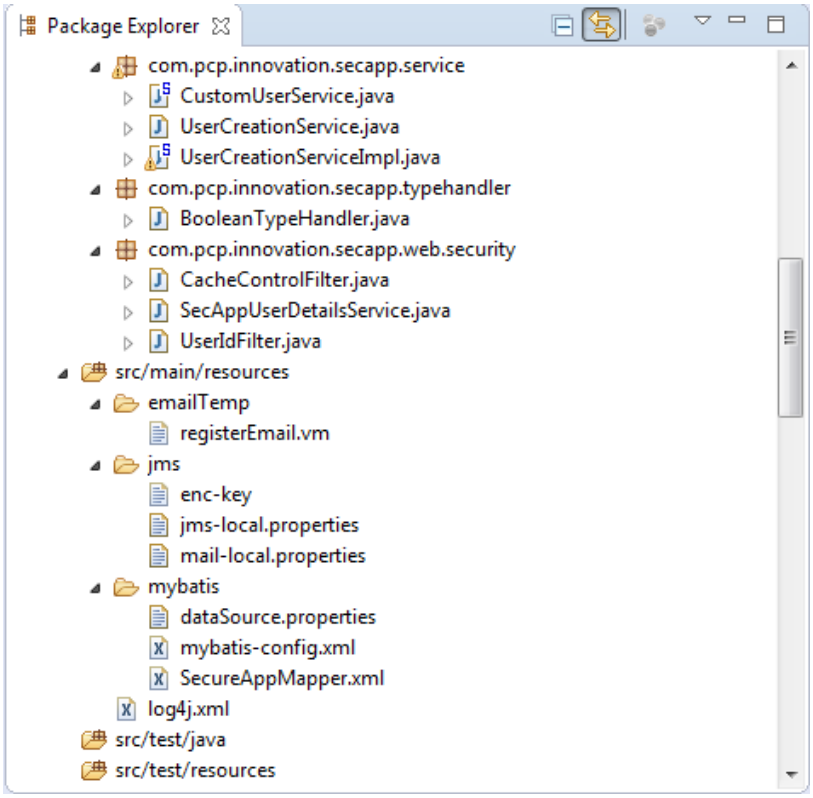
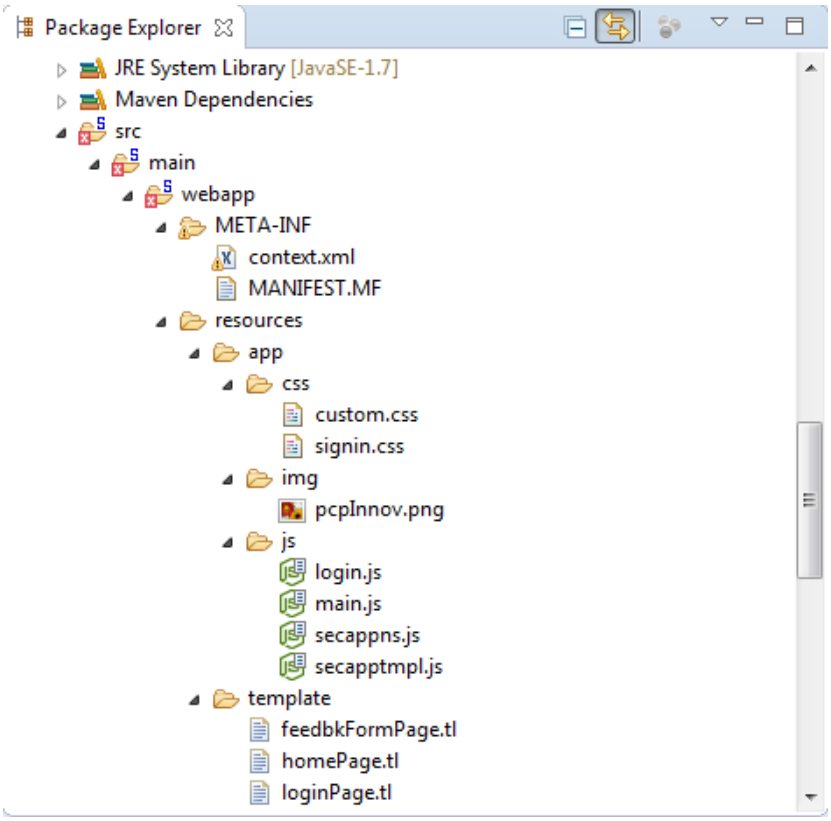
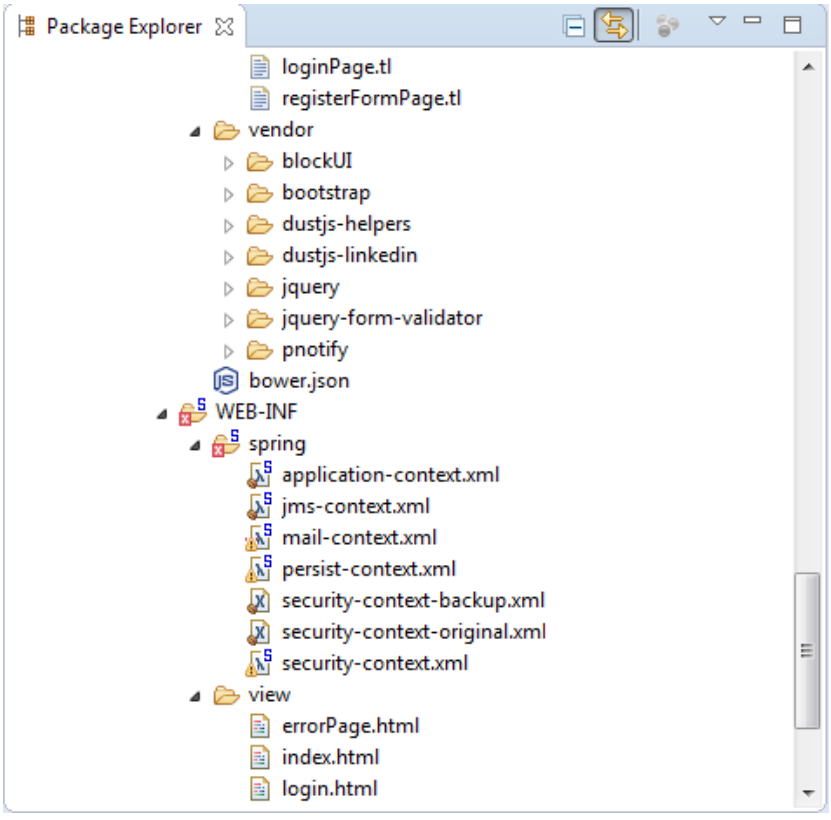
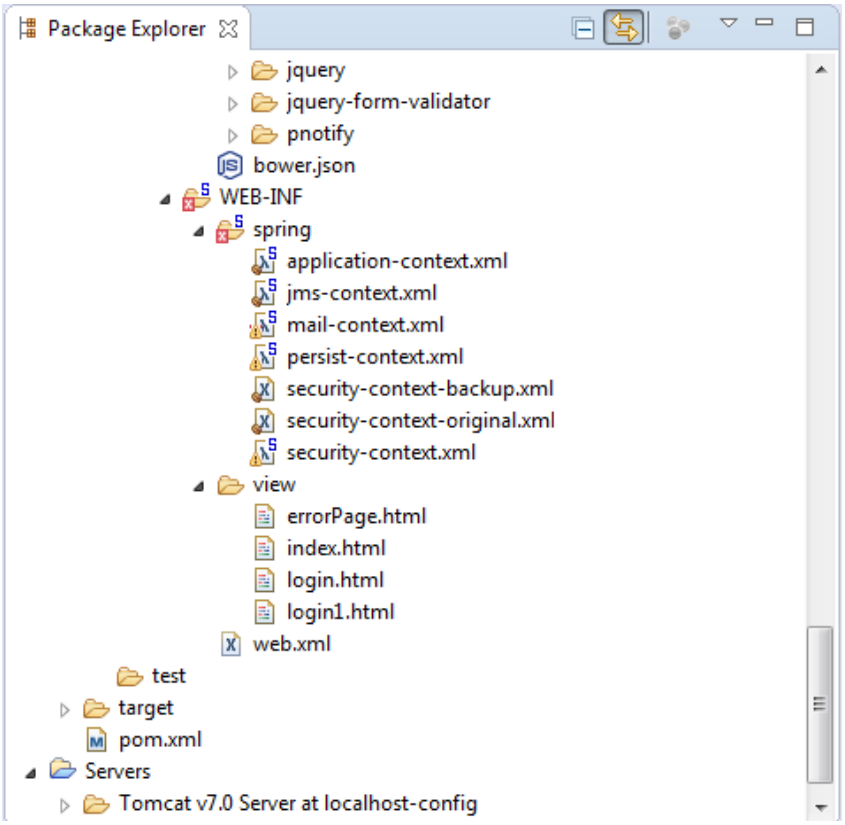
Project Structure






Package Explorer
JRE System Library [JavaSE-1.7]
Maven Dependencies
src
main
webapp
META-INF
context.xml
MANIFEST.MF
resources
app
css
custom.css
signin.css
img
pcpInnov.png
js
login.js
main.js
secappns.js
secapptmpl.js
template
feedbkFormPage.tl
homePage.tl
loginPage.tl


Package Explorer
loginPage.tl
registerFormPage.tl
vendor
blockUI
bootstrap
dustjs-helpers
dustjs-linkedin
jquery
jquery-form-validator
pnotify
bower.json
WEB-INF
spring
application-context.xml
jms-context.xml
mail-context.xml
persist-context.xml
security-context-backup.xml
security-context-original.xml
security-context.xml
view
errorPage.html
index.html
login.html



The Views

errorPage.html

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

index.html

```
<!DOCTYPE html>
<html>
<head>
	<title>My Secured App</title>
	<meta charset="utf-8">
	<meta http-equiv="X-UA-Compatible" content="IE=edge">
	<meta name="viewport" content="width=device-width,initial-scale=1">
	<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/app/js/secapptmpl.js"></script>
    <script src="resources/app/js/secappns.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/pcpInnov.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/pcpInnov.png"
style="display: inline-block;"></span>
                  <span style="display: inline-block;">My Secured App</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><a href="#Feedback">Feedback</a></li>
                  <li><a href="#Logout">Logout</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 -->

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

login.html

```
<!DOCTYPE html>
<html>
<head>
	<title>My Secured App</title>
	<meta charset="utf-8">
	<meta http-equiv="X-UA-Compatible" content="IE=edge">
	<meta name="viewport" content="width=device-width,initial-scale=1">
	<script src="resources/vendor/jquery/dist/jquery.min.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/blockUI/jquery.blockUI.js"></script>
	<script src="resources/app/js/secapptmpl.js"></script>
    <script src="resources/app/js/secappns.js"></script>
	<link href="resources/vendor/bootstrap/dist/css/bootstrap.css"
rel="stylesheet">
	<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 href="resources/app/css/signin.css" rel="stylesheet">
	<link rel="icon" href="resources/app/img/pcpInnov.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/pcpInnov.png"
style="display: inline-block;"></span>
	            <span style="display: inline-block;">My Secured App</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><a href="#">Login</a></li>
```

```
            <li><a href="#Register">Register</a></li>
              <li><a href="#Help">Help</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 -->
       <!-- Template -->
    </div> <!-- /container -->
<script src="resources/app/js/login.js"></script>
</body>
</html>
```

login1.html (Not Used)

```
<!DOCTYPE html>
<html>
<head>
    <title>Login Page</title>
    <script src="resources/vendor/jquery/dist/jquery.min.js"></script>
</head>
<body>
<h4>Please Enter your User Name and Password</h4>
<form action="check/security" method='POST'>
    Name: <input type="text" name="username"><br>
    Password: <input type="password" name="password"><br>
    <input type="submit" value="Submit">
</form>
<br>
<button id="logout" type="button">Logout</button>
<script>
      $(function() {
            $("#logout").click(function (e) {
                  alert("Hello!");
                  e.preventDefault();
                  $.ajax({
                type: "POST",
                url: "http://localhost:8080/securedapp/logout",
                success: function (data) {
                    console.log("SUCCESSFULLY LOGGED OUT");
                },
                error: function (jqXHR, textStatus, errorThrown) {
                 console.log("ERROR LOGGING OUT");
                },
                complete: function() {
                    console.log("LOGOUT FUNCTION COMPLETE");
                }

              });
            })
      })
</script>
</body>
```

```
</html>
```

The CSS

custom.css

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

.navbar-brand {
	padding-top: 5px;
}
.main-logo {
	height: 40px;
	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;
}
```

signin.css

```
body {
  padding-top: 40px;
  padding-bottom: 40px;
  background-color: #eee;
}

#registerFormPage {
  max-width: 800px;
  padding: 15px;
  margin: 0 auto;
}

.form-signin {
  max-width: 330px;
  padding: 15px;
  margin: 0 auto;
}
.form-signin .form-signin-heading,
.form-signin .checkbox {
  margin-bottom: 10px;
}
.form-signin .checkbox {
  font-weight: normal;
}
.form-signin .form-control {
  position: relative;
  height: auto;
  -webkit-box-sizing: border-box;
          box-sizing: border-box;
  padding: 10px;
  font-size: 16px;
}
.form-signin .form-control:focus {
  z-index: 2;
}
.form-signin input[type="text"] {
  margin-bottom: -1px;
  border-bottom-right-radius: 0;
  border-bottom-left-radius: 0;
}
.form-signin input[type="password"] {
  margin-bottom: 10px;
  border-top-left-radius: 0;
  border-top-right-radius: 0;
}
```

The Javascripts

login.js

```
//Sec App Scripts
//Author: Prabhu Purohit

SecApp.namespace("SecApp.Login");
SecApp.Login = {

    util: {
        getDate: function() {
            var d = new Date();
            var date = d.getFullYear() + '-' +
              ('0' + (d.getMonth() + 1)).slice(-2) + '-' +
              ('0' + d.getDate()).slice(-2);
            return date;
        }
    },

    model: {

        //Registration data
        regdnData : {},

        //Form data to be exchanged
        formData : {
            mode: '',
            regdn: {}
        },
    },

    view: {

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

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

        renderLoginFormEvents: function() {
              $("#loginSubmit").click(function(e){
                 e.preventDefault();
                 //Ajax POST request
                 SecApp.Login.controller.submitLoginForm();
             });
        },

        renderRegisterFormPage: function(formData) {
            //Compiled with dustc compiler
            //dustc --pwd=template/ template/*.tl -o app/js/secapptmpl.js
            dust.render('registerFormPage', formData, function(err, out) {
                        $("#mainContent").html(out);
            });
```

```
    },

    registerRegnFormEvents: function() {
        $("#pageActionSubmit").click(function(e){
            e.preventDefault();
            //Ajax POST request
            if (SecApp.Login.controller.isValidRegnForm()) {
                SecApp.Login.controller.submitRegnCreateForm();
            }
        });

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

    registerRegnFormValidator: function() {
        $.validate({
            form : '#registerForm',
            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);
                SecApp.Login.view.sendNotification("error", "Error:
"+respData.message, "Message: "+respData.severity+": "+respData.msgDesc);
            }
            else if (jqXHR.readyState === 0) {
                console.log("FAILURE: Network");
                SecApp.Login.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);
                SecApp.Login.view.sendNotification("error", "Error:
"+errorData.message, "Message: "+errorData.severity+": "+errorData.msgDesc);
            }
            else {
                console.log("FAILURE: "+errorThrown);
                SecApp.Login.view.sendNotification("error", "Error: HTTP", "Message:
"+errorThrown);
            }
        }

    },

    controller: {

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

        getLatestRegnFormValues: function() {
            return {
             "userName": $("#userName").val().trim(),
                "userPass": $("#userPass").val().trim(),
                "secretQuest": $("#secretQuest").val().trim(),
                "secretAns": $("#secretAns").val().trim(),
                "isActive": "Y"
            };
        },

        submitLoginForm: function() {
             var contextRoot = SecApp.contextRoot;
            var url = contextRoot + "/check/security";
            var formData = $('#loginForm')
            $.ajax({
                type: "POST",
                url: url,
```

```
                data: formData.serialize(),
                /*beforeSend: function(){
                    $.blockUI();
                },*/
                success: function (data) {
                    console.log("SUCCESSFULLY LOGGED IN");
                    SecApp.Login.view.sendNotification("success", "Success: LOGIN",
data.msgDesc);
                    //Show the Home page
                    window.location.href = SecApp.contextRoot + '/';
                },
                error: function (jqXHR, textStatus, errorThrown) {
                    SecApp.Login.exHandler.handleError(null, jqXHR, textStatus,
errorThrown);
                },
                complete: function() {
                    console.log("LOGIN FUNCTION COMPLETE");
                    //$.unblockUI();
                }

            });
        },

        submitRegnCreateForm : function() {
            var contextRoot = SecApp.contextRoot;
            var url = contextRoot + "/registration?action=create";
            var params = SecApp.Login.controller.getLatestRegnFormValues();
            SecApp.Login.model.formData.feedback = 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") {
                        SecApp.Login.exHandler.handleError(data,null,null,null);
                    } else {
                        if (params.imageUrl || params.pdfUrl) {
                            SecApp.Login.controller.uploadFiles();
                        }
                        console.log("SUCCESSFULLY CREATED");
                        SecApp.Login.view.sendNotification("success", "Success:
REGISTER", data.msgDesc);
                        //Show the List page
                        window.location.hash = '#';
                    }
```

```
                },
                error: function (jqXHR, textStatus, errorThrown) {
                    SecApp.Login.exHandler.handleError(null, jqXHR, textStatus,
errorThrown);
                },
                complete: function() {
                    console.log("REGISTER 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() {
                    SecApp.Login.view.renderLoginPage();
                    SecApp.Login.view.renderLoginFormEvents();
                },

                // The Homepage. This is for IE compatibility.
                '#': function() {
                    SecApp.Login.view.renderLoginPage();
                    SecApp.Login.view.renderLoginFormEvents();
                },

                // Register page
                '#Register': function() {
                    SecApp.Login.view.renderRegisterFormPage();
                    SecApp.Login.view.registerRegnFormEvents();
                    SecApp.Login.view.registerRegnFormValidator();
                },
            };

            // 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");
                SecApp.Login.view.sendNotification("error", "Error!", "URL mapping
not found");
            }
        }
    }
```

```
};

$(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(){
        SecApp.Login.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');
})
```

main.js

```
//Sec App Scripts
//Author: Prabhu Purohit

SecApp.namespace("SecApp.Main");
SecApp.Main = {

    util: {
        getDate: function() {
            var d = new Date();
            var date = d.getFullYear() + '-' +
              ('0' + (d.getMonth() + 1)).slice(-2) + '-' +
              ('0' + d.getDate()).slice(-2);
            return date;
        }
    },

    model: {

        //Feedback data
        feedbackData : {},

        //Form data to be exchanged
        formData : {
            mode: '',
            feedback: {}
        },
    },

    view: {

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

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

```
    },

    renderFeedbackFormPage: function(formData) {
        //Compiled with dustc compiler
        //dustc --pwd=template/ template/*.tl -o app/js/secapptmpl.js
        dust.render('feedbkFormPage', formData, function(err, out) {
                    $("#mainContent").html(out);
        });
    },

    registerFdbkFormEvents: function() {
        $("#pageActionSubmit").click(function(e){
            e.preventDefault();
            //Ajax POST request
            if (SecApp.Main.controller.isValidFdbkForm()) {
                SecApp.Main.controller.submitFdbkCreateForm();
            }
        });

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

    registerFdbkFormValidator: function() {
        $.validate({
            form : '#feedbackForm',
            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);
                SecApp.Main.view.sendNotification("error", "Error:
"+respData.message, "Message: "+respData.severity+": "+respData.msgDesc);
            }
            else if (jqXHR.readyState === 0) {
                console.log("FAILURE: Network");
                SecApp.Main.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);
                SecApp.Main.view.sendNotification("error", "Error:
"+errorData.message, "Message: "+errorData.severity+": "+errorData.msgDesc);
            }
            else {
                console.log("FAILURE: "+errorThrown);
                SecApp.Main.view.sendNotification("error", "Error: HTTP", "Message:
"+errorThrown);
            }
        }

    },

    controller: {

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

        getLatestFdbkFormValues: function() {
            return {
             "feedbackId": null,
                "feedbackName": $("#feedbackName").val().trim(),
                "feedbackText": $("#feedbackText").val().trim(),
                "publishDate": SecApp.Main.util.getDate(),
            };
        },
```

```
        submitFdbkCreateForm : function() {
            var contextRoot = SecApp.contextRoot;
            var url = contextRoot + "/feedback?action=create";
            var params = SecApp.Main.controller.getLatestFdbkFormValues();
            SecApp.Main.model.formData.feedback = 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") {
                        SecApp.Main.exHandler.handleError(data,null,null,null);
                    } else {
                        if (params.imageUrl || params.pdfUrl) {
                            SecApp.Main.controller.uploadFiles();
                        }
                        console.log("SUCCESSFULLY CREATED");
                        SecApp.Main.view.sendNotification("success", "Success:
CREATE", data.msgDesc);
                        //Show Home page
                        window.location.hash = '#';
                    }
                },
                error: function (jqXHR, textStatus, errorThrown) {
                    SecApp.Main.exHandler.handleError(null, jqXHR, textStatus,
errorThrown);
                },
                complete: function() {
                    console.log("CREATE FUNCTION COMPLETE");
                    $.unblockUI();
                }

            });
        },

        initiateLogout: function() {
             var contextRoot = SecApp.contextRoot;
            var url = contextRoot + "/logout";
            $.ajax({
                type: "POST",
                url: url,
                success: function (data) {
                    if (data.message !== "0000") {
                        SecApp.Main.exHandler.handleError(data,null,null,null);
                    } else {
```

```
                        console.log("SUCCESSFULLY LOGGED OUT");
                           //Show Login Page
                           window.location.href = SecApp.contextRoot + '/login';
                    }
                },
                error: function (jqXHR, textStatus, errorThrown) {
                    SecApp.Main.exHandler.handleError(null, jqXHR, textStatus,
errorThrown);
                },
                complete: function() {
                    console.log("LOGOUT 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() {
                    SecApp.Main.view.renderHomePage();
                },

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

                // Feedback page.
                '#Feedback': function() {
                    SecApp.Main.view.renderFeedbackFormPage();
                    SecApp.Main.view.registerFdbkFormEvents();
                    SecApp.Main.view.registerFdbkFormValidator();
                },

                // Logout
                '#Logout': function() {
                    SecApp.Main.controller.initiateLogout();
                },
            };

            // 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");
                SecApp.Main.view.sendNotification("error", "Error!", "URL mapping not
found");
            }
        }
    }

};

$(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(){
        SecApp.Main.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');
})
```

secappns.js

```
var SecApp = SecApp || {};
SecApp.namespace = function(ns_string) {
    var parts = ns_string.split('.'), parent = SecApp, i;
    if (parts[0] == "SecApp") {
        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;

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

secapptmpl.js

```
(function(dust){dust.register("loginPage",body_0);function body_0(chk,ctx){return
chk.w("<div class=\"container\" id=\"loginPage\"><form id=\"loginForm\" class=\"form-
signin\" action=\"check/security\" method='POST'><h2 class=\"form-signin-
heading\">Please sign in</h2><label for=\"inputUser\" class=\"sr-only\">User
Name</label><input type=\"text\" id=\"inputUser\" name=\"username\" class=\"form-
```

```
control\" placeholder=\"User Name\" required autofocus><label for=\"inputPassword\"
class=\"sr-only\">Password</label><input type=\"password\" id=\"inputPassword\"
name=\"password\" class=\"form-control\" placeholder=\"Password\" required><div
class=\"checkbox\"><label><input type=\"checkbox\" name=\"remember-me\"> Remember
me</label></div><button id=\"loginSubmit\" class=\"btn btn-lg btn-primary btn-block\"
type=\"submit\">Sign in</button></form></div>");}body_0.__dustBody=!0;return
body_0}(dust));
(function(dust){dust.register("feedbkFormPage",body_0);function
body_0(chk,ctx){return chk.w("<div class=jumbotron id=\"feedbkFormPage\"><h2
id=\"formHeader\">Submit Feedback</h2><br><form id=\"feedbackForm\" class=\"form-
horizontal\"><div class=\"form-group publishDate\" hidden><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\"></div></div><div
class=\"form-group\"><label for=\"feedbackName\" class=\"col-sm-2 control-
label\">Feedback Name</label><div class=\"col-sm-10\"><input type=\"text\"
class=\"form-control\" id=\"feedbackName\" placeholder=\"Feedback Name\" data-
validation=\"required\"></div></div><div class=\"form-group\"><label
for=\"feedbackText\" class=\"col-sm-2 control-label\">Feedback Text</label><div
class=\"col-sm-10\"><textarea class=\"form-control\" rows=\"5\" id=\"feedbackText\"
placeholder=\"Feedback Text\" data-validation=\"length\" data-validation-
length=\"max255\"></textarea></div></div><button id=\"pageActionSubmit\" class=\"btn
btn-primary page-action\" type=\"button\"><span class=\"glyphicon glyphicon-floppy-
disk\" aria-hidden=\"true\"></span> Submit</button><button id=\"pageActionCancel\"
class=\"btn btn-primary page-action\" type=\"button\"><span class=\"glyphicon
glyphicon-remove\" aria-hidden=\"true\"></span>
Cancel</button></form></div>");}body_0.__dustBody=!0;return body_0}(dust));
(function(dust){dust.register("homePage",body_0);function body_0(chk,ctx){return
chk.w("<!-- 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>PCP Innov Secured App</h2><br><p>This is a
secured website</p><p>Powered by Spring Security</p><p>Designed with Static HTML and
Ajax Login</p><br><p><a class=\"btn btn-lg btn-primary\" href=\"#Feedback\"
role=\"button\">Give Feedback &raquo;</a></p></div><!-- </script> --
>");}body_0.__dustBody=!0;return body_0}(dust));
(function(dust){dust.register("registerFormPage",body_0);function
body_0(chk,ctx){return chk.w("<div class=jumbotron id=\"registerFormPage\"><h2
id=\"formHeader\">New User Registration</h2><br><form id=\"registerForm\"
class=\"form-horizontal\"><div class=\"form-group\"><label for=\"userName\"
class=\"col-sm-2 control-label\">User Name</label><div class=\"col-sm-10\"><input
type=\"text\" class=\"form-control\" id=\"userName\" placeholder=\"User Name\" data-
validation=\"required\"></div></div><div class=\"form-group\"><label for=\"userPass\"
class=\"col-sm-2 control-label\">Password</label><div class=\"col-sm-10\"><input
type=\"password\" class=\"form-control\" id=\"userPass\" placeholder=\"Password\"
data-validation=\"required\"></div></div><div class=\"form-group\"><label
for=\"secretQuest\" class=\"col-sm-2 control-label\">Secret Question</label><div
class=\"col-sm-10\"><select class=\"form-control\" id=\"secretQuest\" data-
validation=\"required\"><option value=\"\" selected>Select One</option><option
value=\"1\">What is the name of the city you were born?</option><option
value=\"2\">What is the make of your first vehicle?</option><option value=\"3\">What
is your mothers maiden name?</option></select></div></div><div class=\"form-
group\"><label for=\"secretAns\" class=\"col-sm-2 control-label\">Secret
Answer</label><div class=\"col-sm-10\"><input type=\"password\" class=\"form-
control\" id=\"secretAns\" placeholder=\"Secret Answer\" data-
validation=\"required\"></div></div><br><button id=\"pageActionSubmit\" class=\"btn
btn-primary page-action\" type=\"button\"><span class=\"glyphicon glyphicon-floppy-
```

```
disk\" aria-hidden=\"true\"></span> Submit</button><button id=\"pageActionCancel\"
class=\"btn btn-primary page-action\" type=\"button\"><span class=\"glyphicon
glyphicon-remove\" aria-hidden=\"true\"></span>
Cancel</button></form></div>");}body_0.__dustBody=!0;return body_0}(dust));
```

The Templates

feedbkFormPage.tl

```
<div class=jumbotron id="feedbkFormPage">
  <h2 id="formHeader">Submit Feedback</h2>
  <br>
  <form id="feedbackForm" class="form-horizontal">
    <div class="form-group publishDate" hidden>
      <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">
      </div>
    </div>
    <div class="form-group">
      <label for="feedbackName" class="col-sm-2 control-label">Feedback Name</label>
      <div class="col-sm-10">
        <input type="text" class="form-control" id="feedbackName"
placeholder="Feedback Name" data-validation="required">
      </div>
    </div>
    <div class="form-group">
      <label for="feedbackText" class="col-sm-2 control-label">Feedback Text</label>
      <div class="col-sm-10">
        <textarea class="form-control" rows="5" id="feedbackText"
placeholder="Feedback Text" data-validation="length" data-validation-
length="max255"></textarea>
      </div>
    </div>

    <button id="pageActionSubmit" class="btn btn-primary page-action"
type="button"><span class="glyphicon glyphicon-floppy-disk" aria-
hidden="true"></span> Submit</button>
    <button id="pageActionCancel" class="btn btn-primary page-action"
type="button"><span class="glyphicon glyphicon-remove" aria-hidden="true"></span>
Cancel</button>
  </form>
</div>
```

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>PCP Innov Secured App</h2>
           <br>
           <p>This is a secured website</p>
           <p>Powered by Spring Security</p>
```

```
          <p>Designed with Static HTML and Ajax Login</p>
          <br>
          <p>
            <a class="btn btn-lg btn-primary" href="#Feedback" role="button">Give
Feedback &raquo;</a>
          </p>
      </div>
<!-- </script> -->
```

loginPage.tl

```
<div class="container" id="loginPage">
  <form id="loginForm" class="form-signin" action="check/security" method='POST'>
    <h2 class="form-signin-heading">Please sign in</h2>
    <label for="inputUser" class="sr-only">User Name</label>
    <input type="text" id="inputUser" name="username" class="form-control"
placeholder="User Name" required autofocus>
    <label for="inputPassword" class="sr-only">Password</label>
    <input type="password" id="inputPassword" name="password" class="form-control"
placeholder="Password" required>
    <div class="checkbox">
      <label>
        <input type="checkbox" name="remember-me"> Remember me
      </label>
    </div>
    <button id="loginSubmit" class="btn btn-lg btn-primary btn-block"
type="submit">Sign in</button>
  </form>
</div>
```

registerFormPage.tl

```
<div class=jumbotron id="registerFormPage">
  <h2 id="formHeader">New User Registration</h2>
  <br>
  <form id="registerForm" class="form-horizontal">
    <div class="form-group">
      <label for="userName" class="col-sm-2 control-label">User Name</label>
      <div class="col-sm-10">
        <input type="text" class="form-control" id="userName" placeholder="User Name"
data-validation="required">
      </div>
    </div>
    <div class="form-group">
      <label for="userPass" class="col-sm-2 control-label">Password</label>
      <div class="col-sm-10">
        <input type="password" class="form-control" id="userPass"
placeholder="Password" data-validation="required">
      </div>
    </div>
       <div class="form-group">
      <label for="secretQuest" class="col-sm-2 control-label">Secret Question</label>
      <div class="col-sm-10">
        <select class="form-control" id="secretQuest" data-validation="required">
            <option value="" selected>Select One</option>
```

```
            <option value="1">What is the name of the city you were born?</option>
            <option value="2">What is the make of your first vehicle?</option>
            <option value="3">What is your mothers maiden name?</option>
        </select>
      </div>
    </div>
    <div class="form-group">
      <label for="secretAns" class="col-sm-2 control-label">Secret Answer</label>
      <div class="col-sm-10">
        <input type="password" class="form-control" id="secretAns"
placeholder="Secret Answer" data-validation="required">
      </div>
    </div>
    <br>
    <button id="pageActionSubmit" class="btn btn-primary page-action"
type="button"><span class="glyphicon glyphicon-floppy-disk" aria-
hidden="true"></span> Submit</button>
    <button id="pageActionCancel" class="btn btn-primary page-action"
type="button"><span class="glyphicon glyphicon-remove" aria-hidden="true"></span>
Cancel</button>
  </form>
</div>
```

bower.json

```
{
  "name": "securedapp",
  "authors": [
    "Prabhu Purohit"
  ],
  "description": "This website is designed to test Spring security",
  "main": "index.html",
  "keywords": [
    "pcp",
    "innovation",
    "securedapp"
  ],
  "license": "PCP Innovation",
  "homepage": "http://securedapp.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#*",
    "jquery-form-validator": "^2.3.74"
  }
```

}

Spring Contexts

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.pcp.innovation.secapp.controller,
		com.pcp.innovation.secapp.dao,
		com.pcp.innovation.secapp.service,
		com.pcp.innovation.secapp.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 -->
	<!-- There is an alternate approach by which related files can be placed
	in the respective context files. Please check comments in jms-context.xml and
	persist-context.xml -->

	<!-- <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:jms/jms-local.properties</value>
	        </list>
	    </property>
	</bean> -->

	<!-- <beans profile="local">
		JMS Properties
		<context:property-placeholder order="1"
				ignore-unresolvable="true"
				location="classpath:jms/jms-local.properties" />
		Data Source Properties
		<context:property-placeholder order="2"
				location="classpath:mybatis/dataSource.properties" />
		Mail Properties
		<util:properties id="mailProperties"
			location="classpath:jms/mail-local.properties" />
	</beans> -->

</beans>
```

jms-context.xml

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

```
        http://activemq.apache.org/schema/core
        http://activemq.apache.org/schema/core/activemq-core-5.5.0.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context-4.0.xsd
        http://www.springframework.org/schema/util
        http://www.springframework.org/schema/util/spring-util-4.0.xsd
        http://www.springframework.org/schema/jms
        http://www.springframework.org/schema/jms/spring-jms.xsd
        http://www.springframework.org/schema/tx
        http://www.springframework.org/schema/tx/spring-tx.xsd">

        <!-- Activemq connection factory -->
        <bean id="amqConnectionFactory"
class="org.apache.activemq.ActiveMQConnectionFactory">
              <constructor-arg index="0" value="${jms.broker.url}" />
        </bean>

        <!-- ConnectionFactory Definition -->
        <bean id="connectionFactory"
              class="org.springframework.jms.connection.CachingConnectionFactory">
              <constructor-arg ref="amqConnectionFactory" />
        </bean>

        <!-- Sample configuration of JMS with ActiveMQ Connection Pooling -->
        <!-- <bean id="connectionFactory"
              class="org.apache.activemq.pool.PooledConnectionFactory" destroy-
method="stop">
              <constructor-arg value="tcs://localhost:61616" />
        </bean> -->

        <!-- Default Destination Queue Definition -->
        <bean id="defaultDestination"
class="org.apache.activemq.command.ActiveMQQueue">
              <constructor-arg index="0" value="${jms.queue.name}" />
        </bean>

        <!-- JmsTemplate Definition -->
        <bean id="jmsTemplate" class="org.springframework.jms.core.JmsTemplate">
              <property name="connectionFactory" ref="connectionFactory" />
              <property name="defaultDestination" ref="defaultDestination" />
              <property name="sessionTransacted" value="false" />
              <property name="receiveTimeout" value="5000" />
        </bean>

        <!-- Message Sender Definition -->
        <bean id="messageSender" class="com.pcp.innovation.secapp.jms.MessageSender">
              <constructor-arg index="0" ref="jmsTemplate" />
        </bean>

        <!-- Message Receiver Definition -->
      <bean id="messageReceiver" class="com.pcp.innovation.secapp.jms.MessageReceiver">
      </bean>

      <bean class="org.springframework.jms.listener.SimpleMessageListenerContainer">
          <property name="connectionFactory" ref="connectionFactory" />
```

```
        <property name="destinationName" value="${jms.queue.name}" />
        <property name="messageListener" ref="messageReceiver" />
    </bean>

    <beans profile="local">
            <!-- JMS Properties -->
            <!-- In case multiple properties elements are present in the Spring
context,
            there are a few best practices that should be followed:
            the order attribute needs to be specified to fix the order in which
            these are processed by Spring all property placeholders minus the last
one (highest order)
            should have ignore-unresolvable=”true” to allow the resolution mechanism
to pass
            to others in the context without throwing an exception -->
            <context:property-placeholder order="1"
                        ignore-unresolvable="true"
                        location="classpath:jms/jms-local.properties" />
            <!-- Mail Properties if required in the Service Implementation -->
            <!-- These properties have to be accessed with @Resource annotation -->
            <!-- <util:properties id="mailProperties"
                  location="classpath:jms/mail-local.properties" /> -->
      </beans>

</beans>
```

mail-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:encryption="http://www.jasypt.org/schema/encryption"
       xmlns:int="http://www.springframework.org/schema/integration"
       xmlns:int-mail="http://www.springframework.org/schema/integration/mail"
       xsi:schemaLocation="http://www.springframework.org/schema/util
http://www.springframework.org/schema/util/spring-util.xsd
          http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-4.0.xsd
          http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context.xsd
          http://www.springframework.org/schema/integration
http://www.springframework.org/schema/integration/spring-integration-4.0.xsd
          http://www.jasypt.org/schema/encryption
http://www.jasypt.org/schema/encryption/jasypt-spring31-encryption-1.xsd
          http://www.springframework.org/schema/integration/mail
http://www.springframework.org/schema/integration/mail/spring-integration-mail-
4.0.xsd">

       <!-- application encryptor and algorithm used for decrypting sensitive
properties -->
       <encryption:string-encryptor id="appEncryptor"
             algorithm="PBEWITHMD5ANDDES" password="A16tSJoEJmQ3Lt1r/XVEXgAX13Y6Ngbh"
/>
```

```
	<!-- All property placeholders minus the last one (highest order)
		should have ignore-unresolvable=”true” to allow the resolution mechanism
to pass
		to others in the context without throwing an exception -->
	<encryption:encryptable-property-placeholder order="2"
			encryptor="appEncryptor" ignore-unresolvable="true"
			location="classpath:jms/mail-local.properties" />

    <!-- Default mail properties are defined here -->
    <bean id="javaMailSender"
class="org.springframework.mail.javamail.JavaMailSenderImpl">
        <property name="host" value="${smtp.host}" />
        <!-- <property name="protocol" value="smtp" /> -->
        <property name="port" value="25" />
        <property name="username" value="${smtp.user}" />
        <property name="password" value="${smtp.password}" />
        <property name="javaMailProperties">
            <props>
                <prop key="mail.transport.protocol">smtp</prop>
                <prop key="mail.smtp.auth">true</prop>
                <prop key="mail.smtp.starttls.enable">false</prop>
                <prop key="mail.debug">false</prop>
            </props>
        </property>
    </bean>

    <bean id="velocityEngine"
          class="org.springframework.ui.velocity.VelocityEngineFactoryBean">
          <property name="velocityProperties">
                <props>
                      <prop key="resource.loader">class</prop>
                      <prop key="class.resource.loader.class">

	org.apache.velocity.runtime.resource.loader.ClasspathResourceLoader
                      </prop>
                </props>
          </property>
	</bean>

	<!-- BEGIN: Verify Registration e-Mail: POP3 email channel configuration -->
	<int:channel id="receivePop3Channel"></int:channel>

	<util:properties id="javaMailProperties">
          <prop key="mail.pop3.socketFactory.fallback">false</prop>
          <prop key="mail.debug">false</prop>
          <prop key="mail.pop3.port">995</prop>
          <prop
key="mail.pop3.socketFactory.class">javax.net.ssl.SSLSocketFactory</prop>
          <prop key="mail.pop3.socketFactory.port">995</prop>
          <prop key="mail.pop3.ssl.enable">false</prop>
          <prop key="mail.debug.auth">false</prop>
	</util:properties>
```

```
	<!-- We can make auto-startup="false" and start with a Service Application UI
-->
	<int-mail:inbound-channel-adapter id="pop3Adaptor"
		store-uri="${pop3.store-uri}" java-mail-properties="javaMailProperties"
		channel="receivePop3Channel" should-delete-messages="true"
		auto-startup="true">
		<!-- Will poll every 20 seconds -->
		<int:poller fixed-rate="20000" />

	</int-mail:inbound-channel-adapter>
	<!-- END: Verify Registration e-Mail: POP3 email channel configuration -->

	<bean id="secAppMailPoller"
		class="com.pcp.innovation.secapp.jms.SecAppMailPoller"
		init-method="initPoller">
	</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.pcp.innovation.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.pcp.innovation.secapp.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">

		<!-- In case multiple properties elements are present in the Spring
context,
		there are a few best practices that should be followed:
		the order attribute needs to be specified to fix the order in which
		these are processed by Spring all property placeholders minus the last
one (highest order)
		should have ignore-unresolvable=”true” to allow the resolution mechanism
to pass
		to others in the context without throwing an exception -->

		<context:property-placeholder order="3"
				location="classpath:mybatis/dataSource.properties" />

		<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="stage,prod">
		<jee:jndi-lookup id="dataSource" jndi-
name="java:/comp/env/jdbc/securedAppDS" />
	</beans>

</beans>
```

security-context-backup.xml (Not Used)

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/security"
```

```
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:beans="http://www.springframework.org/schema/beans"
	xsi:schemaLocation="
	http://www.springframework.org/schema/security
http://www.springframework.org/schema/security/spring-security-3.2.xsd
	http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

	<!-- enable code below if we want to protect methods with @PreFilter,
@PreAuthorize,
		@PostFilter, @PostAuthorize, commented spring root and expression
handler
		override -->
	<global-method-security pre-post-annotations="enabled">
	</global-method-security>

	<http pattern="/resources/**" security="none" />
	<http pattern="/security/**" security="none" />

	<!-- The URL rewriting can be prevented by adding the following conf to
		http element -->
	<!-- disable-url-rewriting="true" -->
	<!-- To use Spring SpEL include use-expressions="true" -->
	<http auto-config="true" use-expressions="true">

		<!-- This is for concurrency management -->
		<!-- If the second authentication takes place through another non-
interactive
			mechanism, such as “remember-me”, an “unauthorized” (402) error
will be sent
			to the client. If instead you want to use an error page, you can
add the
			attribute session-authentication-error-url to the session-
management element. -->

		<!-- session-management configuration attribute invalid-session-
url="/invalidSession"
			causes a problem during logout as the session gets invalidated
and upon hitting
			the logout-success-url it goes to invalid-session-url -->

		<session-management
			session-authentication-error-url="/sessionAuthError">
			<!-- By specifying error-if-maximum-exceeded="true", the second
login
				will fail and the authentication-failure-url will be
called. If error-if-maximum-exceeded="true"
				is not specified, on a successful second login session it
will invalidate
				the first session. So trying the first session will call
invalid-session-url -->
			<concurrency-control max-sessions="1"
				error-if-maximum-exceeded="true" expired-
url="/sessionExpired" />
		</session-management>
```

```
            <!-- <intercept-url pattern="/" access="isAuthenticated()" /> -->
            <intercept-url pattern="/" access="hasRole('ROLE_USER')" />
            <intercept-url pattern="/login" access="permitAll" />
            <intercept-url pattern="/admin**" access="hasRole('ROLE_USER')" />

            <!-- Handler for insufficient privilege -->
            <access-denied-handler error-page="/access/error" />

            <!-- authentication-failure-handler-ref = authFailHandlerBean
authentication-success-handler-ref
                  = authSuccHandlerBean -->

            <form-login login-processing-url="/check/security"
                  login-page="/login" authentication-failure-url="/login/error"
                  username-parameter="username" password-parameter="password" />

            <!-- success-handler-ref = logoutSuccessHandlerBean -->
            <!-- This will be handled through an Ajax logout request -->
            <!-- <logout logout-success-url="/login?logout" invalidate-
session="true"
                  logout-url="/logout" delete-cookies="JSESSIONID" /> -->

            <!-- Remember Me Token valid for 1 day -->
            <remember-me
                  key="secAppKey"
                  data-source-ref="dataSource"
                  remember-me-parameter="remember-me"
                  token-validity-seconds="43200"/>
      </http>

      <authentication-manager>
            <authentication-provider ref="authProvider" />
      </authentication-manager>

      <beans:bean id="authProvider"

      class="org.springframework.security.authentication.dao.DaoAuthenticationProvid
er">
            <beans:property name="userDetailsService" ref="customUserService" />
            <beans:property name="passwordEncoder" ref="encoder" />
      </beans:bean>

      <beans:bean id="encoder"
class="org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder">
            <beans:constructor-arg name="strength" value="10" />
      </beans:bean>

      <beans:bean id="customUserService"
            class="com.pcp.innovation.secapp.service.CustomUserService" />

</beans:beans>
```

security-context-original.xml (Not Used)

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/security"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:beans="http://www.springframework.org/schema/beans"
	xsi:schemaLocation="
	http://www.springframework.org/schema/security
http://www.springframework.org/schema/security/spring-security-3.2.xsd
	http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

	<!-- enable code below if we want to protect methods with @PreFilter,
@PreAuthorize,
		@PostFilter, @PostAuthorize, commented spring root and expression
handler override -->
	<global-method-security pre-post-annotations="enabled">
	</global-method-security>

	<http pattern="/resources/**" security="none" />
	<http pattern="/security/**" security="none" />

	<!-- The URL rewriting can be prevented by adding the following conf to http
element -->
	<!-- disable-url-rewriting="true" -->
	<!-- To use Spring SpEL include use-expressions="true" -->
	<http auto-config="true" use-expressions="true">

		<!-- This is for concurrency management -->
		<!-- If the second authentication takes place through another non-
interactive mechanism,
		such as “remember-me”, an “unauthorized” (402) error will be sent to the
client. If instead
		you want to use an error page, you can add the attribute session-
authentication-error-url to
		the session-management element. -->

		<!-- session-management configuration attribute invalid-session-
url="/invalidSession" causes a problem during
		logout as the session gets invalidated and upon hitting the logout-
success-url it goes to
		invalid-session-url -->

		<session-management session-authentication-error-
url="/sessionAuthError">
			<!-- By specifying error-if-maximum-exceeded="true", the second
login will fail
			and the authentication-failure-url will be called.
			If error-if-maximum-exceeded="true" is not specified, on a
successful second login session
			it will invalidate the first session. So trying the first session
will call invalid-session-url  -->
		<concurrency-control max-sessions="1" error-if-maximum-exceeded="true"
			expired-url="/sessionExpired" />
	</session-management>

	<intercept-url pattern="/" access="isAuthenticated()" />
```

```
	<intercept-url pattern="/login" access="permitAll" />
		<intercept-url pattern="/admin**" access="hasRole('ROLE_USER')" />

		<!-- authentication-failure-handler-ref = authFailHandlerBean
			authentication-success-handler-ref = authSuccHandlerBean -->

		<form-login
			login-processing-url="/check/security"
		    login-page="/login"
		    default-target-url="/"
			authentication-failure-url="/login?error"
			username-parameter="username"
			password-parameter="password" />

		<!-- success-handler-ref = logoutSuccessHandlerBean -->
		<logout logout-success-url="/login?logout" invalidate-session="true"
			logout-url="/logout" delete-cookies="JSESSIONID" />

	</http>

	<authentication-manager>
		<authentication-provider>
		  <user-service>
			<user name="pcpinnov" password="123456" authorities="ROLE_USER"
/>
		  </user-service>
		</authentication-provider>
	</authentication-manager>

</beans:beans>
```

security-context.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/security"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:beans="http://www.springframework.org/schema/beans"
	xsi:schemaLocation="
	http://www.springframework.org/schema/security
http://www.springframework.org/schema/security/spring-security-3.2.xsd
	http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

	<!-- enable code below if we want to protect methods with @PreFilter,
@PreAuthorize,
		@PostFilter, @PostAuthorize, commented spring root and expression
handler
		override -->
	<global-method-security pre-post-annotations="enabled">
	</global-method-security>

	<http pattern="/resources/**" security="none" />
	<http pattern="/security/**" security="none" />

	<!-- The URL rewriting can be prevented by adding the following conf to
```

```
            http element -->
      <!-- disable-url-rewriting="true" -->
      <!-- To use Spring SpEL include use-expressions="true" -->
      <http auto-config="true" use-expressions="true">

            <!-- This is for concurrency management -->
            <!-- If the second authentication takes place through another non-
interactive
                  mechanism, such as “remember-me”, an “unauthorized” (402) error
will be sent
                  to the client. If instead you want to use an error page, you can
add the
                  attribute session-authentication-error-url to the session-
management element. -->

            <!-- session-management configuration attribute invalid-session-
url="/invalidSession"
                  causes a problem during logout as the session gets invalidated
and upon hitting
                  the logout-success-url it goes to invalid-session-url -->

            <session-management
                  session-authentication-error-url="/sessionAuthError">
                  <!-- By specifying error-if-maximum-exceeded="true", the second
login
                        will fail and the authentication-failure-url will be
called. If error-if-maximum-exceeded="true"
                        is not specified, on a successful second login session it
will invalidate
                        the first session. So trying the first session will call
invalid-session-url -->
                  <concurrency-control max-sessions="1"
                        error-if-maximum-exceeded="true" expired-
url="/sessionExpired" />
            </session-management>

            <!-- <intercept-url pattern="/" access="isAuthenticated()" /> -->
            <intercept-url pattern="/" access="hasRole('ROLE_USER')" />
            <intercept-url pattern="/login" access="permitAll" />
            <intercept-url pattern="/admin**" access="hasRole('ROLE_USER')" />

            <!-- Handler for insufficient privilege -->
            <access-denied-handler error-page="/access/error" />

            <!-- authentication-failure-handler-ref = authFailHandlerBean
authentication-success-handler-ref
                  = authSuccHandlerBean -->

            <form-login login-processing-url="/check/security"
                  login-page="/login" authentication-failure-url="/login/error"
                  username-parameter="username" password-parameter="password" />

            <!-- success-handler-ref = logoutSuccessHandlerBean -->
            <!-- This will be handled through an Ajax logout request -->
```

```
            <!-- <logout logout-success-url="/login?logout" invalidate-
session="true"
                    logout-url="/logout" delete-cookies="JSESSIONID" /> -->

            <!-- Remember Me Token valid for 1 day -->
            <remember-me services-ref="rememberMeServices" />
      </http>

      <beans:bean id="jdbcTokenRepositoryImpl"

      class="org.springframework.security.web.authentication.rememberme.JdbcTokenRep
ositoryImpl">
            <beans:property name="dataSource" ref="dataSource"/>
      </beans:bean>

      <beans:bean id="rememberMeServices"
class="org.springframework.security.web.authentication.rememberme.PersistentTokenBase
dRememberMeServices">
          <beans:constructor-arg name="key" type="java.lang.String"
value="secAppKey"/>
          <beans:constructor-arg name="userDetailsService" ref="customUserService"/>
          <beans:constructor-arg name="tokenRepository"
ref="jdbcTokenRepositoryImpl"/>
          <beans:property name="tokenValiditySeconds" value="43200"/>
          <beans:property name="parameter" value="remember-me" />
          <beans:property name="cookieName" value="SECAPP_RM"/>
      </beans:bean>

      <beans:bean id="rememberMeFilter"
class="org.springframework.security.web.authentication.rememberme.RememberMeAuthentic
ationFilter">
            <beans:constructor-arg name="rememberMeServices"
ref="rememberMeServices"/>
            <beans:constructor-arg name="authenticationManager"
ref="authenticationManager" />
      </beans:bean>

      <beans:bean id="rememberMeAuthenticationProvider"
class="org.springframework.security.authentication.RememberMeAuthenticationProvider">
      <beans:constructor-arg name="key" value="secAppKey"/>
      </beans:bean>

    <authentication-manager alias="authenticationManager">
      <authentication-provider ref="daoAuthenticationProvider" />
      <authentication-provider ref="rememberMeAuthenticationProvider" />
      </authentication-manager>

      <beans:bean id="daoAuthenticationProvider"

      class="org.springframework.security.authentication.dao.DaoAuthenticationProvid
er">
            <beans:property name="userDetailsService" ref="customUserService" />
            <beans:property name="passwordEncoder" ref="encoder" />
      </beans:bean>
```

```
	<beans:bean id="encoder"
class="org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder">
		<beans:constructor-arg name="strength" value="10" />
	</beans:bean>

	<beans:bean id="customUserService"
		class="com.pcp.innovation.secapp.service.CustomUserService" />

</beans:beans>
```

The 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/jms-context.xml
             /WEB-INF/spring/mail-context.xml
            /WEB-INF/spring/persist-context.xml
            /WEB-INF/spring/security-context.xml
        </param-value>
  </context-param>

  <listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-
class>
  </listener>

  <servlet>
    <servlet-name>SecAppDispatcher</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>SecAppDispatcher</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/securedAppDS</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
```

```
    <res-auth>Container</res-auth>
  </resource-ref>

  <!-- Filters -->
  <!-- Spring Security Delegating Filter Proxy -->

  <filter>
    <filter-name>springSecurityFilterChain</filter-name>
    <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
  </filter>
  <filter-mapping>
    <filter-name>springSecurityFilterChain</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>

  <!-- This is for enabling the concurrent session-control support -->
  <listener>
    <listener-class>
      org.springframework.security.web.session.HttpSessionEventPublisher
    </listener-class>
  </listener>

  <!-- This is for setting timeout for all HTTP sessions and preventing URL rewriting
-->
  <session-config>
    <session-timeout>15</session-timeout>
    <tracking-mode>COOKIE</tracking-mode>
  </session-config>

  <!-- This is a custom filter for Setting No Cache for Login Page -->
  <filter>
    <filter-name>SetCacheControl</filter-name>
    <filter-class>com.pcp.innovation.secapp.web.security.CacheControlFilter</filter-
class>
  </filter>
  <filter-mapping>
    <filter-name>SetCacheControl</filter-name>
    <url-pattern>/login</url-pattern>
  </filter-mapping>

  <!-- 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>
```

The 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.pcp.innovation</groupId>
  <artifactId>securedapp</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>
  <properties>
       <spring.version>4.0.6.RELEASE</spring.version>
       <springsecurity.version>3.2.3.RELEASE</springsecurity.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>securedapp</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>

<!-- Spring security -->
<dependency>
	<groupId>org.springframework.security</groupId>
	<artifactId>spring-security-core</artifactId>
	<version>${springsecurity.version}</version>
</dependency>
<dependency>
	<groupId>org.springframework.security</groupId>
	<artifactId>spring-security-web</artifactId>
	<version>${springsecurity.version}</version>
</dependency>
<dependency>
	<groupId>org.springframework.security</groupId>
	<artifactId>spring-security-config</artifactId>
	<version>${springsecurity.version}</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-jms -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-jms</artifactId>
    <version>4.3.10.RELEASE</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.apache.activemq/activemq-all -->
<dependency>
    <groupId>org.apache.activemq</groupId>
    <artifactId>activemq-all</artifactId>
    <version>5.13.0</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>

<!-- Java Mail -->
<dependency>
           <groupId>javax.mail</groupId>
           <artifactId>mail</artifactId>
           <version>1.4.3</version>
    </dependency>
```

```
        <!-- Velocity Template -->
        <dependency>
                <groupId>org.apache.velocity</groupId>
                <artifactId>velocity</artifactId>
                <version>1.7</version>
        </dependency>

        <!-- Spring Integration Core -->
        <dependency>
                <groupId>org.springframework.integration</groupId>
                <artifactId>spring-integration-mail</artifactId>
                <version>4.0.1.RELEASE</version>
        </dependency>

        <!-- Jasypt for encrypted properties -->
        <dependency>
                <groupId>org.jasypt</groupId>
                <artifactId>jasypt</artifactId>
                <version>1.9.2</version>
                <scope>compile</scope>
        </dependency>
        <dependency>
                <groupId>org.jasypt</groupId>
                <artifactId>jasypt-spring31</artifactId>
                <version>1.9.2</version>
                <scope>compile</scope>
        </dependency>

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

Tha Java Layer

SecureAppController.java

```
package com.pcp.innovation.secapp.controller;

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

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import javax.servlet.http.Cookie;

import org.apache.log4j.Logger;
```

```java
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.http.HttpStatus;
import org.springframework.security.authentication.AnonymousAuthenticationToken;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.web.authentication.RememberMeServices;
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.bind.annotation.ResponseStatus;
import org.springframework.web.multipart.MultipartFile;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.pcp.innovation.secapp.dao.SecureAppDao;
import com.pcp.innovation.secapp.exception.AppException;
import com.pcp.innovation.secapp.exception.DBException;
import com.pcp.innovation.secapp.exception.ExceptionConstants;
import com.pcp.innovation.secapp.exception.RestExceptionHandler;
import com.pcp.innovation.secapp.model.Feedback;
import com.pcp.innovation.secapp.model.RestResponse;
import com.pcp.innovation.secapp.model.User;
import com.pcp.innovation.secapp.model.UserRole;
import com.pcp.innovation.secapp.service.UserCreationService;

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

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

	@Autowired
	SecureAppDao secureAppDao;

	@Autowired
	RememberMeServices rememberMeServices;

	@Autowired
	UserCreationService userCreationService;

	@Autowired
	RestExceptionHandler restExceptionHandler;

	//If we want to handle the Login Page from the Index Page, we need to give 2
request mapping for
```

```
	//the below handler method such as "/" and "/login". After rendering the Index
page with Header
	//and bare minimum details, we need to fire an Ajax call to see if the
Security Session has been
	//established or not. Based on the result of this Ajax, we need to either
render the Login page
	//or the complete index page.

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

		return "/view/index.html";

	}

	@RequestMapping(value= "/login", method = RequestMethod.GET)
	public String dislayLogin(Model model, HttpServletRequest request){

		Authentication auth =
SecurityContextHolder.getContext().getAuthentication();
		if (!(auth instanceof AnonymousAuthenticationToken)) {
		    /* The user is logged in :) */
		    return "redirect:/";
		}

		return "/view/login.html";

	}

	@ResponseStatus(value=HttpStatus.FORBIDDEN)
	@RequestMapping(value= "/access/error", method = RequestMethod.GET)
	public @ResponseBody RestResponse handleAccessError(Model model,
HttpServletRequest request){

		RestResponse restResp = new RestResponse();
		restResp.setMessage(ExceptionConstants.ACCESS_DENIED);
		restResp.setSeverity(ExceptionConstants.HIGH);

	restResp.setMsgDesc(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.ACCE
SS_DENIED));

		restResp.setPayload(null);
		return restResp;

	}

	@ResponseStatus(value=HttpStatus.UNAUTHORIZED)
	@RequestMapping(value= "/login/error", method = RequestMethod.GET)
	public @ResponseBody RestResponse errorLogin(Model model, HttpServletRequest
request){

		RestResponse restResp = new RestResponse();
		restResp.setMessage(ExceptionConstants.INVALID_CRED);
		restResp.setSeverity(ExceptionConstants.HIGH);
```

```
	restResp.setMsgDesc(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.INVALID_CRED));

			restResp.setPayload(null);
			return restResp;

	}

	@RequestMapping(value = "/feedback", method = RequestMethod.POST)
	public @ResponseBody RestResponse createFeedback(@RequestBody Feedback feedback,
			@RequestParam(value="action") String action,
			HttpServletRequest request,
			RedirectAttributes redirectAttributes) throws DBException, AppException {

			try {
				RestResponse restResp = new RestResponse();
				Integer insertCount = 0;
				Integer newFeedbackId = 0;
				if (action.equals("create")) {
					Integer currentFeedbackId = secureAppDao.getCurrFeedbackId();
					if (currentFeedbackId == null) {
						currentFeedbackId = 0;
					}
					newFeedbackId = currentFeedbackId + 1;
					feedback.setFeedbackId(newFeedbackId);
					insertCount = secureAppDao.insFeedback(feedback);
					if (insertCount == 1) {
						LOGGER.info("Feedback Inserted Successfully");
					} else {
						restResp.setMessage(ExceptionConstants.TXN_FAILED);
						restResp.setSeverity(ExceptionConstants.MEDIUM);

	restResp.setMsgDesc(ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.TXN_FAILED));
					}
				}
				restResp.setPayload(newFeedbackId);
				return restResp;

			} catch (DBException e) {
				throw new DBException("FEEDBACK " + ExceptionConstants.INSERT_FAILED, e);
			}

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

	@RequestMapping(value = "/registration", method = RequestMethod.POST)
	public @ResponseBody RestResponse registerUser(@RequestBody User user,
```

```
                    @RequestParam(value="action") String action,
                    HttpServletRequest request,
                    RedirectAttributes redirectAttributes) throws DBException,
AppException {

             try {
                    RestResponse restResp = new RestResponse();
                    int insertCount = 0;

                    if (action.equals("create")) {

                           UserRole userRole = new UserRole();
                           userRole.setRoleName("ROLE_USER");
                           userRole.setUserName(user.getUserName());
                           insertCount =
userCreationService.createEncryptedUser(user, userRole);

                           if (insertCount == 2) {
                                  LOGGER.info("User Details inserted successfully");
                                  userCreationService.sendNotification(user);
                           } else {
                                  restResp.setMessage(ExceptionConstants.TXN_FAILED);
                                  restResp.setSeverity(ExceptionConstants.MEDIUM);

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

                    restResp.setPayload(insertCount);
                    return restResp;

             } catch (DBException e) {
                    throw new DBException("REGISTRATION " +
ExceptionConstants.INSERT_FAILED, e);
             }

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

      //This is for handling logout through an Ajax request
      //Also the Spring security logout configuration doesn't remove the JSESSIONID
cookie
      //because Tomcat doesn't go hand-in-hand. Hence this necessitates this handler
method.
      @RequestMapping(value = {"/logout"}, method = RequestMethod.POST)
      public @ResponseBody RestResponse logoutDo(HttpServletRequest request,
                    HttpServletResponse response) throws AppException {
             try {
                    HttpSession session= request.getSession(false);
                   SecurityContextHolder.clearContext();
              session= request.getSession(false);
              if(session != null) {
```

```
                    session.invalidate();
               }

               //Cancelling JSESSION ID cookie; this will still leave and expired
cookie in the browser
               //To completely remove the cookie, use the following technique with
remember-me cookie
               for(Cookie cookie : request.getCookies()) {
                    if (cookie.getName().equals("JSESSIONID")) {
                         cookie.setMaxAge(0);
                    }
                }

               //Cancelling remember-me cookie and removing it from the browser
               LOGGER.debug("Cancelling Remember Me cookie");
               Cookie cookie = new Cookie("SECAPP_RM", null);
               cookie.setMaxAge(0);
               cookie.setPath(getCookiePath(request));
               response.addCookie(cookie);

               RestResponse restResp = new RestResponse();
                    restResp.setMessage(ExceptionConstants.SUCCESSFUL);
                    restResp.setSeverity(ExceptionConstants.LOW);
                    restResp.setMsgDesc("LOGOUT" +
ExceptionConstants.MESSAGE_MAP.get(ExceptionConstants.SUCCESSFUL));

                    restResp.setPayload(null);
                    return restResp;
               }

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

            //return "forward:/login" will send a POST request within the server
            //But we need a GET request so we have to use redirect
          //This is handled in the UI side, so commenting the redirect
            //return "redirect:/login";
       }

       private String getCookiePath(HttpServletRequest request) {
         String contextPath = request.getContextPath();
         return contextPath.length() > 0 ? contextPath : "/";
     }

}
```

SecureAppDao.java

```
package com.pcp.innovation.secapp.dao;

import java.util.List;

import org.apache.ibatis.annotations.Param;
```

```
import com.pcp.innovation.secapp.exception.DBException;
import com.pcp.innovation.secapp.model.Feedback;
import com.pcp.innovation.secapp.model.User;
import com.pcp.innovation.secapp.model.UserRole;

public interface SecureAppDao {

	public Integer insFeedback(Feedback feedback) throws DBException;
	public Integer insUser(User user) throws DBException;
	public Integer insUserRole(UserRole userRole) throws DBException;
	public Integer getCurrFeedbackId() throws DBException;
	public User getUser(@Param("userName") String userName) throws DBException;
	public List<UserRole> getUserRoles(@Param("userName") String userName) throws
DBException;

}
```

SecureAppDaoImpl.java

```
package com.pcp.innovation.secapp.dao;

import java.util.List;

import org.apache.ibatis.annotations.Param;
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.pcp.innovation.secapp.exception.DBException;
import com.pcp.innovation.secapp.mappers.SecureAppMapper;
import com.pcp.innovation.secapp.model.Feedback;
import com.pcp.innovation.secapp.model.User;
import com.pcp.innovation.secapp.model.UserRole;

@Repository
public class SecureAppDaoImpl implements SecureAppDao {

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

	@Autowired
	SecureAppMapper securedAppMapper;

	@Override
	@Transactional
	public Integer insFeedback(Feedback feedback) throws DBException {
		try {
			Integer insertCount = 0;
			insertCount = securedAppMapper.createFeedback(feedback);
			return insertCount;
		} catch (Exception e) {
			throw new DBException(e);
		}
```

```
}

@Override
@Transactional
public Integer insUser(User user) throws DBException {
	try {
		Integer insertCount = 0;
		insertCount = securedAppMapper.createUser(user);
		return insertCount;
	} catch (Exception e) {
		throw new DBException(e);
	}
}

@Override
@Transactional
public Integer insUserRole(UserRole userRole) throws DBException {
	try {
		Integer insertCount = 0;
		insertCount = securedAppMapper.createUserRole(userRole);
		return insertCount;
	} catch (Exception e) {
		throw new DBException(e);
	}
}

@Override
public Integer getCurrFeedbackId() throws DBException {
	try {
		Integer currFeedbackId = securedAppMapper.getCurrentFeedbackId();
		return currFeedbackId;
	} catch (Exception e) {
		throw new DBException(e);
	}
}

@Override
public User getUser(String userName) throws DBException {
	try {
		User usrDtl = securedAppMapper.getUserByName(userName);
		return usrDtl;
	} catch (Exception e) {
		throw new DBException(e);
	}
}

@Override
public List<UserRole> getUserRoles(String userName) throws DBException {
	try {
		List<UserRole> usrRole = securedAppMapper.getUserRoles(userName);
		return usrRole;
	} catch (Exception e) {
		throw new DBException(e);
	}
}
```

```
}
```

AppException.java

```
package com.pcp.innovation.secapp.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.pcp.innovation.secapp.exception;
```

```java
/**
 * @author Prabhu Purohit
 *
 */
public class DBException extends RuntimeException {

	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

```java
package com.pcp.innovation.secapp.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 INVALID_CRED = "2002";
	public static final String ACCESS_DENIED = "2003";
	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(INVALID_CRED, "INVALID USER NAME OR PASSWORD");
		MESSAGE_MAP.put(ACCESS_DENIED, "YOU ARE NOT AUTHORIZED TO ACCESS");
		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.pcp.innovation.secapp.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.pcp.innovation.secapp.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;

	}
}
```

MessageReceiver.java

```
package com.pcp.innovation.secapp.jms;

import java.util.Date;
import java.util.HashMap;
import java.util.Map;
import java.util.Properties;

import javax.annotation.Resource;
import javax.jms.Message;
```

```
import javax.jms.MessageListener;
import javax.jms.TextMessage;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
import javax.mail.internet.MimeMessage.RecipientType;

import org.apache.log4j.Logger;
import org.apache.velocity.app.VelocityEngine;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.mail.javamail.JavaMailSender;
import org.springframework.mail.javamail.JavaMailSenderImpl;
import org.springframework.mail.javamail.MimeMessageHelper;
import org.springframework.ui.velocity.VelocityEngineUtils;

public class MessageReceiver implements MessageListener {

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

	@Autowired
	JavaMailSenderImpl javaMailSender;

	@Autowired
	private VelocityEngine velocityEngine;

	/*@Resource(name = "mailProperties")
	private Properties mailProperties;*/

	private static final String HTML_HEADERS = "text/html; charset=UTF-8";
    private static final String TEXT_HEADERS = "text/plain; charset=UTF-8";

    public void onMessage(final Message message) {

	try {

		if (message instanceof TextMessage) {
		   final TextMessage textMessage = (TextMessage) message;
		    LOGGER.info("The Message Received in the SecApp Queue: " +
textMessage);
		    //Create a MIME message using the mail sender implementation
		    MimeMessage mimeMessage = javaMailSender.createMimeMessage();

		    //Create a MIME message using Constructor
		    //MimeMessage mimeMessage1 = new
MimeMessage(javaMailSender.getSession());

		    //Set To and From addresses
		    final InternetAddress from = new
InternetAddress("prabhu.purohit@ge.com");
		    final InternetAddress[] replyTo = {new InternetAddress("no-
reply@ge.com")};
		    final InternetAddress[] sendTO = {new
InternetAddress("prabhu.purohit@ge.com")};
		    final InternetAddress[] sendCC = null;
		    final InternetAddress[] sendBCC = null;
		    final String subject = "This is a test subject";
```

```
                    //Build the body using Velocity Template
                    String body = "";
                    Map<String, Object> model = new HashMap<String, Object>();
                    model.put("userName", "Prabhu");
                    model.put("userEmailId", "prabhu.purohit@ge.com");

                    body = VelocityEngineUtils.mergeTemplateIntoString(velocityEngine,
                       "emailTemp/registerEmail.vm", "UTF-8", model);

                    //final String body = "<html><body>This is a test
body</body></html>";
                    final boolean isHTML = true;

                    // Or create the message using the Spring MIME message helper
template
                    /*MimeMessageHelper helper;
                    try {
                      helper = new MimeMessageHelper(mimeMessage, true, "UTF-8");
                    } catch (Exception ex) {
                      ex.printStackTrace();
                      throw new MailException("Not able to create a MIME message", ex);
                    }

                    helper.setFrom(from);
                    helper.addBcc(sendBCC);
                    helper.addTo(sendTO);
                    helper.setSubject(subject);
                    helper.setSentDate(new Date());
                    helper.setText(body, isHTML);*/

                    mimeMessage.setFrom(from);
                    mimeMessage.setReplyTo(replyTo);
                    mimeMessage.setRecipients(RecipientType.TO, sendTO);
                    mimeMessage.setRecipients(RecipientType.CC, sendCC);
                    mimeMessage.setRecipients(RecipientType.BCC, sendBCC);
                    mimeMessage.setSubject(subject);
                    mimeMessage.setContent(body, isHTML ? HTML_HEADERS : TEXT_HEADERS);
                    javaMailSender.send(mimeMessage);

                }

        } catch (Exception e) {
                //Need to check if we need to send any notification to the user
                //when the receive message fails
                LOGGER.error(e.getMessage(), e);
        }
    }
}
```

MessageSender.java

```
package com.pcp.innovation.secapp.jms;
```

```java
import java.util.Map;

import javax.jms.JMSException;
import javax.jms.Message;
import javax.jms.Session;
import javax.jms.TextMessage;

import org.apache.log4j.Logger;
import org.springframework.jms.core.JmsTemplate;
import org.springframework.jms.core.MessageCreator;

public class MessageSender {

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

	private final JmsTemplate jmsTemplate;

    public MessageSender(final JmsTemplate jmsTemplate) {
        this.jmsTemplate = jmsTemplate;
    }

    public void sendMessage(final String message) {
        jmsTemplate.send(new MessageCreator() {
             public Message createMessage(Session session) throws JMSException {
                   TextMessage tm = session.createTextMessage();
                   tm.setText(message);
                   return tm;
             }
        });
    }

    /*public void send(final Map map) {
        jmsTemplate.convertAndSend(map);
    }*/
}
```

SecAppMailPoller.java

```java
package com.pcp.innovation.secapp.jms;

import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.ApplicationContext;
import org.springframework.integration.channel.DirectChannel;
import org.springframework.messaging.Message;
import org.springframework.messaging.MessageHandler;
import org.springframework.messaging.MessagingException;

public class SecAppMailPoller {

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

	@Autowired
    private ApplicationContext appContext;
```

```
	public void initPoller() {
		LOGGER.info("Initiating the Message Poller");
		DirectChannel inputChannel = appContext.getBean("receivePop3Channel",
DirectChannel.class);
		inputChannel.subscribe(new MessageHandler() {
			@Override
			public void handleMessage(Message<?> message) throws
MessagingException {
				LOGGER.info("Message: " + message);
			}
		});
	}

}
```

SecureAppMapper.java

```
package com.pcp.innovation.secapp.mappers;

import java.util.List;

import org.springframework.stereotype.Repository;

import com.pcp.innovation.secapp.model.Feedback;
import com.pcp.innovation.secapp.model.User;
import com.pcp.innovation.secapp.model.UserRole;

//@Repository
public interface SecureAppMapper {

	public Integer getCurrentFeedbackId();
	public Integer createFeedback(Feedback feedback);
	public Integer createUser(User user);
	public Integer createUserRole(UserRole userRole);
	public User getUserByName(String userName);
	public List<UserRole> getUserRoles(String userName);

}
```

Feedback.java

```
package com.pcp.innovation.secapp.model;

public class Feedback {

	private Integer feedbackId;
	private String feedbackName;
	private String feedbackText;
	private String publishDate;
	public Integer getFeedbackId() {
		return feedbackId;
	}
	public void setFeedbackId(Integer feedbackId) {
```

```
		this.feedbackId = feedbackId;
	}
	public String getFeedbackName() {
		return feedbackName;
	}
	public void setFeedbackName(String feedbackName) {
		this.feedbackName = feedbackName;
	}
	public String getFeedbackText() {
		return feedbackText;
	}
	public void setFeedbackText(String feedbackText) {
		this.feedbackText = feedbackText;
	}
	public String getPublishDate() {
		return publishDate;
	}
	public void setPublishDate(String publishDate) {
		this.publishDate = publishDate;
	}
}
```

RestResponse.java

```
package com.pcp.innovation.secapp.model;

import com.pcp.innovation.secapp.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;
	}
}
```

SecAppUserDetails.java

```
package com.pcp.innovation.secapp.model;

import java.util.Collection;

import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.stereotype.Component;

public class SecAppUserDetails implements UserDetails {

	private static final long serialVersionUID = -1L;

	private User user;
	private Collection<? extends GrantedAuthority> authorities;

	public User getUser() {
		return user;
	}

	public void setUser(User user) {
		this.user = user;
	}

	public void setAuthorities(Collection<? extends GrantedAuthority> authorities)
{
		this.authorities = authorities;
	}

	@Override
```

```
	public Collection<? extends GrantedAuthority> getAuthorities() {
		return authorities;
	}

	@Override
	public String getPassword() {
		return user.getUserPass();
	}

	@Override
	public String getUsername() {
		return user.getUserName();
	}

	@Override
	public boolean isAccountNonExpired() {
		return isEnabled();
	}

	@Override
	public boolean isAccountNonLocked() {
		return isEnabled();
		//return true;
	}

	@Override
	public boolean isCredentialsNonExpired() {
		return isEnabled();
	}

	@Override
	public boolean isEnabled() {
		if (user.getIsActive().equals("Y")) {
			return true;
		} else {
			return false;
		}
	}

}
```

User.java

```
package com.pcp.innovation.secapp.model;

public class User {

	private String userName;
	private String userPass;
	private String secretQuest;
	private String secretAns;
	private String isActive;
	public String getUserName() {
		return userName;
```

```
	}
	public void setUserName(String userName) {
		this.userName = userName;
	}
	public String getUserPass() {
		return userPass;
	}
	public void setUserPass(String userPass) {
		this.userPass = userPass;
	}
	public String getSecretQuest() {
		return secretQuest;
	}
	public void setSecretQuest(String secretQuest) {
		this.secretQuest = secretQuest;
	}
	public String getSecretAns() {
		return secretAns;
	}
	public void setSecretAns(String secretAns) {
		this.secretAns = secretAns;
	}
	public String getIsActive() {
		return isActive;
	}
	public void setIsActive(String isActive) {
		this.isActive = isActive;
	}
}
```

UserRole.java

```
package com.pcp.innovation.secapp.model;

public class UserRole {

	private String roleName;
	private String userName;

	public String getRoleName() {
		return roleName;
	}

	public void setRoleName(String roleName) {
		this.roleName = roleName;
	}

	public String getUserName() {
		return userName;
	}

	public void setUserName(String userName) {
		this.userName = userName;
	}
```

```
}
```

CustomUserService.java

```
package com.pcp.innovation.secapp.service;

import java.util.ArrayList;
import java.util.Collection;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.security.core.authority.SimpleGrantedAuthority;

import com.pcp.innovation.secapp.dao.SecureAppDao;
import com.pcp.innovation.secapp.model.SecAppUserDetails;
import com.pcp.innovation.secapp.model.User;
import com.pcp.innovation.secapp.model.UserRole;

public class CustomUserService implements UserDetailsService {

	@Autowired
	SecureAppDao secureAppDao;

	/* (non-Javadoc)
	 * @see
org.springframework.security.core.userdetails.UserDetailsService#loadUserByUsername(j
ava.lang.String)
	 */
	public UserDetails loadUserByUsername(String username)
			throws UsernameNotFoundException {

		SecAppUserDetails secAppUserDtl = new SecAppUserDetails();
		Collection<SimpleGrantedAuthority> auths = new
ArrayList<SimpleGrantedAuthority>();

		try {
			User usr = secureAppDao.getUser(username);
			if (usr == null) {
				throw new UsernameNotFoundException("User Not Found: " +
username);
			} else {
				List<UserRole> usrRole =
secureAppDao.getUserRoles(username);
				for (UserRole role : usrRole) {
				    SimpleGrantedAuthority sga = new
SimpleGrantedAuthority(role.getRoleName());
				    auths.add(sga);
				}

				secAppUserDtl.setUser(usr);
				secAppUserDtl.setAuthorities(auths);
```

```
                        return secAppUserDtl;
                  }
            } catch (Exception e) {
                  throw new UsernameNotFoundException("User Not Found: " +
username, e);
            }

      }

}
```

UserCreationService.java

```
package com.pcp.innovation.secapp.service;

import com.pcp.innovation.secapp.exception.AppException;
import com.pcp.innovation.secapp.exception.DBException;
import com.pcp.innovation.secapp.model.User;
import com.pcp.innovation.secapp.model.UserRole;

public interface UserCreationService {

      public Integer createEncryptedUser(User user, UserRole userRole) throws
DBException, AppException;

      public void sendNotification(User user) throws AppException;

}
```

UserCreationServiceImpl.java

```
package com.pcp.innovation.secapp.service;

import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.pcp.innovation.secapp.controller.SecureAppController;
import com.pcp.innovation.secapp.dao.SecureAppDao;
import com.pcp.innovation.secapp.exception.AppException;
import com.pcp.innovation.secapp.exception.DBException;
import com.pcp.innovation.secapp.exception.ExceptionConstants;
import com.pcp.innovation.secapp.jms.MessageSender;
import com.pcp.innovation.secapp.model.User;
import com.pcp.innovation.secapp.model.UserRole;

@Service
public class UserCreationServiceImpl implements UserCreationService {
```

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

	@Autowired
	SecureAppDao secureAppDao;

	@Autowired
	MessageSender messageSender;

	//Need to check if the Transaction will role back as a whole if Checked
Exception occurs
	//Currently the DB Exception extends RUntimeException and hence is not a
checked exception
	//Also both the insert operation has @Transactional annotation in which case
by default
	//the transaction should propagate as a default behavior

	@Transactional
	public Integer createEncryptedUser(User user, UserRole userRole)
			throws DBException, AppException {

		try {

			int insertUserCount = 0;
			int insertUserRoleCount = 0;

			//Encrypt the Password and Secret Answer before saving in
Database
			BCryptPasswordEncoder passwordEncoder = new
BCryptPasswordEncoder();

			String hashedPassword =
passwordEncoder.encode(user.getUserPass());
			user.setUserPass(hashedPassword);

			String hashedSecAns =
passwordEncoder.encode(user.getSecretAns());
			user.setSecretAns(hashedSecAns);

			//Save the user details in the database
			insertUserCount = secureAppDao.insUser(user);
			if (insertUserCount == 1) {
				LOGGER.info("User Inserted Successfully");
				//Save the user roles in the database
				insertUserRoleCount = secureAppDao.insUserRole(userRole);
				if (insertUserRoleCount == 1) {
					LOGGER.info("User Role Inserted Successfully");

				} else {
					throw new DBException();
				}

			} else {
				throw new DBException();
			}
```

```
                        return (insertUserCount+insertUserRoleCount);

                } catch(DBException e) {
                        throw new DBException(e);
                } catch(Exception e) {
                        throw new AppException(e);
                }
        }

        public void sendNotification(User user) throws AppException {

                try {
                        messageSender.sendMessage("USER SUCCESSFULLY CREATED: " +
user.getUserName());
                }
                catch (Exception e) {
                        throw new AppException(e);
                }
        }

}
```

BooleanTypeHandler.java (Not Used)

```
package com.pcp.innovation.secapp.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.");
    }
  }

}
```

CacheControlFilter.java

```
package com.pcp.innovation.secapp.web.security;

import java.io.IOException;

import javax.servlet.Filter;
import javax.servlet.FilterChain;
import javax.servlet.FilterConfig;
import javax.servlet.ServletException;
import javax.servlet.ServletRequest;
import javax.servlet.ServletResponse;
import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.pcp.innovation.secapp.controller.SecureAppController;

/*This Filter is to avoid the Login page to be cached in the browser and show up
again even if the
user has successfully logged in earlier and not logged out yet. This will remove
cache control
for the url /login and force a server call*/

public class CacheControlFilter implements Filter {

      public static final Logger LOGGER =
Logger.getLogger(SecureAppController.class);
      public void doFilter(ServletRequest request, ServletResponse response,
FilterChain chain)
                  throws IOException, ServletException {

            LOGGER.error("Cache Filter Applied");
```

```
            HttpServletResponse resp = (HttpServletResponse) response;
            resp.setHeader("Cache-Control",
                    "no-store, no-cache, max-age=0, private, must-revalidate");
            chain.doFilter(request, response);
        }

    public void destroy() {}

    public void init(FilterConfig arg0) throws ServletException {}

}
```

SecAppUserDetailsService.java (Not Used)

```
package com.pcp.innovation.secapp.web.security;

public class SecAppUserDetailsService {

}
```

UserIdFilter.java (Not Used)

```
package com.pcp.innovation.secapp.web.security;

public class UserIdFilter {

}
```

The Resources

emailTemp/registerEmail.vm

```
<!DOCTYPE html>
<html>
<head>
    <title>Register Template</title>
</head>

<body>
    <h1 style='color: #ffffff; font-weight: normal; font-size: 18px; font-family: "GE
Inspira", "ge-inspira", "Helvetica Neue", Helvetica, Arial, sans-serif; background:
#333; text-align: left;margin-top:0px;margin-bottom: 0px;padding-bottom: 5px;padding-
top: 5px;padding-left: 10px'>
        <span>
            <img class="main-logo"
src="http:localhost:8080/securedapp/resources/app/img/pcpInnov.png" style="display:
inline-block;height: 40px;width: auto;margin-right: 5px;background:
transparent;margin-bottom: 0px;vertical-align: middle">
        </span>
        My Secured App
    </h1>
```

```
    <h4 style='color: #000; font-weight: bold; font-size: 16px; font-family: "GE
Inspira", "ge-inspira", "Helvetica Neue", Helvetica, Arial, sans-serif; background:
#99ccff; text-align: left;margin-top: 0px;margin-bottom: 0px;padding-bottom:
10px;padding-top: 10px;padding-left: 10px'>
        Registration Success!
    </h4>
    <div style='color: #000; font-weight: normal; font-size: 14px; font-family:
Arial, Helvetica, sans-serif; background: #eee; text-align: left;padding-left: 10px'>
        <p style='margin-top: 0px;padding-top: 20px'>Thank You for registering with
us. Please find below your registration details.</p>
        <br>
        <p style="font-weight: bold">Your User Name: $userName</p>
        <p style="font-weight: bold">Your Registered e-Mail: $userEmailId</p>
        <br>
        <p>You need to change your password for first time login.</p>
        <p>Follow the link below to login to the SecApp application.</p>
        <br>
        <p><a href="http://localhost:8080/securedapp/login">Login Now</a></p>
        <br>
    </div>
    <div style='color: #ffffff; font-weight: normal; font-size: 12px; font-family:
"GE Inspira", "ge-inspira", "Helvetica Neue", Helvetica, Arial, sans-serif;
background: #293d3d; text-align:center; ;padding-left: 10px'>
        <br><br><br><br>
        <u>THIS IS AN UNMONITORED MAILBOX, PLEASE DO NOT REPLY TO IT.</u>
        <br><br>
        <p>Copyright Protected by PCP Innovation</p>
        <br><br><br><br>
    </div>

</body>
</html>
```

jms/jms-local.properties

```
jms.broker.url=tcp://localhost:61616?daemon=true
jms.queue.name=SECAPP.LOCAL.MESSAGE.QUEUE
```

jms/mail-local.properties

```
smtp.host=mail.ad.ge.com
smtp.user=ENC(6i+PTK4Wb7DJcFszlE4DbNkXb/vgJ7rU)
smtp.password=ENC(ELU/rUp21aEuKE3/NRYEOa5lvyLLzcI6)

#pop3.store-uri=pop3://322004773:MyPassword@webmail.ge.com/INBOX
pop3.store-
uri=ENC(fQPqV8/6Dj13xIwGgz3S15P1eJiYwSWpPdInNsj151zw+QeAzcbtec5MJH9UGvrXKob+IipA2iw=)
```

mybatis/dataSource.properties

```
jdbc.driver=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/test
jdbc.username=root
jdbc.password=Pra@bhu01
```

mybatis/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.pcp.innovation.secapp.model.Feedback" alias="Feedback" />
    <typeAlias type="com.pcp.innovation.secapp.model.User" alias="User" />
    <typeAlias type="com.pcp.innovation.secapp.model.UserRole" alias="UserRole" />
  </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.pcp.innovation.secapp.typehandler.BooleanTypeHandler"
                 javaType="java.lang.Boolean" jdbcType="VARCHAR" />
  </typeHandlers>

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

</configuration>
```

mybatis/SecureAppMapper.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.pcp.innovation.secapp.mappers.SecureAppMapper">

      <resultMap id="UserResultMap" type="User" >
            <result column="USER_NAME" property="userName" jdbcType="VARCHAR" />
            <result column="USER_PASS" property="userPass" jdbcType="VARCHAR" />
            <result column="SECRET_QUEST" property="secretQuest" jdbcType="VARCHAR"
/>
            <result column="SECRET_ANS" property="secretAns" jdbcType="VARCHAR" />
            <result column="IS_ACTIVE" property="isActive" jdbcType="VARCHAR" />
      </resultMap>

      <resultMap id="UserRoleResultMap" type="UserRole" >
            <result column="USER_ROLE" property="userRole" jdbcType="VARCHAR" />
```

```
	</resultMap>

	<select id="getUserByName" parameterType="String" resultMap="UserResultMap" >
		SELECT
			USER_NAME,
			USER_PASS,
			SECRET_QUEST,
			SECRET_ANS,
			IS_ACTIVE
		FROM TEST.USER
		WHERE USER_NAME = #{userName, jdbcType=VARCHAR}
	</select>

	<select id="getUserRoles" parameterType="String" resultMap="UserRoleResultMap"
>
		SELECT
			ROLE_NAME
		FROM TEST.USER_ROLE
		WHERE USER_NAME = #{userName, jdbcType=VARCHAR}
	</select>

	<select id="getCurrentFeedbackId" resultType="Integer" >
		SELECT ID
		  FROM TEST.FEEDBACK A
		ORDER BY A.ID DESC
		LIMIT 1;
	</select>

	<!-- foreach construct can also be used for the column list and value list -->
	<insert id="createFeedback" parameterType="Feedback">
		INSERT INTO
			TEST.FEEDBACK

			(ID,
			<if test="feedbackName != null">
			NAME,
			</if>
			<if test="feedbackText != null">
			COMMENT,
			</if>
			PUBLISH_DATE
			)

			VALUES(
				#{feedbackId,jdbcType=NUMERIC},
			<if test="feedbackName != null">
				#{feedbackName,jdbcType=VARCHAR},
			</if>
			<if test="feedbackText != null">
				#{feedbackText,jdbcType=VARCHAR},
			</if>
				#{publishDate,jdbcType=DATE}
			)
	</insert>
```

```
	<insert id="createUser" parameterType="User">
		INSERT INTO
			TEST.USER

			(USER_NAME, USER_PASS, SECRET_QUEST, SECRET_ANS, IS_ACTIVE)

			VALUES(
				#{userName,jdbcType=VARCHAR},
				#{userPass,jdbcType=VARCHAR},
				#{secretQuest,jdbcType=VARCHAR},
				#{secretAns,jdbcType=VARCHAR},
				#{isActive,jdbcType=VARCHAR}
			)
	</insert>

	<insert id="createUserRole" parameterType="UserRole">
		INSERT INTO
			TEST.USER_ROLE

			(ROLE_NAME, USER_NAME)

			VALUES(
				#{roleName,jdbcType=VARCHAR},
				#{userName,jdbcType=VARCHAR}
			)
	</insert>

</mapper>
```

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.pcp.innovation">
        <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>
```

The Server

context.xml

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

MANIFEST.MF

```
Manifest-Version: 1.0
```