Credit Task 4.4 Web UI + AJAX

Design justification:

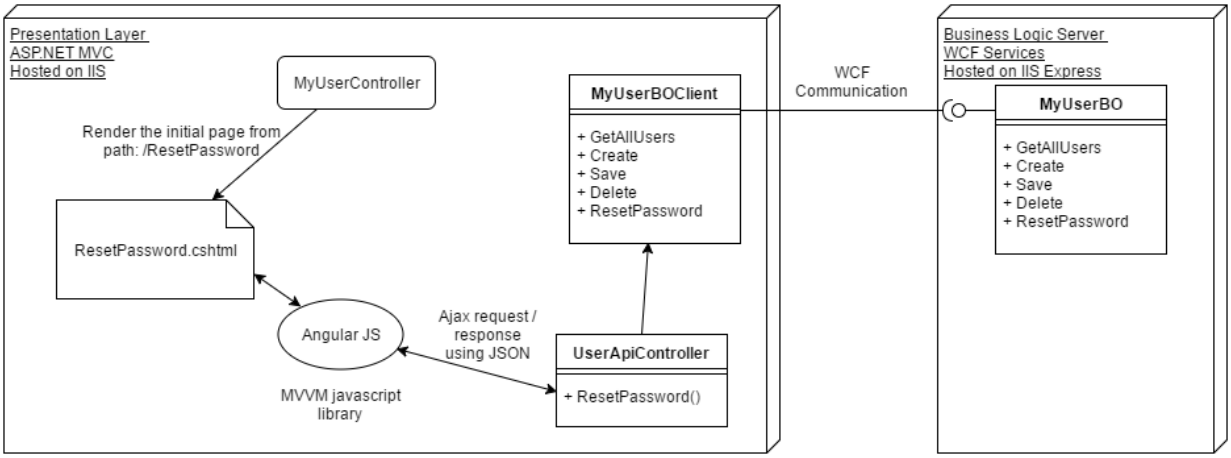


Figure 1. Architecture diagram.

Business Logic Server: This layer design is similar to the design on PassTask 4.2: The BLL is implemented using WCF Services and contains 1 service MyUserBO. This object use the IMyUserDAO interfaces provided by the DAL to interact with the persistence storage. The Business Logic Service is deployed to an IIS hosting service with a specific port number and URL address.

Presentation Layer: The presentation layer is coded using ASP.NET MVC 4 framework. It contains a proxy class to handle the communication with the equivalent business object in the Business Logic Server. Unlike the previous implementation that handle user request on the MyUserController, this proxy class is used inside the UserApiController to handle AJAX request. From the client site, the ResetPassword.cshtml page is rendered by the MyUserController with initial data and an AngularJS app attached. Once the page rendered completely, consequences interactions and requests from the client is handled in AJAX by AngularJS. AngularJS is a trending technology that implement MVVM design model that provides a superb performance and clear structure between HTML markup and data model, which effectively a JSON object. The model is defined as follow in javascript:

```
vm.model = {
    UserID: '@Model.UserID',
    Name: '@Model.Name',
    SecQn: '@Model.SecQn',
    SecAns: ''
};
```

The ajax request is effectively done by using the $http service from angular js core:

```
$http.post('/api/UserApi/ResetPassword', vm.model)
.then(function (success) {
    // navigate to success page
    window.location.href = '@Url.Action("ResetPasswordSuccess")';
}, function (error) {
    vm.errorMessage = error.data.Message;
});
```

The binding structure on html page is very simple:

```
    <div ng-controller="accCtrl as vm">
        <div class="form form-horizontal">
            <div class="form-group">
                <div class="input-group">
                    <span class="input-group-addon" ng-bind="vm.model.SecQn"></span>
                    <input type="text" class="form-control" ng-model="vm.model.SecAns" />
                </div>

                <div class="error" ng-bind="vm.errorMessage" ng-show="vm.errorMessage"></div>
            </div>
            <div class="form-group">
                <button class="btn btn-primary" ng-click="vm.resetPassword()">Reset
password</button>
                @Html.ActionLink("Back to user detail", "Edit", new { id = Model.UserID }, new {
@class = "btn btn-default" })
            </div>
        </div>
    </div>
```

By using AngularJS, it is far less javascript code than using other library such as Jquery and the code is much more readable from a maintenance perspective.

Test cases:

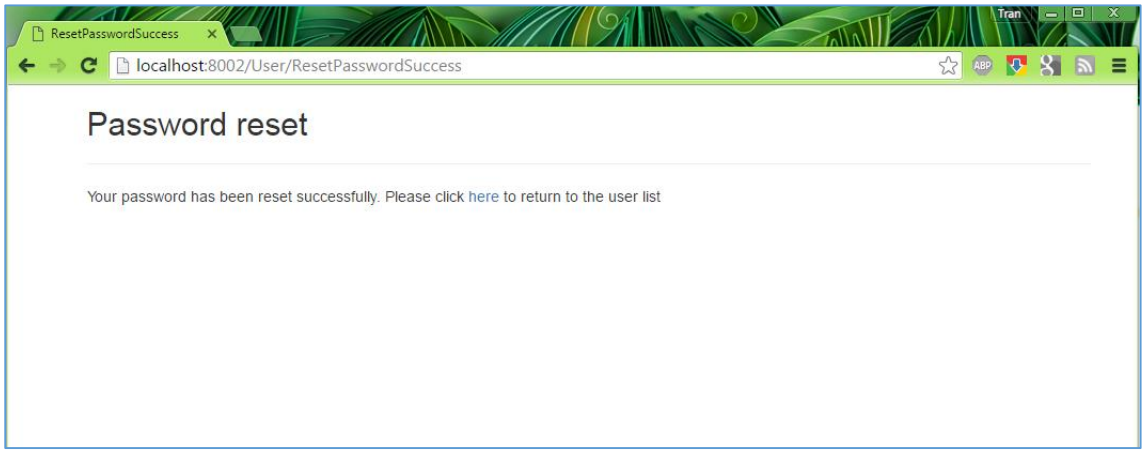
The following test cases have been taken out to make sure that the whole application works as expected:

ID	Test Case Name	Step	Expected Output	Result
01	Reset password success	- Click to the Edit link of one user in the list - Click on the “Reset Password” button - Enter the correct answer for security question	A success message will display The password is randomly modified A log file is record on the Business Logic Server	Pass
02	Reset password fail	- Click to the Edit link of one user in the list - Click on the “Reset Password” button - Enter the wrong answer for security question	**The page is not refreshed** and an error message is display.	Pass

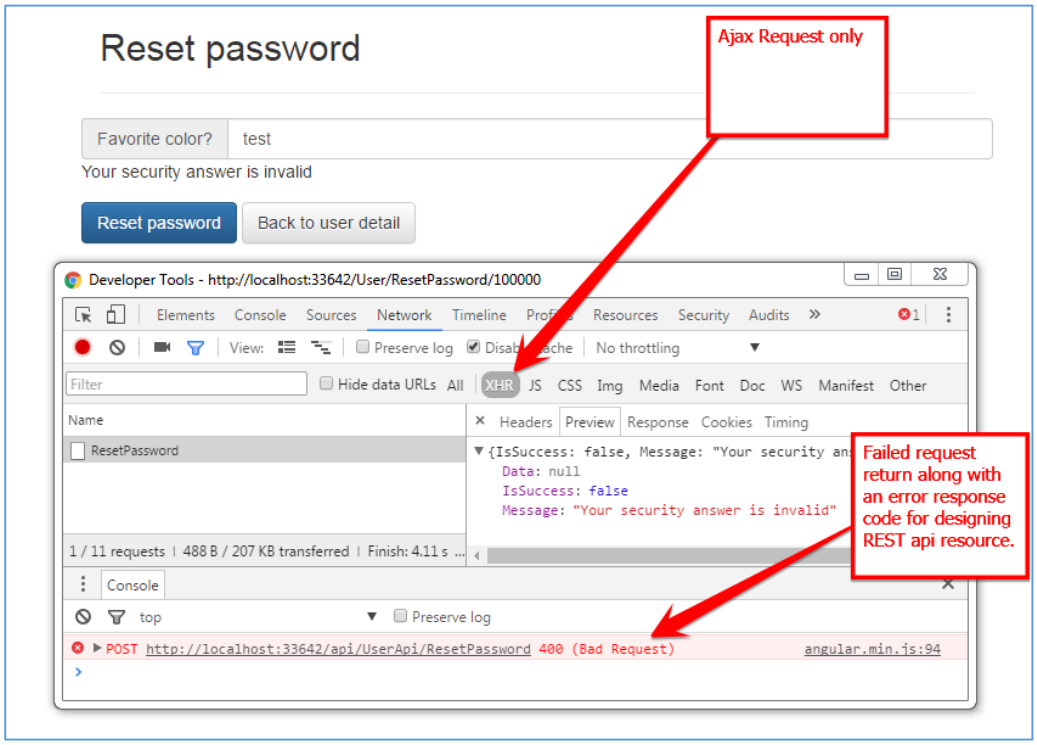
Table 1. Test cases and test results

Test case screen shots and success screen shots:

Test 01 – Reset password success



Test 02 – Reset password fail



References:

(1) 2016, Angularjs – Superheroic Javascript MVW framework, Google Inc, viewed 11 Apr 2016, <https://angularjs.org/>

(2) BitBucket source code: <https://bitbucket.org/werynguyen/swinschool/src/9bf01211be52511183e602bf4e40c5ada043f69b/?at=CT44>