Ads – SPA with AngularJS Lab

You are assigned to design and implement a web site for **Online Ads Publishing** as **single page application (SPA)** using **HTML5** and **AngularJS**. The app manages users and their ads organized by towns and categories. Anonymous site visitors can view published ads. Users can register, login, view their ads, add, edit and delete ads and logout. Administrators approve ads before publishing and can manage the users, ads, categories and towns. You are given the server-side REST services to be called by your app with AJAX requests so you do not need to develop a back-end.

All application screens are provided as UI prototype: Ads-Project-AngularJS-Screens.pdf.

The **goal of this lab** is to start the development of the Online Ads AngularJS application: create the project structure.

Problem 1. Create Ads AngularJS Project Structure

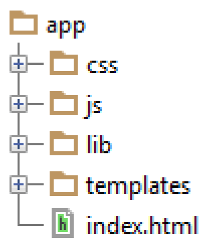
Create an AngularJS project structure for your application based on AngularJS, Bootstrap and jQuery.

Step 1. Unzip the provided project skeleton. You will get the following project assets:

- Libraries required for your project:
    - Angular libraries: **angular.js**, **angular-resource.js**, **angular-routes.js**
    - Bootstrap libraries: **bootstrap-3.1.1.js**.
    - jQuery: **jquery-2.1.3.js**.
    - jQuery Noty (a notification jQuery plug-in): **jquery.noty.js**.
- Some **CSS files** (the Bootstrap styles and theme + **app.css** – the main application CSS styles).
- A local Node.js based **Web server** for development purposes.
    - It can be started by the **start-web-server.bat** / **start-web-server.sh** shell scripts.
    - It will run at http://localhost:1234/.
- JetBrains **WebStorm project files** (if you want to use WebStorm as development environment).

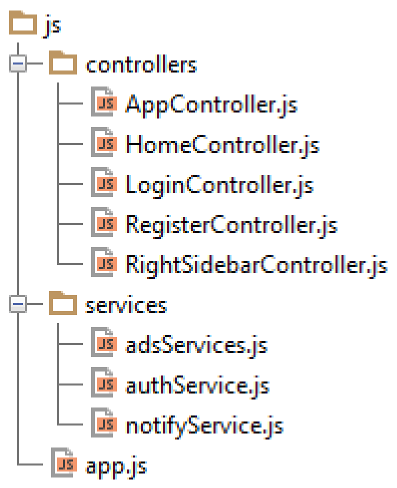
Step 2. Create the following **files and folders** (you already have some of them, so create the missing files only):

App Main Structure



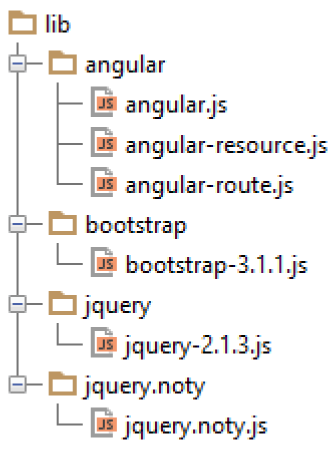
```
app
├─ css
├─ js
├─ lib
├─ templates
└─ index.html
```

JavaScript Files



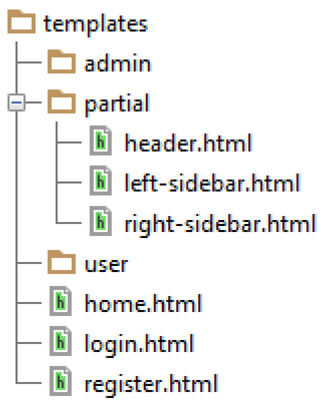
```
js
├─ controllers
│   ├─ AppController.js
│   ├─ HomeController.js
│   ├─ LoginController.js
│   ├─ RegisterController.js
│   └─ RightSidebarController.js
├─ services
│   ├─ adsServices.js
│   ├─ authService.js
│   └─ notifyService.js
└─ app.js
```

JS Libraries



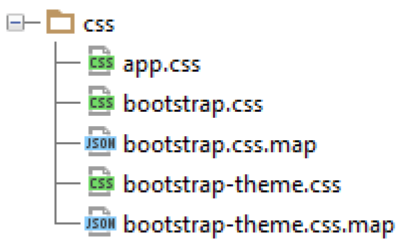
```
lib
├─ angular
│   ├─ angular.js
│   ├─ angular-resource.js
│   └─ angular-route.js
├─ bootstrap
│   └─ bootstrap-3.1.1.js
├─ jquery
│   └─ jquery-2.1.3.js
└─ jquery.noty
    └─ jquery.noty.js
```

HTML Templates



```
templates
├─ admin
├─ partial
│  ├─ header.html
│  ├─ left-sidebar.html
│  └─ right-sidebar.html
├─ user
├─ home.html
├─ login.html
└─ register.html
```

CSS Styles



```
css
├─ app.css
├─ bootstrap.css
├─ bootstrap.css.map
├─ bootstrap-theme.css
└─ bootstrap-theme.css.map
```

10 score

Step 3. Create the application main HTML file (**index.html**). It is the application entry point. Its goal it to define the application layout (header, main content with left and right sidebars and footer, Bootstrap-based), load CSS styles, and include the project JS libraries and the JS source code files: controllers, services, etc.

index.html

```html
<!DOCTYPE html>

<html ng-app="app" ng-controller="AppController">

<head>
    <!-- Load CSS files -->
    <link rel="stylesheet" href="css/bootstrap.css" />

<!--
    TODO: include bootstrap-theme.css
    TODO: include app.css
-->

    <meta charset="utf-8">
    <title>Ads</title>
</head>

<body>

<header id="app-header">
    <div ng-include="'templates/partial/header.html'"></div>
</header>

<div id='app-content'>
    <div class='col-md-2 sidebar'>
        <div ng-include="'templates/partial/left-sidebar.html'"></div>
    </div>

    <div class='col-md-8' ng-view>
    </div>

    <div class='col-md-2 sidebar' ng-controller="RightSidebarController">
        <div ng-include="'templates/partial/right-sidebar.html'"></div>
    </div>
```

```
</div>

<footer id="app-footer">
    &copy; 2014 by Software University Foundation, No Rights Reserved
</footer>

<!-- Load app libraries -->
<script src="lib/angular/angular.js"></script>

<!--
    TODO: include angular-route.js
    TODO: include angular-resource.js
    TODO: include jquery-2.1.3.js
    TODO: include jquery.noty.js
    TODO: include bootstrap-3.1.1.js
-->

<!-- Load app source code: main script, services, controllers, etc. -->
<script src="js/app.js"></script>

<script src="js/services/adsServices.js"></script>
<!--
    TODO: include the other services (authService.js and notifyService.js)
-->

<script src="js/controllers/AppController.js"></script>
<!--
    TODO: include the other controllers (HomeController, RightSidebarController,
LoginController, RegisterController)
-->

</body>

</html>
```

The **AppController** is the main controller for your application. It is attached to the app **<html>** element, so its logic is shared between all screens. It will hold common logic needed by the app header, app content and sidebars.

The header and sidebars are separated in different HTML templates to simplify the project maintenance.

The **RightSidebarController** will control the content shown in the right sidebar.

5 score

Step 4. Create the application main JS file (**app.js**). It defines your Angular application, a few constants and a few routes for the home, login and register screens:

app.js

```
'use strict';

var app = angular.module('app', ['ngRoute', 'ngResource']);

app.constant('baseServiceUrl', 'http://softuni-ads.azurewebsites.net');
app.constant('pageSize', 2);
app.config(function ($routeProvider) {

    $routeProvider.when('/', {
        templateUrl: 'templates/home.html',
        controller: 'HomeController'
    });

    $routeProvider.when('/login', {
        templateUrl: 'templates/login.html',
        controller: 'LoginController'
    });

    // TODO: define a route for the register controller

    $routeProvider.otherwise(
        { redirectTo: '/' }
    );

});
```

5 score

Step 5. Define and **AppController** skeleton that makes the authorization service available in the entire application (it is injected in the **$scope**, which is inherited in all elements inside the **<html>** element):

AppController.js

```
'use strict';

// The AppController holds the presentation logic for the entire app (common for all
screens)
app.controller('AppController',
    function ($scope, authService) {
      // Put the authService in the $scope to make it accessible from all screens
      $scope.authService = authService;
    }
);
```

5 score

Step 6. Define a **HomeController** skeleton that will hold the presentation logic for the home screen:

HomeController.js

```
'use strict';

// The HomeController holds the presentation logic for the home screen
app.controller('HomeController',
    function ($scope, $rootScope, adsService, notifyService, pageSize) {
      // TODO
    }
);
```

5 score

Step 7. Define a **LoginController** skeleton that will hold the presentation logic for the login screen:

LoginController.js

```
'use strict';

// The LoginController is responsible for the "Login" screen
app.controller('LoginController',
    function ($scope, $location, authService, notifyService) {
        // TODO
    }
);
```

5 score

Step 8. Define a **RegisterController** skeleton that will hold the presentation logic for the user registration screen. It should be very similar to the **LoginController**.

10 score

Step 9. Define a **RightSidebarController** skeleton that will hold the presentation logic for the right sidebar:

RightSidebarController.js

```
'use strict';

// The RightSidebarController controls the content displayed in the right sidebar
app.controller('RightSidebarController',
    function ($scope, categoriesService, townsService) {
      // TODO
    }
);
```

5 score

Step 10. Define **adsService**, **townsService** and **categoriesService** skeletons that will hold the business logic for loading ads, towns and categories:

adsServices.js

```
'use strict';

app.factory('adsService',
   function ($resource, baseServiceUrl) {
      return {
         // TODO: implement a service to get ads
      };
    }
);

app.factory('townsService',
   function ($resource, baseServiceUrl) {
      return {
         // TODO: implement a service to get towns
      };
    }
);

app.factory('categoriesService',
   function ($resource, baseServiceUrl) {
      return {
         // TODO: implement a service to get categories
      };
    }
);
```

5 score

Step 11. Define **authService** skeleton that will hold the business logic for user login, registration, logout, will hold the currently logged-in user, whether it is normal user or administrator and other user session related functionality:

authService.js

```
'use strict';

app.factory('authService',
    function ($http, baseServiceUrl) {
        return {
            login: function(userData, success, error) {
                // TODO
            },

            register: function(userData, success, error) {
                // TODO
            },

            logout: function() {
                // TODO
            },

            getCurrentUser : function() {
                // TODO
            },

            isAnonymous : function() {
                // TODO
            },

            isLoggedIn : function() {
                // TODO
            },

            isNormalUser : function() {
                // TODO
            },

            isAdmin : function() {
                // TODO
            },

            getAuthHeaders : function() {
                // TODO
            }
        }
    }
);
```

5 score

Step 12. Define **notifyService** that will be used to display info and error messages. Typically, when the user performs some action, it will either succeed or fail. In both cases, we need some kind of notifications. We will use the jQuery Noty to display info / error messages. We will add some logic to display correctly the error messages returned from the Ads REST Services:

notifyService.js

```
'use strict';

app.factory('notifyService',
    function () {
        return {
            showInfo: function(msg) {
                noty({
                    text: msg,
                    type: 'info',
                    layout: 'topCenter',
                    timeout: 1000}
                );
            },
            showError: function(msg, serverError) {
                // Collect errors to display from the server response
                var errors = [];
                if (serverError && serverError.error_description) {
                    errors.push(serverError.error_description);
                }
                if (serverError && serverError.modelState) {
                    var modelStateErrors = serverError.modelState;
                    for (var propertyName in modelStateErrors) {
                        var errorMessages = modelStateErrors[propertyName];
                        var trimmedName =
                            propertyName.substr(propertyName.indexOf('.') + 1);
                        for (var i = 0; i < errorMessages.length; i++) {
                            var currentError = errorMessages[i];
                            errors.push(trimmedName + ' - ' + currentError);
                        }
                    }
                }
                if (errors.length > 0) {
                    msg = msg + ":<br>" + errors.join("<br>");
                }
                noty({
                    text: msg,
                    type: 'error',
                    layout: 'topCenter',
                    timeout: 5000}
                );
            }
        }
    }
);
```

5 score

Step 13. Define a **home.html** template skeleton for your home screen:

home.html

```html
<h2>Home Screen</h2>
TODO
```

5 score

Step 14. Define a **left-sidebar.html** template skeleton for your left sidebar that will hold different navigation menus depending on the current user (anonymous site visitor / normal user / administrator):

left-sidebar.html

```html
<!--

TODO: implement logic to show different navigation menu for:
  - anonymous site visitors
  - logged in users
  - logged in administrators

-->

<!-- Navigation box for anonymous site visitors -->
<div class="box">
  <h2>Navigation</h2>
  <ul class="sidebar-menu">
      <li><a href="#/">Home</a></li>
      <li><a href="#/login">Login</a></li>
      <li><a href="#/register">Register</a></li>
  </ul>
</div>
```

5 score

Step 15. Define a skeleton **login.html** for your user login screen:

login.html

```html
<div class="box">
  <h2>Login</h2>
  TODO
</div>
```

3 score

Step 16. Define skeleton **register.html** for your user registration screen (just like the login screen)

3 score

Step 17. Define a skeleton **header.html** for your site header. It will display different content depending on the current user (anonymous site visitor / normal user / administrator):

header.html

```
Ads

<!--

TODO:

1) Implement logic to show different app header for:
    - anonymous site visitors
    - logged in users
    - logged in administrators

2) Implement logic to show different title for different screens

-->
```

3 score

Step 18. Define a skeleton **right-sidebar.html** for your right sidebar. It will hold the filter by category and town for the screens that need it:

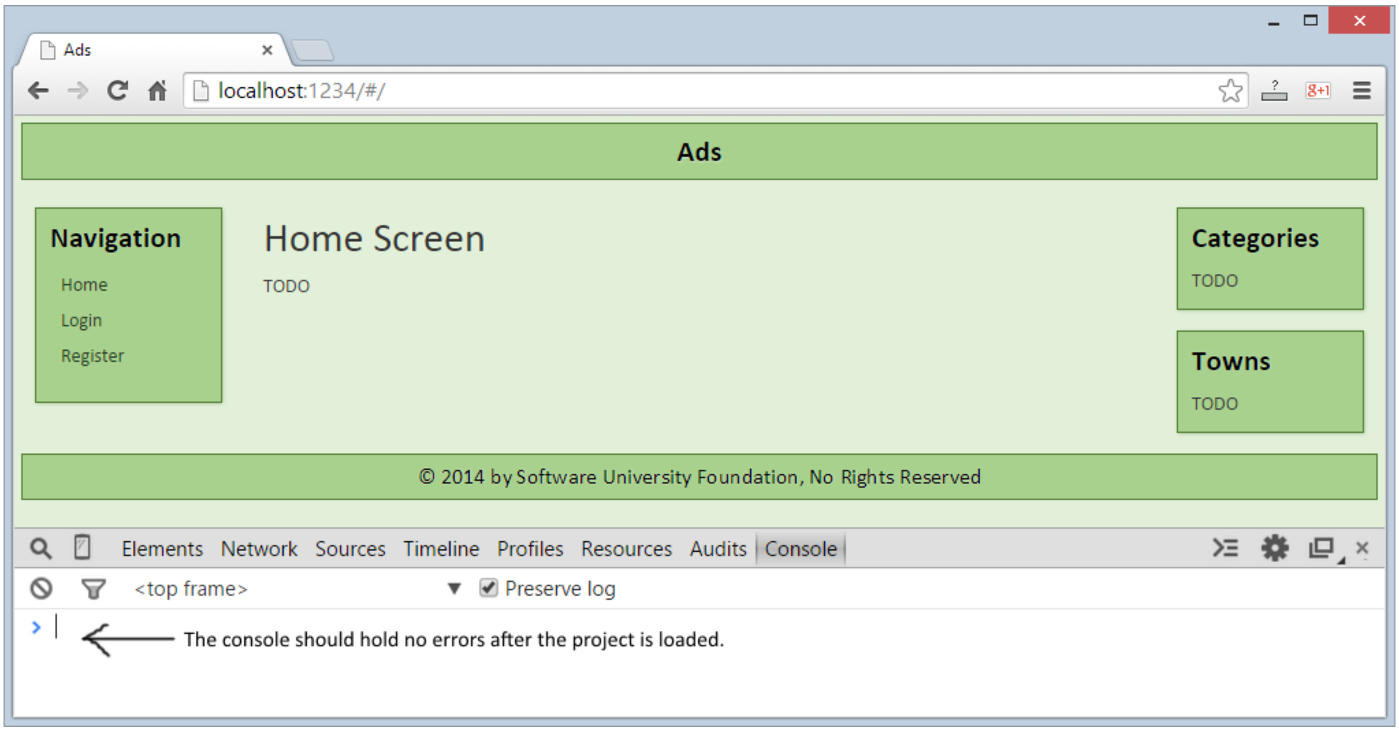
right-sidebar.html

```
<div class="box">
  <h2>Categories</h2>
 TODO
</div>

<div class="box">
  <h2>Towns</h2>
 TODO
</div>
```

3 score

Step 19. Start and test your application. Fix all bugs. Check the development console. It should be error free:


Ads
localhost:1234/#/
Ads
Navigation
Home
Login
Register
Home Screen
TODO
Categories
TODO
Towns
TODO
© 2014 by Software University Foundation, No Rights Reserved
Elements Network Sources Timeline Profiles Resources Audits Console
<top frame>
Preserve log

The console should hold no errors after the project is loaded.

10 score