**App.js**

'use strict';

angular.module('openWeatherApp', [

'ngRoute',

'openWeatherApp.filters',

'openWeatherApp.services',

'openWeatherApp.directives',

'openWeatherApp.controllers',

"iso-3166-country-codes"

]).

config(['$routeProvider', function($routeProvider) {

$routeProvider.when('/forecast', {templateUrl: 'partials/forecast.html', controller: 'OpenWeatherCtrl'});

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

}]);

**Controller.js**

'use strict';

angular.module('openWeatherApp.controllers', [])

.controller('OpenWeatherCtrl',

['$scope','openWeatherMap','exampleLocations','stormLocations','ISO3166',

function($scope,openWeatherMap,exampleLocations,stormLocations,ISO3166) {

$scope.message = '';

$scope.hasState = '';

$scope.exampleLocations = exampleLocations;

$scope.stormLocations = stormLocations;

$scope.iconBaseUrl = 'https://www.metaweather.com/api/';

$scope.forecast = openWeatherMap.queryForecastDaily({

location: exampleLocations[ 0 ]

});

$scope.getForecastByLocation = function() {

if ($scope.location == '' || $scope.location == undefined) {

$scope.hasState = 'has-warning';

$scope.message = 'Please provide a location';

return;

}

$scope.hasState = 'has-success';

$scope.forecast = openWeatherMap.queryForecastDaily({

location: $scope.location

});

};

$scope.setLocation = function(loc) {

$scope.location = loc;

$scope.getForecastByLocation();

};

$scope.getIconImageUrl = function(iconName) {

return (iconName ? $scope.iconBaseUrl + iconName + '.png' : '');

};

}])

**Directive.js**

'use strict';

angular.module('openWeatherApp.directives', [])

.directive('appVersion', ['version', function(version) {

return function(scope, elm, attrs) {

elm.text(version);

};

}])

.directive('weatherPanel',[function factory() {

return {

restrict: 'EA',

scope: {

useDayForecast: '=showEntry',

forecast: '=weatherPanel'

},

templateUrl: 'partials/\_weather-panel-light.html',

link: function(scope, element, attrs) {

scope.getIconImageUrl = function(iconName) {

return (iconName ? 'https://www.metaweather.com/api//static/img/weather/png/' + iconName + '.png' : '');

};

scope.parseDate = function (time) {

return new Date(time \* 1000);

};

}

}

}])

.directive('weatherPanelWind',[function factory() {

return {

restrict: 'EA',

scope: {

useDayForecast: '=showEntry',

forecast: '=weatherPanel'

},

templateUrl: 'partials/\_weather-panel-wind.html',

link: function(scope, element, attrs) {

scope.getIconImageUrl = function(iconName) {

return (iconName ? 'https://www.metaweather.com/api/static/img/weather/png/' + iconName + '.png' : '');

};

scope.parseDate = function (time) {

return new Date(time \* 1000);

};

}

}

}]);

**Filter.js**

'use strict';

angular.module('openWeatherApp.filters', [])

.filter('interpolate', ['version', function(version) {

return function(text) {

return String(text).replace(/\%VERSION\%/mg, version);

}

}])

.filter('placeholder', [function() {

return function (input,phvalue) {

return (angular.isUndefined(input) || input == '') ? phvalue : input;

};

}])

**Service.js**

'use strict';

/\* Services \*/

angular.module('openWeatherApp.services', ['ngResource'])

.factory('openWeatherMap', function($resource) {

var apiKey = '279b4be6d54c8bf6ea9b12275a567156';

var apiBaseUrl = 'https://www.metaweather.com/api/';

return $resource(apiBaseUrl + ':path/:subPath?query=:location',

{

mode: 'json',

callback: 'JSON\_CALLBACK',

units: 'metric',

lang: 'en'

},

{

queryWeather: {

method: 'JSONP',

params: {

path: 'location',

subPath: 'search'

},

isArray: false,

headers: {

'x-api-key': apiKey

}

},

queryForecast: {

method: 'JSONP',

params: {

path: 'forecast'

},

isArray: false,

headers: {

'x-api-key': apiKey

}

}/\*,

queryForecastDaily: {

method: 'JSONP',

params: {

path: 'forecast',

subPath: 'daily',

cnt: 7

},

isArray: false,

headers: {

'x-api-key': apiKey

}

}\*/

}

)

});