Corvisa Weather Application

# Scope:

## Application Description

Using the free weather service ( <http://openweathermap.org/API> ) you will build a modern javascript application that allows end users to figure what the weather is doing today and what it might be doing during the week. You application should make it easy to get the weather anytime anywhere.

The application should work in all modern browsers: Chrome, Firefox, IE 9+

## End User Description

End users will be employees at Corvisa. We will set your application on central server and allow them to use it for a day. Most people in the office are on the move and tend to use their cell phone for everything.

## Application Requirements

Details for this application are defined below.

Host The code on a server like github or bitbucket with all of the source file required to run you application.

Include a readme file that outlines how to install, build, and run your application

##### Single Page Application

The application should consist of a single static html page and function entirely with out page reloads.

##### **Current Weather by location**

You application should give users a single view of the current weather at their current location by using the browsers Geo Location.

**Forecast by location**

Your application should provide users with a way to get a projected forecast of either 5 or 14 days ( based on settings ) for a location that they choose.

**Application Configuration / Settings**

You application should allow users to set a number of application level settings to provide a better user experience. Users should be able to set up multiple locations and select which location they would like the application to display. And easily switch between these configurations. At minimum you should allow users to set:

* Forecast Range – 5 or 14 days ( default 5 )
* Forecast Location
* Temperature Units ( F or C )

For example users should be able to define a configuration for Chicago with a 14 day forecast range in Fahrenheit and a configuration for Japan with a 5 day forecast displayed in Celsius.

### Code Expectations

##### **Well** **Structured**

Code should be well structured, maintainable, Object Orientated & extensible and follow best practices

##### **Well Documented**

Build your application with the assumption that other developers will be working on it in the future, so it should be documented enough to make it easy to pick up where you left off.

##### **Error Handling**

All application errors should be handled and display end user friendly messages

##### **Tested**

Your code should be well tested and have fair code coverage.

##### **Static**

Your application should consist entirely of static html css and javascript. The project may run on a web server, but not have any server side requirements such as php or python scripts, rails, etc.

##### **Cross Domain**

You may not use any sort of a proxy server or scripts to quiry the api. You must fetch all data via client side javascript. The API does not support cross domain origin sharing.

### Bonus Features

##### **Data Caching**

Multi day forecast data should be cached for the remainder of the day up to midnight of the current day. Any Cashed data should be loaded when the page loads for the first time

##### **Mobile Support**

Implement a user interface that is optimized for mobile devices. This interface should support iOS, or Android devices.

##### **Templating / Data Binding**

In order to maintain best practices by separating views from business logic implement the application using some kind of client side templating engine. For super brownie points implement one way data binding in that changing the under lying data updates the associated view.

##### **MVC/Deep Linking**

Implement client side URL routing and history management. Urls should connect to and internal controller function that renders / updates a view. The back and forward buttons on the browser should allow the user to navigate with in the application. Navigating directly to the URL should populate the view directly.

##### **Iconography**

Create Icon to coincide with the weather conditions, rainy, cloudy, sunny, etc. You must create the graphics yourself. You must include the source files ( photoshop, illustrator ) with your project

##### **Themes**

What good is a web application if you can’t re-skin it? Implement theme support for your application. You must supply the default theme and an example / boilerplate theme for other developers to work on. You may use any CSS framework and/or pre processor you like. In other words – Make it Purty!

##### **Build**

Keeping with best practices, implement a build step the lints and optimizes your javascript / css and reports any errors during the process