Assignment 3: AJAX, JSON and Responsive Design

Weather Search

Grading Guidelines

Total Points: 15

Deduction of points in each category will NOT exceed the total points of the category.

- **Basic Search Form (3 points in total)**
  - Street, City and State input must be disabled and retain values when choosing "Current location" checkbox and they must be enabled when "Current Location" checkbox is disabled – 0.25 points
  - Disable the "Search" button when text fields are invalid - 0.25 points
  - Search button should be disabled until the user location is determined from IPinfo – 0.25 points
  - Validation for the search inputs and the error message(s), - 0.5 points total
  - "Clear" button resets the search form to the initial state and clears results area - 0.5 points
  - Obtain user location - 0.25 points
  - City Auto complete - 1 points
- **Results (7 points in total)**
  - **Daily tab (1 point in total)**
    - Show every data in the specified format - 0.25 points deduction for each missing data with a maximum of 1 point.
  - **Daily Temp Range tab (1 point in total)**
    - The chart should display the weather on a day basis, 0.5 points will be deducted.
    - The chart should have the corresponding label on the axis, 0.5 points will be deducted
  - **Meteogram tab (1 point in total)**
    - The chart should display the weather on an hourly basis, 0.5 points will be deducted
    - The chart should have the corresponding label on the axis, 0.5 points will be deducted.
  - **Details Pane (2 points in total)**
    - Show all the required fields in a table format – 0.1 points deduction for each missing data with a maximum of 0.5 points.
    - The rows in the table should follow an alternative dark and light background – 0.5 point
    - The google map should be interactive and have be centered on the pin drop – 1 point
  - **Misc (2 point in total)**
    - Functional Twitter button, should open a new tab with the required tweet text prefilled - 0.25 points
    - Navigation bar of tabs should be displayed in the correct format – 0.25 points
    - Title of the results section to be in the form "Forecast at <City>, <State>" - 0.5 points
    - Additional 1 point to be awarded in case both charts are displayed and are interactive – this means that the student was able to use the highcharts angular package. – 1 point

- **Favorites (1.5 point in total)**
  - Add city to favorite list – 0.25 points
  - Delete city from favorite list - 0.25 points
  - Delete city from search result if in favorite list - 0.25 points
  - Favorites should work in the mobile view – 0.25 points
  - Favorites list should update dynamically when removing a city – 0.25 points
  - Favorites should be retained when changing browser (since stored in MongoDB Atlas) – 0.25 points
- **Progress bars, Error messages (0.5 points in total)**
  - Progress bars on loading – 0.25 points
  - Error messages – 0.25 points
- **Animation (1 point in total)**
  - When moving between the results pane and the details pane the app should perform a "slide" animation in the correct direction – 1 point
- **Responsive (2 point in total)**
  - In mobile browsers, all of the pages should be the same as screenshots provided in description document - 1.5 points
  - In mobile browsers, all search, favorites, and Twitter functions must work - 0.5 points
- **Use of Bootstrap**
  - The app should be implemented using Bootstrap. Implementations without Bootstrap will result in a penalty of 2 points.
- **Use of GAE/AWS/Azure**
  - Node.js script must be deployed on GAE/AWS/Azure. To allow the graders to perform this test, an additional link should be added to the list of homework, with a format similar to the following:
    - http://xxx.appspot.com/[path]?[list_of_the_parameters_and_sample_values] OR
    - http://xxx.elasticbeanstalk.com/[path]?[list_of_the_parameters_and_sample_values] OR
    - http://xxx.azurewebsites.net/[path]?[list_of_the_parameters_and_sample_values]
  - This link would guide graders to a page where it displays the JSON data returned by your Node.js script running on the Google App Engine/Amazon AWS/Azure. Failing to provide a valid link will result in **a penalty of 4 points**.
- **Additional Requirements**
  - The entire assignment is to be implemented using Angular, Node.js and Bootstrap. Use of AngularJS (aka Angular 1.0) will result in **zero in the assignment.**
  - All tomorrow.io API requests must be made on **server side NodeJS back-end** (acting as a proxy). Any violation of this rule will result in a **4-point penalty**.
  - Both front end and backend code must be served by the same cloud service. GitHub Pages should not be used for the front end, resulting in a **3-point penalty** if used.
  - All other API calls should be made from client-side front-end.
  - The window should not reload for any kind of data request. All transactions are asynchronous. Any violation (using "synchronous calls) will result in a **2-point penalty**.
  - The program must work in Chrome and Firefox on both desktop, and mobile devices.
  - Graders will use Firefox RWD Tools menu to test for smartphone and tablet.

- Please note, you need to submit your files (HTML, JS, CSS, TS), both backend and front-end as a SINGLE ZIP file. See the instructions in the Assignment document describing which files should be submitted and in what folder format.