Refining and Understanding Weather prediction algorithms
  by way of GeoSpatial and Temporal Visualizations

Front End Functionality
  - Display weather patterns (predicted and real) over large areas of CONUS and
    across various time spans
  - Allow easy toggling (or concurrent viewing) between real and predicted values
  - Allow navigation between different predicted data sets. (where each set is generated
    by a particular algorithm)
  - Provide an overall rating for the accuracy of a given prediction set.

Required Backend Support
  - Must serve real and predicted weather data

Optional Backend Bonus
  - live generation of new predictions (it is acceptable for this to take up to 5 min)

Problem solved:
  - It is difficult to get a deep understanding of the success/faulure of ML algorithms,
    particularly in the context of large datasets like weather.

  [example:
    Given two algorithms for predicting precipitation, one is likely to be better on average
    than the other. However, it is quite possible that this relationship changes when we examine
    very constrained circumstances. For instance, an algorithm which is worse in general may
    excell at predicting course & timing of thunderstorms in the midwest. Discovery of nuances
    like this requires the ability to quickly evaluate many hypotheses. This is best facilitated
    by intuitive visual results and the ability to control what is being tested.
  ]

Open problems
  - Data scale. Weather tends to be big. Serving lots of weather data to the browser for rendering is
    not feasable. This can be resolved by having very granular weather data, serving weather data day
    by day, or rendering maps on the server then delivering as a slide deck

- Generating Live predictions. This would be a hugely awesome feature. But weather prediction
  algorithms tend to be computation and time intensive.

Required Technologies
- utilize d3 (or similar) for fast and dynamic heatmap generation
    Another option would be to serverside render map images, then forward these to the
    front as a slide deck
- use MySQL? Much of this is table type data. Ie. values over time for a large set of locations.
- use Java or similar for server app. We'll probably want to integrate R if we get choose to support
    live algorithm entry.

Optional Bonus
- Generate new predictions Live
  This is extremely valuable and very difficult.
- Combine this with a front end that allows for user selection of input & objective datasets, and
  user input or manipulation of algorithms and suddently the user can answer questions like the following:
  - how well would well known algorithm X have predicted the tornado in Oklahoma earlier this year?
  - does this algorithm perform better if I look only at weather data in the immediate area?
  - does is the algorithm I wrote better or worse than well known algorithm x?