**Next Steps**

As of June 25, 2023, new development on Frizz-Wiz will be suspended in favor of other projects. The latest files have been updated to the [web server](https://frizz-wiz.com/), Github ([private](https://github.com/matt0164/frizz)) and backed up [locally](file:///C:\Users\matta\PycharmProjects\frizz) with several prior versions. This version is labeled Alpha v0.02 and is essentially a proof of concept. This follows approximately one month of development and testing (and another 2-4 weeks of prior learning).

**Commercial Applications**

I envision marketing this web app to product companies that may want to white label a beta or future version. This could range from a low-cost/no-cost model to a more fully functional solution to support a marketing campaign where additional development is funded as part of a formal contract.

Rather than continue to iteratively improve Frizz, the decision was made to move on to other applications and keep the frizz-wiz.com domain active to market what I have thus far.

**Future Roadmap**

Hypothetically, I could continue development. Here are some ideas of future features that could be implemented:

* More sophisticated algorithms
  + Addition of wind and gust
  + Hourly point in time vs. 10AM-3PM average calculations for core day
  + Additional location capabilities
* User interface improvement
  + A table or graph displaying the forecasted humidity, dewpoint and wind over various points in time to show the user what data is used to develop the forecast and any future “point in time” anomalies (see Appendix)
  + Date and time input / selection (tried slider, something else?)
  + Natural language search capabilities (“How should I wear my hair tomorrow?”)
  + Rotation of photos most relevant to user/location or hair conditions
  + Branding and ads for products related to current weather conditions
  + Mapping capabilities (national, regional hair forecast map)
* Web / user support and attraction
  + Integration to other platforms
  + Social media integration

Appendix

I asked [ChatGPT-4](https://chat.openai.com/c/03bca053-df11-4201-a6be-6b47e3d55722) how to implement a table into the code at the bottom and this is what it said:

To display a small table at the bottom showing the humidity and dewpoint each hour, you will need to make some changes in both your Python Flask server code and your HTML. Here's an example of how to achieve this:

1. In your Flask route, when you're gathering the weather data, you'll also want to collect the hourly humidity and dewpoint data. You can create a list of dictionaries, where each dictionary represents one row of the table. For example:

**hourly\_data = []**

**for hour in weather\_data['hourly']:**

**hourly\_data.append({**

**'time': hour['time'],**

**'humidity': hour['humidity'],**

**'dewpoint': hour['dewpoint']**

**})**

Then include this data in your **jsonify** call:

**return jsonify({'result': weather\_data["result"], 'photo': weather\_data["photo"], 'hourly\_data': hourly\_data}), 200**

1. In your AJAX success callback, you can then loop through this hourly data and append it to a table in your HTML. First, add an empty table in your HTML where you want it to appear:

<**table id="hourly-table">**

**<thead>**

**<tr>**

**<th>Time</th>**

**<th>Humidity</th>**

**<th>Dew Point</th>**

**</tr>**

**</thead>**

**<tbody>**

**</tbody>**

**</table>**

1. Then, in your AJAX success callback, add the rows to this table:

**success: function(response) {**

**...**

**var hourlyTable = $('#hourly-table tbody');**

**hourlyTable.empty(); // Clear out the old rows**

**response.hourly\_data.forEach(function(hour) {**

**var row = '<tr><td>' + hour.time + '</td><td>' + hour.humidity + '</td><td>' + hour.dewpoint + '</td></tr>';**

**hourlyTable.append(row);**

**});**

**},**

This is just a simple example. Depending on the format of your weather data and how you want to display it, you may need to adjust it to suit your needs. Also, don't forget to style your table with CSS to make it look as you wish.