NDSU Microsoft Capstone 2020 - Power BI

**Index**

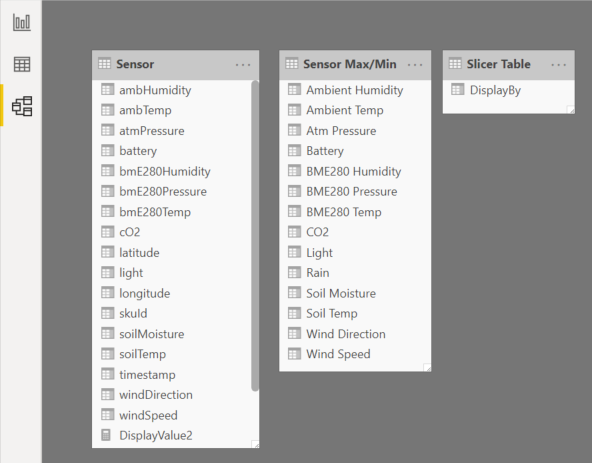
1. **Overview**
2. **Development**
3. **Hosting Reports**

**Overview**

A major portion of our capstone was dedicated to Power BI report development. Our goal was to provide useful visualizations of FarmBeats ground sensor data. Development of these reports was done completely on [Power BI Desktop](https://powerbi.microsoft.com/en-us/desktop/). Finished report designs were then pushed to a Power BI Pro account’s workspace. This can be done within Power BI Desktop.

**Development**

Fortunately, Power BI development is very straightforward. The desktop app is user-friendly, and data connections are easy to set up. Here is how we set up our reports:



**Sensor:** All of the actual sensor data comes from this table. This table is imported using the ‘Web’ option, and is directly connected to our Web API. Since our API was not hosted on Azure during development, we had to run a local copy of it through Visual Studio, then connect to it in Power BI while it was running to refresh our data. Once it is hosted on Azure, data refresh can be scheduled to happen automatically.

**Sensor Max/Min:** This is a manually created table that is ONLY used for defining the expected max/min values for the corresponding FarmBeats sensor measurements. These values are assigned to the maximum/minimum values on the gauge modules, and are used to provide a tighter range for the gauges to work with. For example: ambTemp’s minimum could be set to 32 in the Summer, since we know the temperature will not drop below freezing. These values can be changed at any time. If a new measurement is ever added to a sensor kit, their expected min/max values will have to be manually added to this table.

**Slicer Table:** This is another manually created table that only contains one column: DisplayBy. The DisplayBy column contains records for each FarmBeats sensor measurement. Whatever records are in this table will be directly listed in any slicer that uses DisplayBy. Right now, our reports use a slicer on the line chart and heatmap tabs. If a new measurement is ever added to a sensor kit, it will have to be manually added to this table.

A screenshot of a cell phone

Description automatically generated A screenshot of a cell phone

Description automatically generated

**Map View**

The map is powered by the default Map visualization, and uses the latitude/longitude fields from the Sensor table. We have settings for auto-zoom enabled, which means that it will fit all recognized sensors into view every time the report is refreshed. Make sure that the latitude/longitude fields are in the ‘Don’t summarize’ mode, otherwise location data will be inaccurate.

To achieve the ‘hover’ effect which shows a summary of all sensor measurements, we added every Sensor table field we wanted summarized into the Tooltips section of the map.

Since this is the default ‘home’ page of our report, the Cross-report option must be toggled off, since we don’t want any of the information shown to be filtered.

**Gauge View**

Each gauge is powered by a default gauge visualization, and uses a specific field from the Sensor table and its corresponding min/max fields from the Sensor Max/Min table. Within each gauge’s settings, make sure that all fields used are in ‘Average’ mode, otherwise data shown over time periods will be inaccurate.

Since this is a drill-through tab, cross-report is toggled on, and all fields we wanted to be drillable are also added to the Drillthrough section of the gauge (These can be assigned through any visual on the tab, as they are universal throughout all visuals on the same tab). Be aware that toggling cross-report on means that visuals will not show any data until it is drilled into from the Map View on a specific sensor.

The time selection ribbon along the bottom of the tab is powered by a free visual downloaded from the Marketplace called Timeline 2.1.1. It uses the timestamp value from the Sensor table. Make sure the value is in timestamp mode and not date hierarchy, and also toggle cross-report on.

The Sensor ID label is powered by a default card visualization, and uses the skuId field from the Sensor table. Make sure the value is in ‘Median mode’, otherwise it will inaccurately display the ID.

A back button is automatically added to a tab when it is switched to drill-through. It will bring the user back to the most previous tab, which in our case is the Map View.

**Line Chart View**

The line chart is powered by the default line chart visualization, and uses a custom measure field ‘DisplayValue2’ and the timestamp value from the Sensor table. A custom measure field was needed since what’s being displayed is chosen from our Slicer. Here is the code behind the measure:

A screenshot of text

Description automatically generated

Basically, whatever value is selected within the slicer is assigned to DisplayValue2 and shown on the line chart just like any other normal field. If a new measurement is ever added to a sensor kit, it will have to be manually added to this measure.

Just like the Gauge View, this is a drill-through tab, so cross-report must be toggled on, and all fields you want drillable must be added to the Drillthough section of the visuals.

There is also a direct copy of the Timeline 2.1.1 visual that is synced to the copies in the Gauge View (and the Heatmap View). Syncing can be chosen when initially copying a visual from one tab to another. If syncing is enabled, then values are synced across all instances of that visual. In our case, the time period selected is synced across all tabs which allows for data comparison.

There is also a Sensor ID label and a back button, both are identical to their Gauge View counterparts.

**Heatmap View**

The heatmap is powered by a free visual downloaded from the Marketplace simply called Heat map. It uses latitude, longitude, and the same custom measure field ‘DisplayValue2’ used in Line Chart View. It also uses a copy of the Slicer used in Line Chart View, Timeline 2.1.1, Sensor ID, and back button visuals.

Just like the previous two views, this is a drill-through tab, so cross-report must be toggled on, and all fields you want drillable must be added to the Drillthough section of the visuals.

**Hosting Reports**

Once our report development was finished, we published it to our Power BI Pro account’s workspace. This was done directly through Power BI Desktop. Publishing was needed in order for our web app to access/embed them.

Accessing your published reports can be done by singing into Power BI’s website and navigating to whatever workspace your published to. In order to embed them into a web app, you must click the ‘share’ option and copy/paste the link it gives you into your backend code when it is needed (See web app documentation).

Even after publishing and embedding a report, updates can easily be pushed to the workspace. As long as you publish your changes to your workspace, our web app will automatically use the latest version of the report.

Please reach out to me, Jack Morris, [jjackmorris04@gmail.com](mailto:jjackmorris04@gmail.com) with any questions!