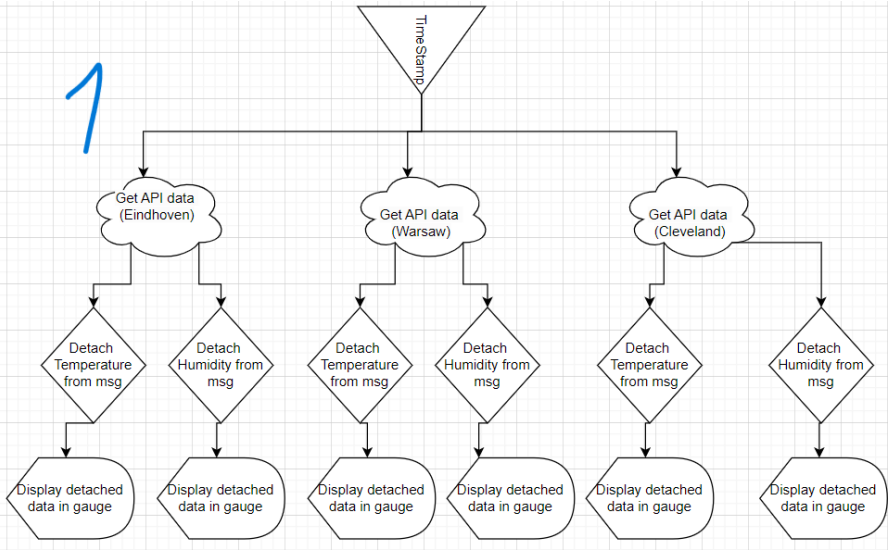
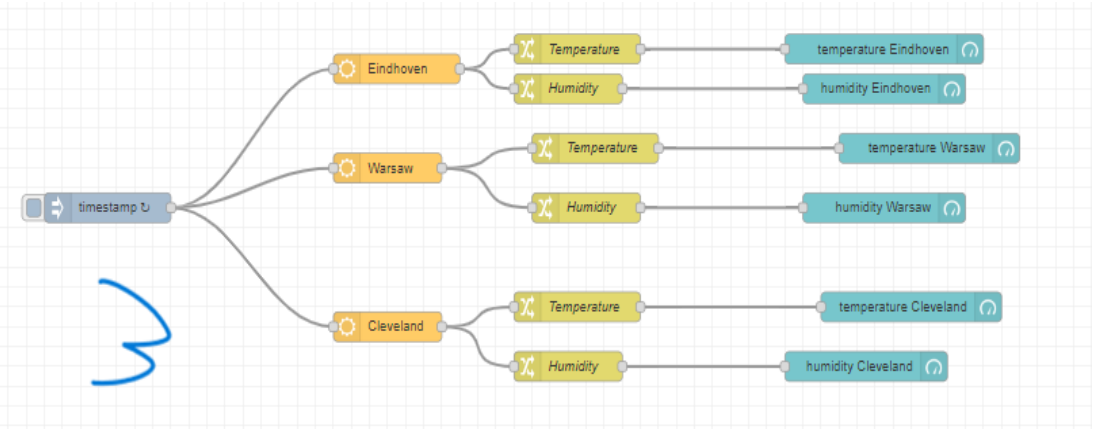
Interactive Dashboard Report
Adam Wiszniowski-Świder
494185

Main goal of this exercise has been revolving around researching and usage of practical nodes and solutions in a node-red dashboard environment. Additional focus has been directed towards corroboration that every used flow is set in the most efficient way, that is fulfilling its requirements without redundant actions that might slow down the process.

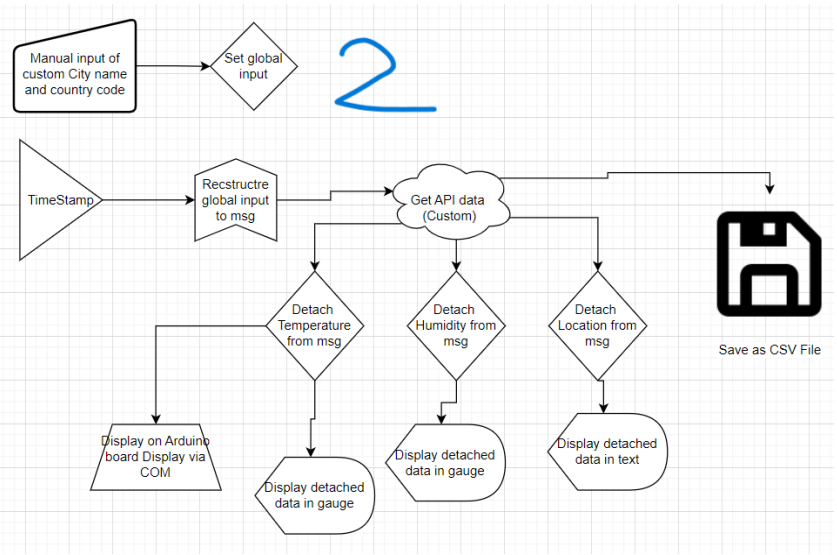
My research has oriented around interconnecting a free, open sourced API Database, with node-red dashboard, and furthermore both sending and receiving data to and from a Hardware device, in this case, and Arduino Uno Board. For that purpose I have studied multiple Node red articles provided by Node red itself, as well as independent online creators.

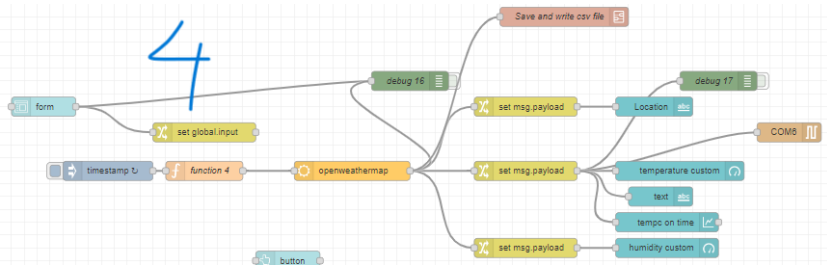
The flow is divided into 2 main functional parts. First one (pic1 & pic3) uses repeating timestamp to generate a request to openwheathermap Database. After receiving data for a particular city, edit nodes come into action. For each city there are 2 of them, one is intercepting information labelled as humidity, and the other is catching data labelled as tempc, which is Temperature in celsius degrees. Later both of those messages are displayed on the dashboard with a gauge node. Temperature gauge node is set to display a range of -20 to 50 degrees, which is sufficient for purposes that this dashboard was created for. For usability reasons gauges are





preset to display blue while temperature is subjectively cold (-20 - 10), green when it's comfortable (10 - 25) and red when it's warm (25 - 50). Similarly the humidity gauge is displaying green when humidity is comfortable (0 - 50) yellow, when it might be uncomfortable (50 - 70) and red when it might cause health problems (70-100). The other main part of the flow is responsible for displaying weather information about the custom city, picked by the user of the dashboard (pic 2 & pic 4). For this part of the dashboard to be activated, first user must manually write a name of a city that they want to see information about, assuming that openwheatermap is operating in said city (despite my efforts I haven't found a city that openwheatermap wouldn't work on, when provided correct spelling), as well as a ISO3166 country code.





(pic5) After typing both of said inputs, and pressing the "Submit" button, the message is set as global.input. When provided with a global variable, timestamp generates a request, that goes into function which is combining 2 global variables into 1 message, which later goes into openwheatermap node, that is not preset to any location. After this step, flow is performing similarly to the flow described above. It separates information about picked city, into humidity and temperature in celsius degrees, however it also

separates a location, so it could be displayed via text node, above all other information, to avoid confusion since the name of column will still be labelled as “Custom”. Additionally after receiving any information from openwheatermap node, a subflow is activated, whose role is first to collect data labelled as humidity, tempc and location with a function that transforms said data into one message. After this step, the message can be transformed into csv format with a csv node, and ultimately saved to an external excel file. Additionally after collecting custom information, a payload with previously detached temperature information is send via COM connection to arduino board, which displays said data instant after receiving it (pic6 &pic7)


```
void loop() {
  // put your main code here, to run repeatedly:

  //K1ButtState = digitalRead(K1Button);
  byte currentState = !digitalRead(K1Button);
  if (currentState != lastK1State){
    lastK1State = currentState;
    if(currentState == HIGH){
      Serial.println(currentState);
    }

  }
  if (Serial.available() > 0){
    float toDisplay = Serial.parseFloat();
    Display.show(toDisplay);
    digitalWrite(PIN_LED, LOW);
  }
}
```

6

Warsaw 10.04.2023

Sources:

- YouTube. (2022, February 3). *Quickly get openweathermap API data with node-red*. YouTube. Retrieved, from https://www.youtube.com/watch?v=B--x7_kOu_Q
- OpenWeatherMap.org. (n.d.). *Current weather data*. OpenWeatherMap. Retrieved, from https://openweathermap.org/current
- YouTube. (2023, February 18). *Build a simple node-red weather dashboard using OpenWeatherMap*. YouTube.Retrieved, from https://www.youtube.com/watch?v=jRF0HwzdXGE&t=167s
- *Red-node-openweathermap*. node. (n.d.). Retrieved, from https://flows.nodered.org/node/node-red-node-openweathermap
-