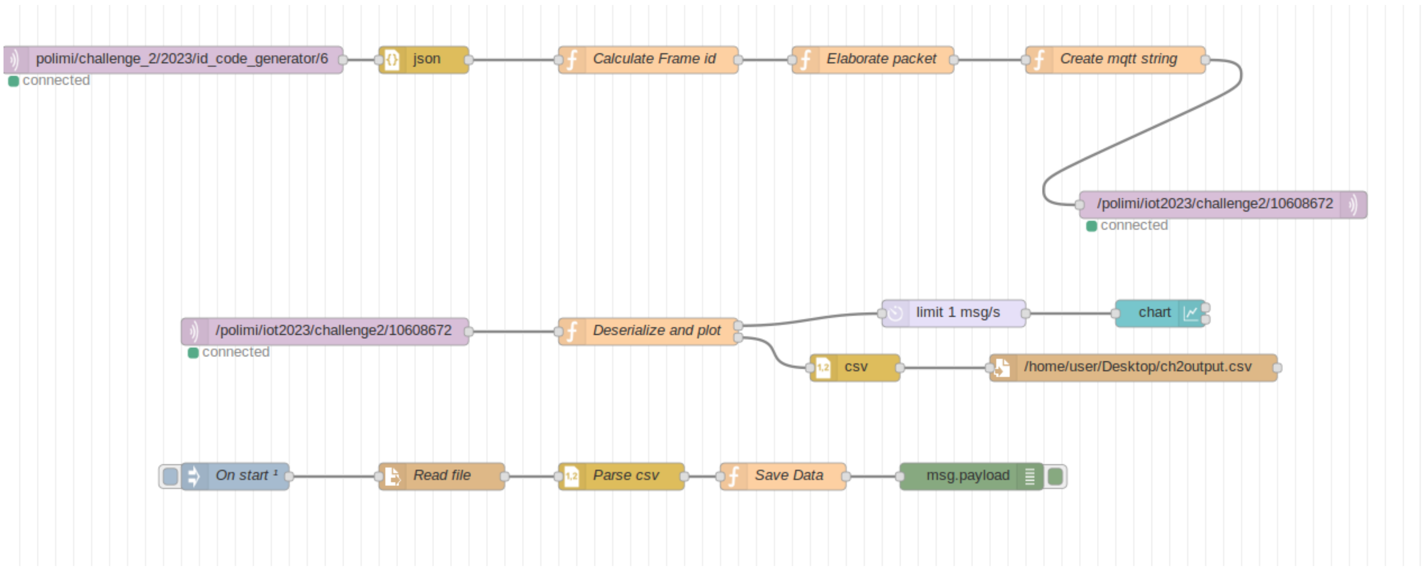
Giovanni De Novellis 10608672 Challenge 2 Report



The program is divided into 3 sub flows:

- The top one, where the incoming id is calculated and the corresponding packet is read and sent to the middle flow if it is of type Publish.
- The middle one, where the content of the payload is read and stored in a csv and in a chart if the temperature is in Celsius.
- The bottom one, where the data needed for the program's execution is read and saved at the beginning.

The detailed explanation of each node is the following:

TOP SUBFLOW

1. MQTT Node: listener to the id_code_generator of the hivemq broker.
2. JSON Node: the incoming string is deserialized into a JavaScript object, assigned to msg.payload and sent to the following function node.
3. Calculate Frame Id: function node where first the counter of the total messages is read and updated, and the message is blocked if the maximum number is exceeded. Otherwise, the packet id is calculated using the incoming id and the person code's last digits and sent to the following node.
4. Elaborate Packet: function node where the packet needed is read from the flow variable, using the input id, and elaborated. First, the type is checked, to see if it is a Publish Message. If it is, then the number of Publish contained in the packet is counted and then the actual payload is deserialized from a JSON string into one or more objects. The objects are then sent to the following node, and in case the number of Publish is higher than the actual object count, then an empty string is sent for the missing ones.
5. Create MQTT string: function node where an object with the timestamp, message id and input payload is created, serialized in a JSON string and sent to the output MQTT node.
6. MQTT Node: sender to the person code channel of the hivemq broker.

MIDDLE SUBFLOW

1. MQTT Node: listener to the person code channel of the hivemq broker for messages sent from the top sub flow.
2. Deserialize and Plot: the input JSON string is deserialized into an object and the unity of measure is read from the payload: if it is Celsius, then the higher value of the temperature range is sent to the Chart node and the whole message is sent to the csv node.
3. Rate Limit: rate limiter to 1 input per second to the chart node in case multiple temperatures are contained in the same packet. This is needed to differentiate them in the x value of time.
4. Chart Node: node used to plot the incoming values in a chart with the Celsius value on the y axis and a timestamp on the x axis.
5. CSV Node: node used to format the incoming payload in a string to be saved in a CSV file.
6. File Node: node used to save the incoming string in a CSV file.

BOTTOM SUBFLOW

1. Inject Node: Node used to inject the sub flow only once, as soon as the whole flow is deployed.
2. File Node: used to read the input CSV file.
3. CSV Node: used to parse the input CSV file into an array of objects, where each object is a packet.
4. Save Data: the array of packets is assigned to a flow variable, that will be accessed by the top sub flow when needed. Also, the counter for the input messages is initialized to 0 and assigned to another flow variable.
5. Debug Node: used to debug the initialization.