IoT Experimental Learning
Week 3 Journal

1. **Describe the experience and what you hope to gain from participating in the experience.**
    - This week's assignment was a very easy for me, I found no issues while trying to complete assignment.
        1. Reviewed weekly assignment material.
        2. Completed Weekly Discussion assignment
        3. Completed Coding assignment
        4. Tested written code
        5. Shared week 4 code base and NodeRed Flow with class

2. **Provide an overview of tasks and key activities (training, discussions, labs, assessments, etc.) in which you were engaged during the week.**

   For week 4 I accomplished the following tasks in chronological order;

- **Monday February 10, 2020**, completed reviewing this week's assignment for the project.
  - I reviewed the Module 4 weekly assignment on the SNHU Brightspace. https://learn.snhu.edu/d2l/le/content/343560/Home

- **Wednesday February 5, 2020**,
  - Completed Weekly Discussion Post
    - https://learn.snhu.edu/d2l/le/343560/discussions/threads/8384590/View

- **Saturday February 08, 2020**,
  - Completed Module 4 Coding assignment using Python, MQTT, and NodeRed. Details as follows:
    - Created the following new Python code files;
      - **Pi_LED_Driver.py**: Main Python flow controller and Raspberry Pi Driver.
      - **MQTTSubscriber.py**: A reusable Python class to encapsulate the MQTT Subscriber behavior as a utility class.
      - **led_driver_msg.json**: JSON Schema for the driver.
      - **Iot-Mod4-flow.json**: JSON representation of the assignment NodeRed Flow.

  - Code can be found on my github at https://github.com/johnnyrich0617/IoT-697

Python Main and Raspberry Pi Driver

```
import time
import grovepi
```

```python
import paho.mqtt.client as mqtt
import jsonpickle
import MQTTSubscriber as ms

# Connect the blue LED to digital port D5
BLUE_LED = 5
RED_LED = 4
GREEN_LED = 6

# Set the blue LED pin to output mode
grovepi.pinMode(BLUE_LED, "OUTPUT")
time.sleep(1)  # give the hardware time to initialize


def on_connect(client, userdata, flags, rc):
    """
     Called each time the client connects to the message broker
     :param client: The client object making the connection
     :param userdata: Arbitrary context specified by the user program
     :param flags: Response flags sent by the message broker
     :param rc: the connection result
     :return: None
     """
    # subscribe to the LEDs topics when connected
    topics = [("SNHU/IT697/leds/red", 2), ("SNHU/IT697/leds/green", 2),
("SNHU/IT697/leds/blue", 2)]
    print("Connected to MQTT Broker........")
    # client.subscribe("SNHU/IT697/leds")
    client.subscribe(topics)
    print("Subscribed to ...", topics)


def on_message(client, userdata, msg):
    """
     Called for each message received
     :param client
     :param userdata:
     :param msg:
     :return: none
     """
    print("Received message from MQTT Broker.....")
    print(msg.topic, msg.payload)
    _topic = msg.topic
    led_payload = jsonpickle.decode(msg.payload)
    if _topic == 'SNHU/IT697/leds/red':
        print("Processing " + _topic + " with msg " + msg.payload)
        # write discrete for RED LED
        grovepi.analogWrite(RED_LED, led_payload['red'])
    elif _topic == 'SNHU/IT697/leds/green':
        print("Processing " + _topic + " with msg " + msg.payload)
        # write discrete for GREEN LED
        grovepi.analogWrite(GREEN_LED, led_payload['green'])
    elif _topic == 'SNHU/IT697/leds/blue':
        print("Processing " + _topic + " with msg " + msg.payload)
        # write discrete for BLUE LED
        grovepi.analogWrite(BLUE_LED, led_payload['blue'])
    else:
        print("No Registered Topic......")


subscriber = ms.MQTTSubscriber(mqtt_host="localhost",
```

```
mqtt_client_id="LOCAL_SUBSCRIBER", port=1883)
subscriber.register_callbacks(on_connect=on_connect, on_message=on_message)
subscriber.connect()
local_client = subscriber.get_client()
local_client.loop_forever()
```

MQTTSubscriber

```
"""
The MQTTSubcriber Class
@author jrichardson
"""

import paho.mqtt.client as mqttClient


class MQTTSubscriber:

    def __init__(self, mqtt_host, mqtt_client_id, port):
        self.mqtt_host = mqtt_host
        self.mqtt_client_id = mqtt_client_id
        self.port = port
        self.mqtt_client = mqttClient.Client(self.mqtt_client_id)

    def connect(self):
        print("MQTTSubscriber::Connecting to client with id = ", self.mqtt_client_id)
        print("MQTTSubscriber::Connecting to host " + self.mqtt_host)
        self.mqtt_client.connect(host=self.mqtt_host, port=self.port)

    def register_callbacks(self, on_connect, on_message):
        self.mqtt_client.on_connect = on_connect
        self.mqtt_client.on_message = on_message
        print("Registered all Callbacks.....")

    def get_client(self):
        return self.mqtt_client

    def get_topic(self):
        return self.topic
```

JSON Schema

```
{
  "red" : "a_value_between_0_and_255",
  "green" : "a_value_between_0_and_255",
  "blue" : "a_value_between_0_and_255"
}
```

Module 4 NodeRed Flow

```
[
    {
        "id": "5fa419d7.3bf168",
        "type": "tab",
        "label": "IOT-697-MOD-4",
        "disabled": false,
        "info": ""
    },
    {
        "id": "7fa95567.40b8cc",
```

```
        "type": "inject",
        "z": "5fa419d7.3bf168",
        "name": "LED_INJECTOR",
        "topic": "",
        "payload": "{\"red\":222,\"green\":221,\"blue\":255}",
        "payloadType": "json",
        "repeat": "",
        "crontab": "",
        "once": false,
        "onceDelay": 0.1,
        "x": 100,
        "y": 260,
        "wires": [
            [
                "f9c7edaf.fedfa"
            ]
        ]
    },
    {
        "id": "f9c7edaf.fedfa",
        "type": "function",
        "z": "5fa419d7.3bf168",
        "name": "CreateDiscreteData",
        "func": "var blueMsg = {payload: msg.payload.blue};\nvar redMsg = {payload:
msg.payload.red};\nvar greenMsg = {payload: msg.payload.green};\n\nreturn [redMsg,
greenMsg, blueMsg];",
        "outputs": 3,
        "noerr": 0,
        "x": 350,
        "y": 260,
        "wires": [
            [
                "2407964.4f72c6a",
                "d6b2aeba.45b17"
            ],
            [
                "feffedee.28657",
                "fb24a10e.78871"
            ],
            [
                "13efc89b.1a8747",
                "36e90291.3b22ee"
            ]
        ]
    },
    {
        "id": "feffedee.28657",
        "type": "debug",
        "z": "5fa419d7.3bf168",
        "name": "GREEN",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "true",
        "targetType": "full",
        "x": 580,
        "y": 200,
        "wires": []
    },
    {
```

```
        "id": "13efc89b.1a8747",
        "type": "debug",
        "z": "5fa419d7.3bf168",
        "name": "BLUE",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "true",
        "targetType": "full",
        "x": 570,
        "y": 329,
        "wires": []
    },
    {
        "id": "2407964.4f72c6a",
        "type": "debug",
        "z": "5fa419d7.3bf168",
        "name": "RED",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "true",
        "targetType": "full",
        "x": 490,
        "y": 100,
        "wires": []
    },
    {
        "id": "a815dd49.b9eb9",
        "type": "debug",
        "z": "5fa419d7.3bf168",
        "name": "REST_PL",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "payload",
        "targetType": "msg",
        "x": 280,
        "y": 100,
        "wires": []
    },
    {
        "id": "12b6a05b.a9e2",
        "type": "debug",
        "z": "5fa419d7.3bf168",
        "name": "BLUE_VALUE",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "true",
        "targetType": "full",
        "x": 740,
        "y": 420,
        "wires": []
    },
    {
        "id": "4e44665f.f60c48",
```

```
        "type": "http in",
        "z": "5fa419d7.3bf168",
        "name": "SetLEDS ",
        "url": "/leds/v2",
        "method": "post",
        "upload": false,
        "swaggerDoc": "",
        "x": 60,
        "y": 100,
        "wires": [
            [
                "1bec80ff.a780df"
            ]
        ]
    },
    {
        "id": "1bec80ff.a780df",
        "type": "function",
        "z": "5fa419d7.3bf168",
        "name": "GetRestPayload",
        "func": "msg.payload = msg.req.body\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 180,
        "y": 160,
        "wires": [
            [
                "f9c7edaf.fedfa",
                "a815dd49.b9eb9",
                "839c12c6.26ed7"
            ]
        ]
    },
    {
        "id": "1106110f.42b87f",
        "type": "debug",
        "z": "5fa419d7.3bf168",
        "name": "GREEN_VALUE",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "true",
        "targetType": "full",
        "x": 740,
        "y": 300,
        "wires": []
    },
    {
        "id": "8b4855c9.909eb8",
        "type": "debug",
        "z": "5fa419d7.3bf168",
        "name": "RED_VALUE",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "true",
        "targetType": "full",
        "x": 730,
        "y": 100,
```

```
        "wires": []
    },
    {
        "id": "839c12c6.26ed7",
        "type": "http response",
        "z": "5fa419d7.3bf168",
        "name": "RestResponse",
        "statusCode": "",
        "headers": {},
        "x": 360,
        "y": 380,
        "wires": []
    },
    {
        "id": "1e8f3bae.72a634",
        "type": "function",
        "z": "5fa419d7.3bf168",
        "name": "CommonMsgFmtr",
        "func": "var LEDS = global.get('LEDS')||{};\nvar topic =
msg.topic;\n\nswitch(topic){\n    \n    case \"red\":\n        msg.payload = {red :
msg.payload};\n        LEDS.red = msg.payload.red;\n        global.set('LEDS',
LEDS);\n        break;\n    \n    case \"green\":\n        msg.payload = {green :
msg.payload};\n        LEDS.green = msg.payload.green;\n        global.set('LEDS',
LEDS);\n        break;\n    \n    case \"blue\":\n        msg.payload = {blue :
msg.payload};\n        LEDS.blue = msg.payload.blue;\n        global.set('LEDS',
LEDS);\n        break;\n}\n\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 910,
        "y": 260,
        "wires": [
            [
                "8c5d6cce.1a1b3",
                "d5926373.f62d"
            ]
        ]
    },
    {
        "id": "8c5d6cce.1a1b3",
        "type": "debug",
        "z": "5fa419d7.3bf168",
        "name": "Formatted_MSG",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "true",
        "targetType": "full",
        "x": 1110,
        "y": 200,
        "wires": []
    },
    {
        "id": "8ac412f2.33dd",
        "type": "change",
        "z": "5fa419d7.3bf168",
        "name": "DeletePayloadMsgId",
        "rules": [
            {
                "t": "delete",
                "p": "payload._msgid",
```

```
                "pt": "msg"
            }
        ],
        "action": "",
        "property": "",
        "from": "",
        "to": "",
        "reg": false,
        "x": 580,
        "y": 560,
        "wires": [
            [
                "839c12c6.26ed7",
                "5186ae1c.d3b37"
            ]
        ]
    },
    {
        "id": "72886002.dd0c",
        "type": "http in",
        "z": "5fa419d7.3bf168",
        "name": "GetLEDValues",
        "url": "/leds/v2",
        "method": "get",
        "upload": false,
        "swaggerDoc": "",
        "x": 90,
        "y": 560,
        "wires": [
            [
                "3561b62.5e6ee4a"
            ]
        ]
    },
    {
        "id": "3561b62.5e6ee4a",
        "type": "function",
        "z": "5fa419d7.3bf168",
        "name": "GET_GLOBALS",
        "func": "msg.payload = global.get('LEDS')||{};\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 320,
        "y": 560,
        "wires": [
            [
                "8ac412f2.33dd",
                "5891436.44954bc"
            ]
        ]
    },
    {
        "id": "5186ae1c.d3b37",
        "type": "debug",
        "z": "5fa419d7.3bf168",
        "name": "DELETE",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "true",
```

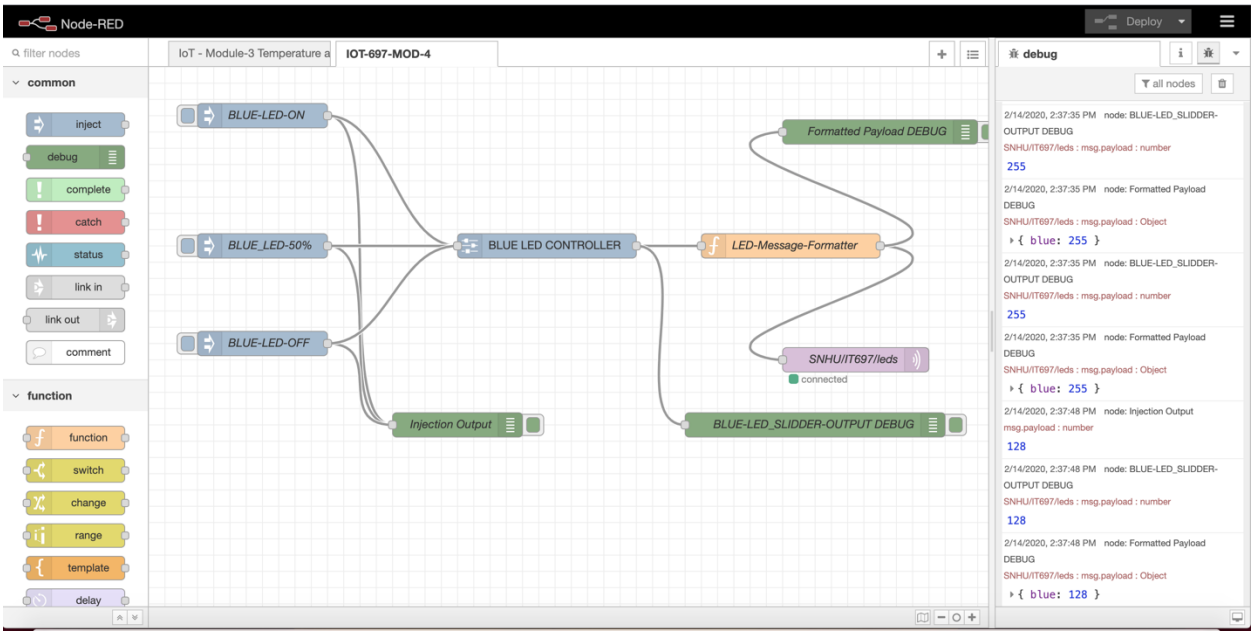
```
        "targetType": "full",
        "x": 700,
        "y": 620,
        "wires": []
    },
    {
        "id": "5891436.44954bc",
        "type": "debug",
        "z": "5fa419d7.3bf168",
        "name": "GET_GLOBALS",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "true",
        "targetType": "full",
        "x": 380,
        "y": 620,
        "wires": []
    },
    {
        "id": "998d95d5.87e458",
        "type": "mqtt out",
        "z": "5fa419d7.3bf168",
        "name": "RED_TOPIC",
        "topic": "SNHU/IT697/leds/red",
        "qos": "2",
        "retain": "",
        "broker": "62d0ad35.389d94",
        "x": 1170,
        "y": 300,
        "wires": []
    },
    {
        "id": "d5926373.f62d",
        "type": "switch",
        "z": "5fa419d7.3bf168",
        "name": "Seperator",
        "property": "topic",
        "propertyType": "msg",
        "rules": [
            {
                "t": "eq",
                "v": "red",
                "vt": "str"
            },
            {
                "t": "eq",
                "v": "green",
                "vt": "str"
            },
            {
                "t": "eq",
                "v": "blue",
                "vt": "str"
            }
        ],
        "checkall": "false",
        "repair": false,
        "outputs": 3,
        "x": 1000,
```

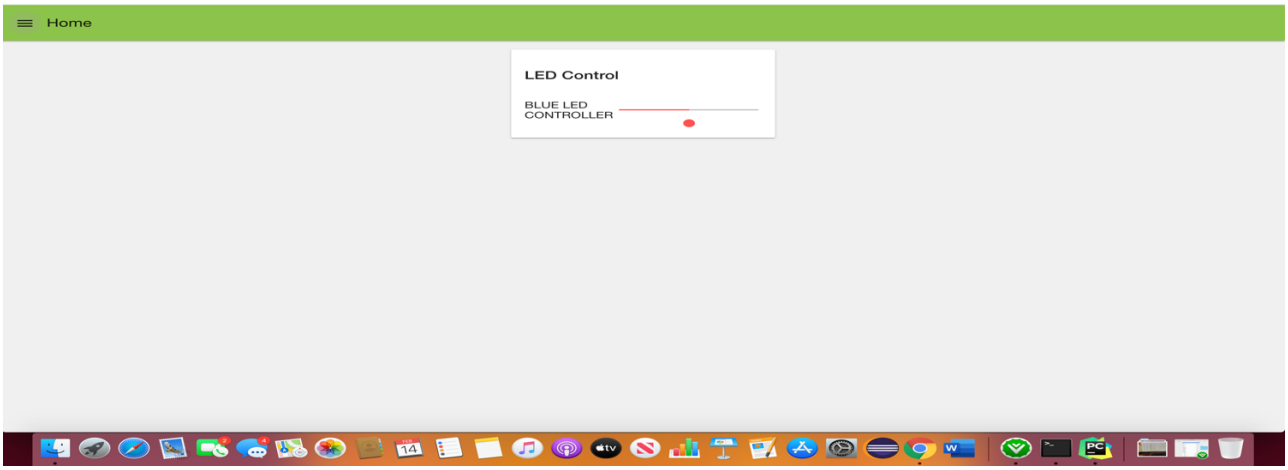
```
        "y": 380,
        "wires": [
            [
                "998d95d5.87e458"
            ],
            [
                "9f00e2ad.2ffcc"
            ],
            [
                "efd08771.93a7d8"
            ]
        ]
    },
    {
        "id": "9f00e2ad.2ffcc",
        "type": "mqtt out",
        "z": "5fa419d7.3bf168",
        "name": "GREEN_TOPIC",
        "topic": "SNHU/IT697/leds/green",
        "qos": "2",
        "retain": "",
        "broker": "62d0ad35.389d94",
        "x": 1180,
        "y": 380,
        "wires": []
    },
    {
        "id": "efd08771.93a7d8",
        "type": "mqtt out",
        "z": "5fa419d7.3bf168",
        "name": "BLUE_TOPIC",
        "topic": "SNHU/IT697/leds/blue",
        "qos": "2",
        "retain": "",
        "broker": "62d0ad35.389d94",
        "x": 1180,
        "y": 460,
        "wires": []
    },
    {
        "id": "d6b2aeba.45b17",
        "type": "ui_slider",
        "z": "5fa419d7.3bf168",
        "tab": "befbb3c6.ccca3",
        "name": "Red Slider",
        "topic": "red",
        "group": "LEDS",
        "order": 1,
        "min": 0,
        "max": "255",
        "x": 570,
        "y": 140,
        "wires": [
            [
                "8b4855c9.909eb8",
                "1e8f3bae.72a634"
            ]
        ]
    },
    {
        "id": "fb24a10e.78871",
```

```
        "type": "ui_slider",
        "z": "5fa419d7.3bf168",
        "tab": "befbb3c6.ccca3",
        "name": "Green Slider",
        "topic": "green",
        "group": "LEDS",
        "order": 1,
        "min": 0,
        "max": "255",
        "x": 570,
        "y": 259,
        "wires": [
            [
                "1106110f.42b87f",
                "1e8f3bae.72a634"
            ]
        ]
    },
    {
        "id": "36e90291.3b22ee",
        "type": "ui_slider",
        "z": "5fa419d7.3bf168",
        "tab": "befbb3c6.ccca3",
        "name": "Blue Slider",
        "topic": "blue",
        "group": "LEDS",
        "order": 1,
        "min": 0,
        "max": "255",
        "x": 570,
        "y": 372,
        "wires": [
            [
                "1e8f3bae.72a634",
                "12b6a05b.a9e2"
            ]
        ]
    },
    {
        "id": "cd098758.55dd18",
        "type": "comment",
        "z": "5fa419d7.3bf168",
        "name": "Discretes",
        "info": "Each discrete has its own topic for performance",
        "x": 1000,
        "y": 440,
        "wires": []
    },
    {
        "id": "62d0ad35.389d94",
        "type": "mqtt-broker",
        "z": "",
        "name": "Localhost Broker",
        "broker": "localhost",
        "port": "1883",
        "clientid": "",
        "usetls": false,
        "compatmode": false,
        "keepalive": "60",
        "cleansession": true,
        "birthTopic": "",
```

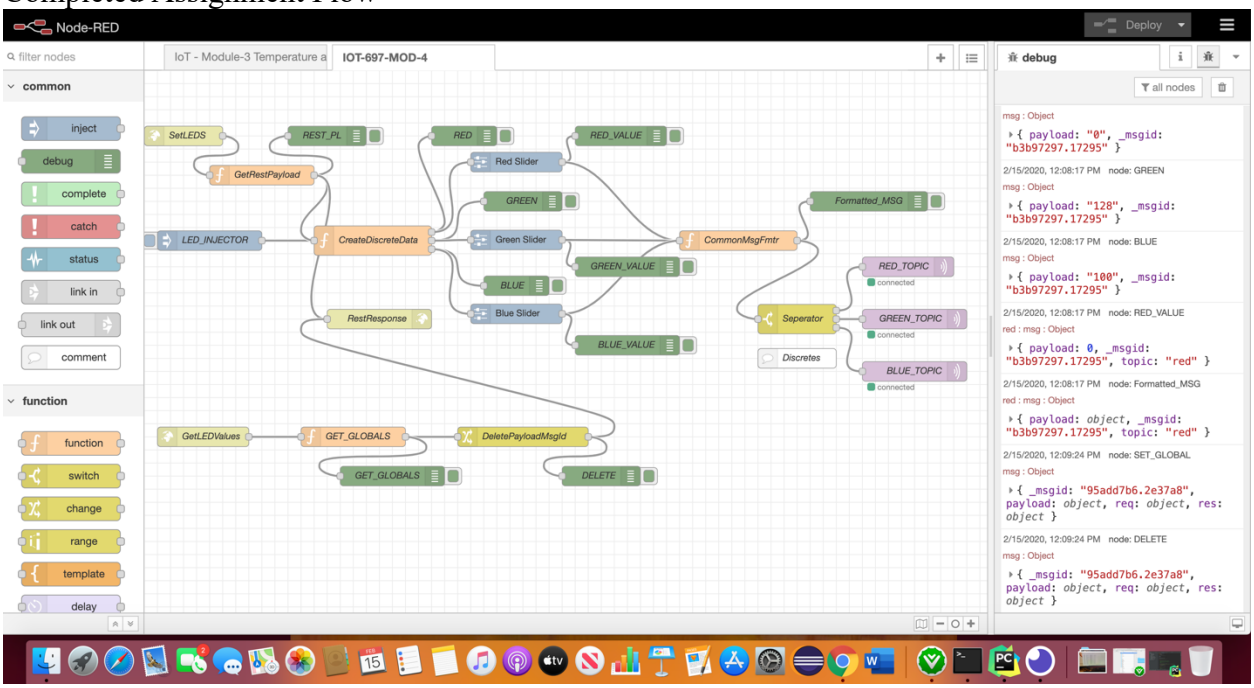
```
        "birthQos": "0",
        "birthPayload": "",
        "closeTopic": "",
        "closeQos": "0",
        "closePayload": "",
        "willTopic": "",
        "willQos": "0",
        "willPayload": ""
    },
    {
        "id": "befbb3c6.ccca3",
        "type": "ui_tab",
        "z": "",
        "name": "Home",
        "icon": "dashboard",
        "order": "1"
    }
]
```

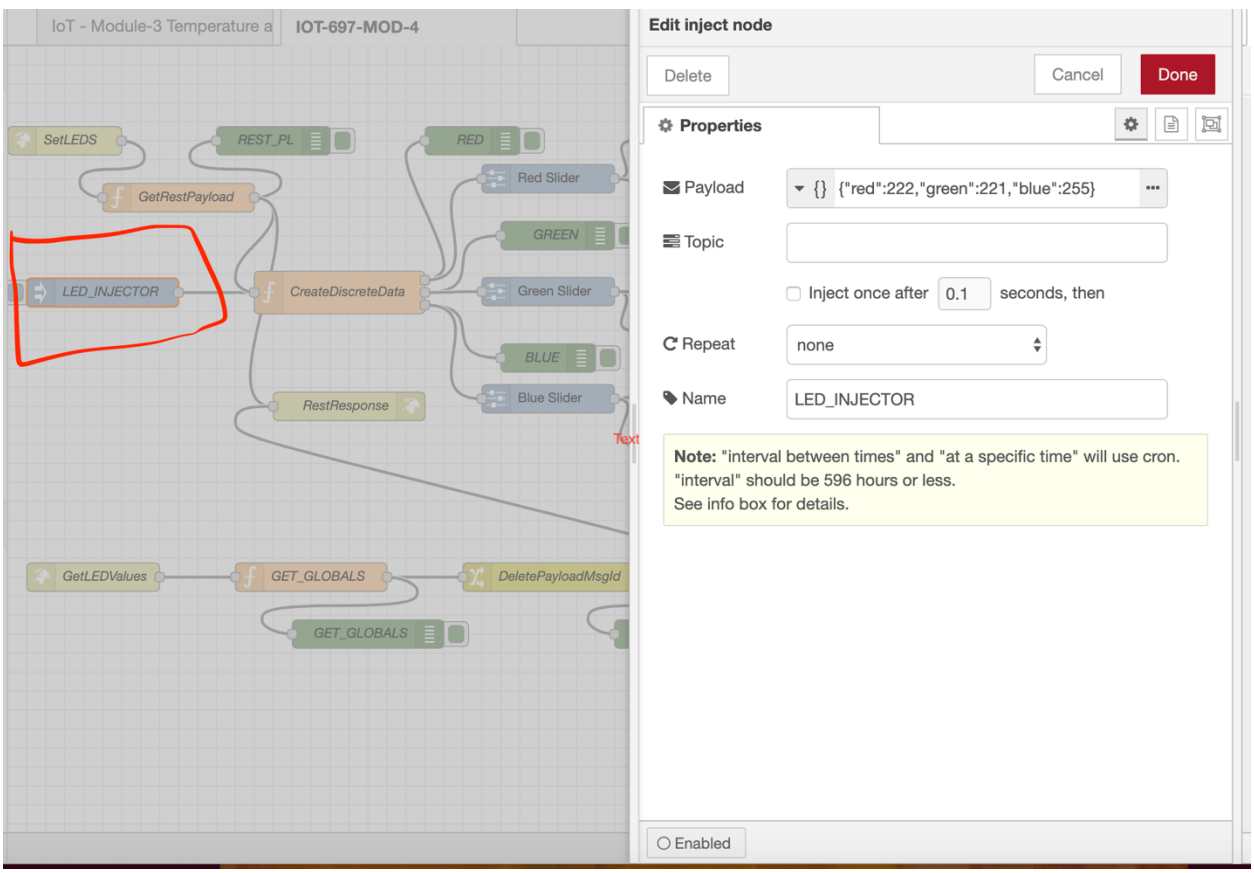

Initial Flow and Module Code Test

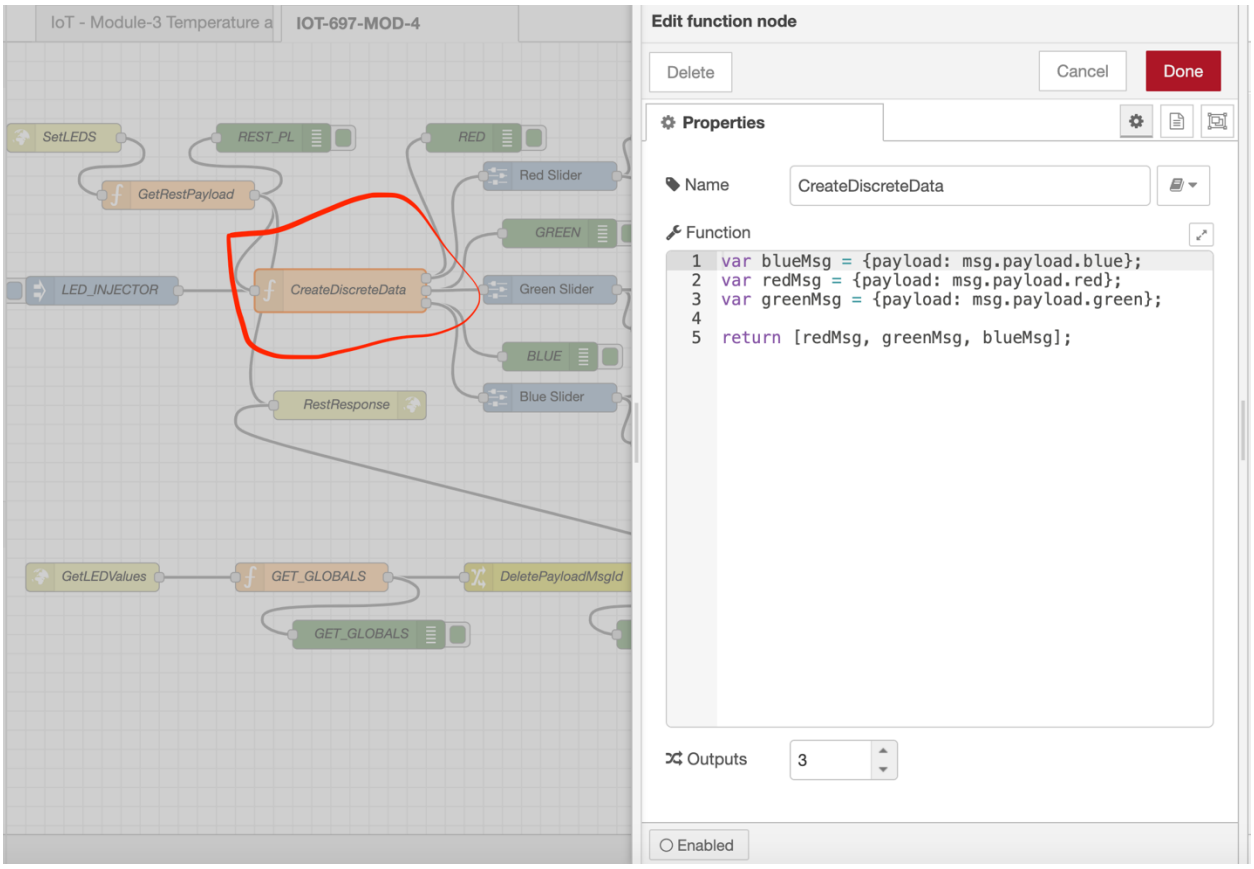


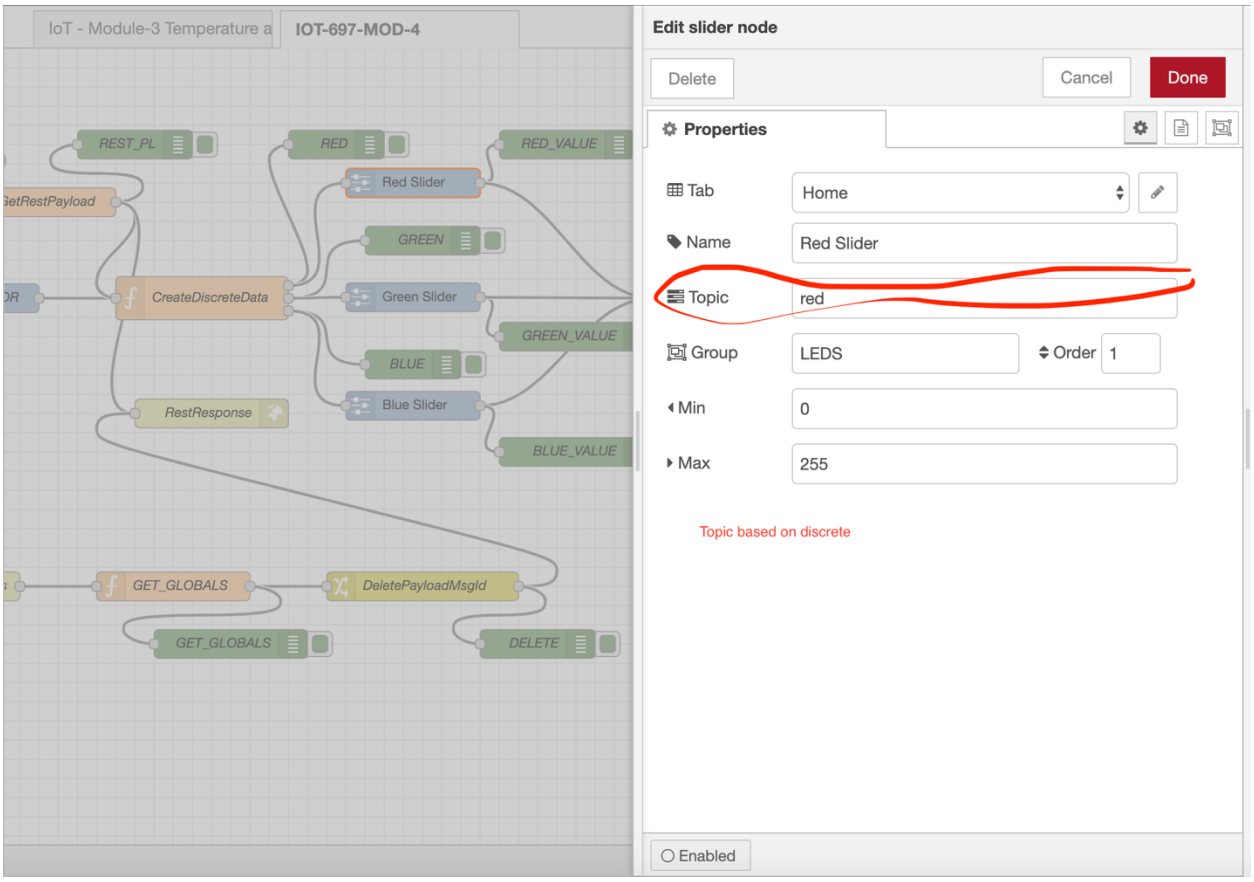


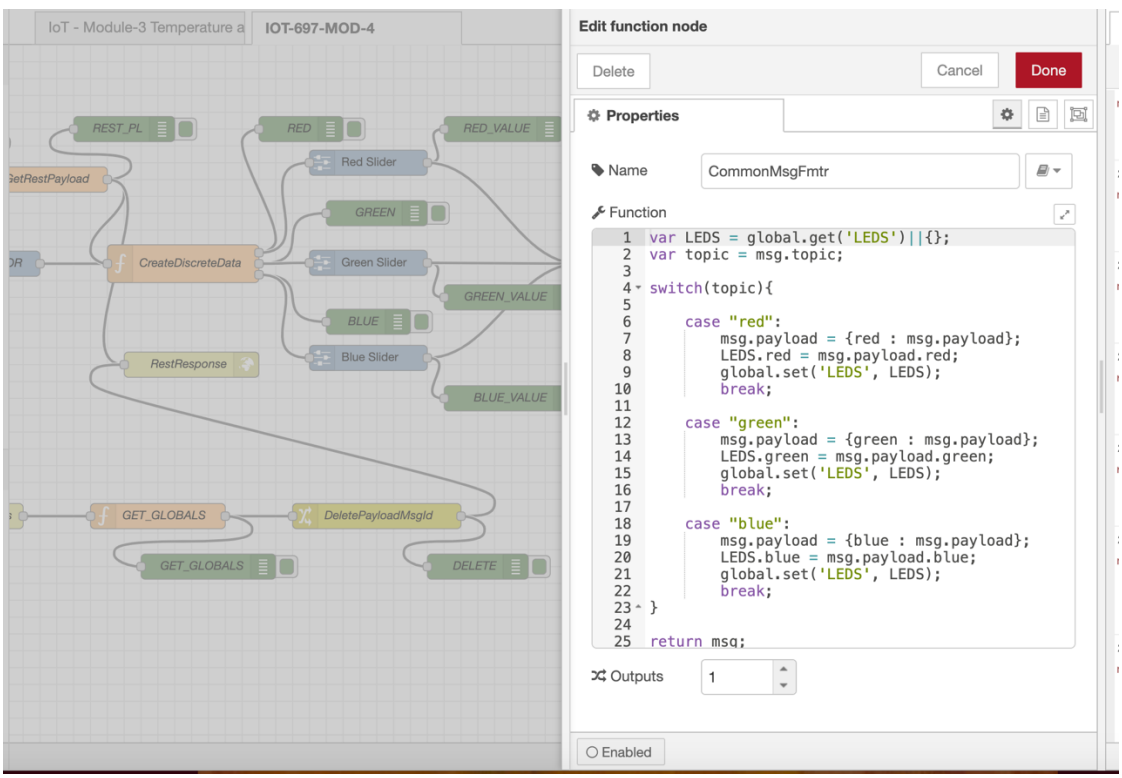
Completed Assignment Flow

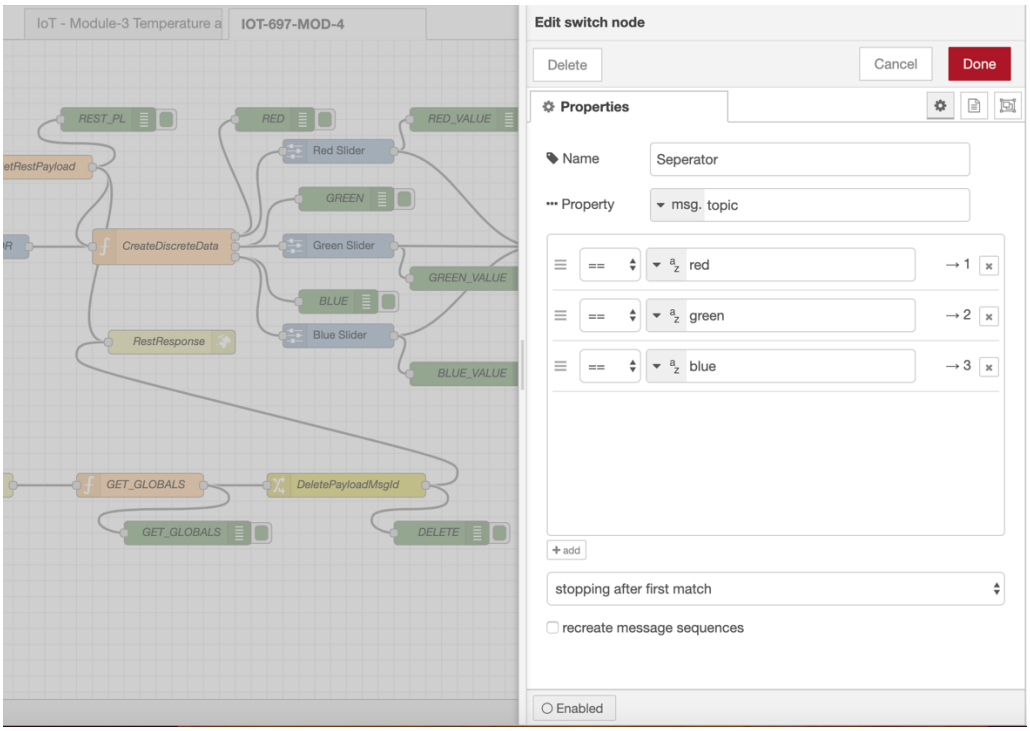


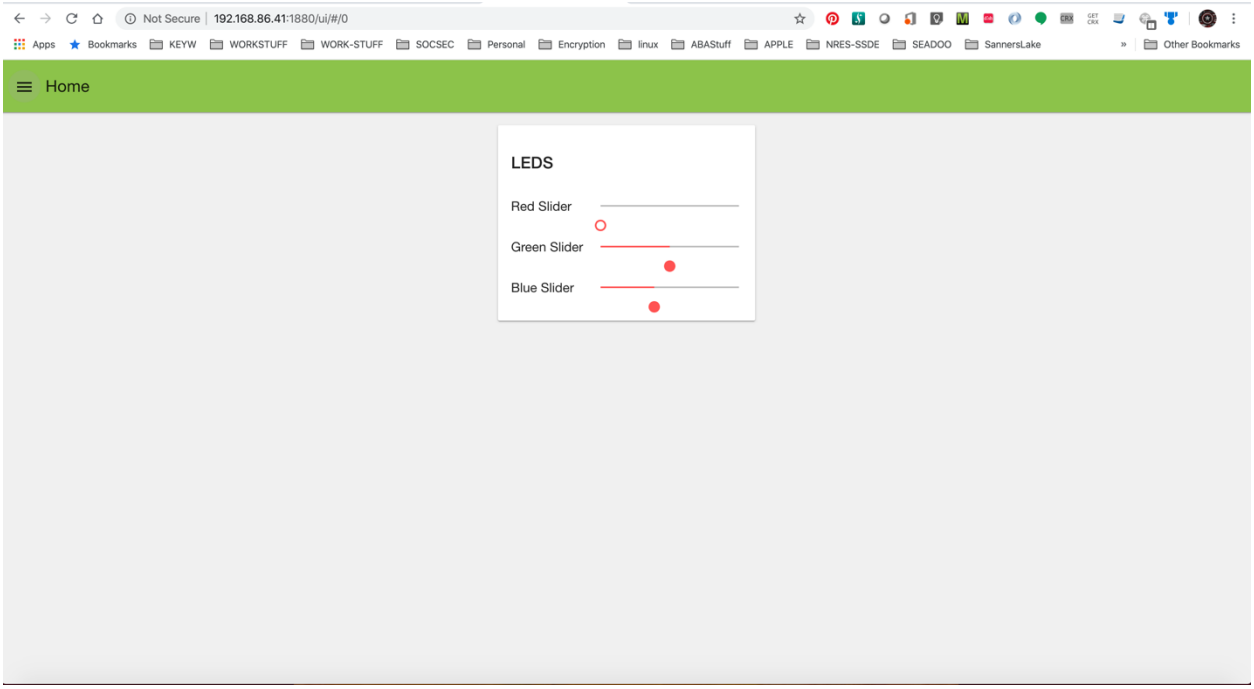



IoT - Module-3 Temperature a
IOT-697-MOD-4
SetLEDS
REST_PL
RED
Red Slider
GetRestPayload
GREEN
LED_INJECTOR
CreateDiscreteData
Green Slider
BLUE
RestResponse
Blue Slider
GetLEDValues
GET_GLOBALS
DeletePayloadMsgId
GET_GLOBALS
Edit function node
Delete
Cancel
Done
Properties
Name
CreateDiscreteData
Function
var blueMsg = {payload: msg.payload.blue};
var redMsg = {payload: msg.payload.red};
var greenMsg = {payload: msg.payload.green};
return [redMsg, greenMsg, blueMsg];
Outputs
3
Enabled

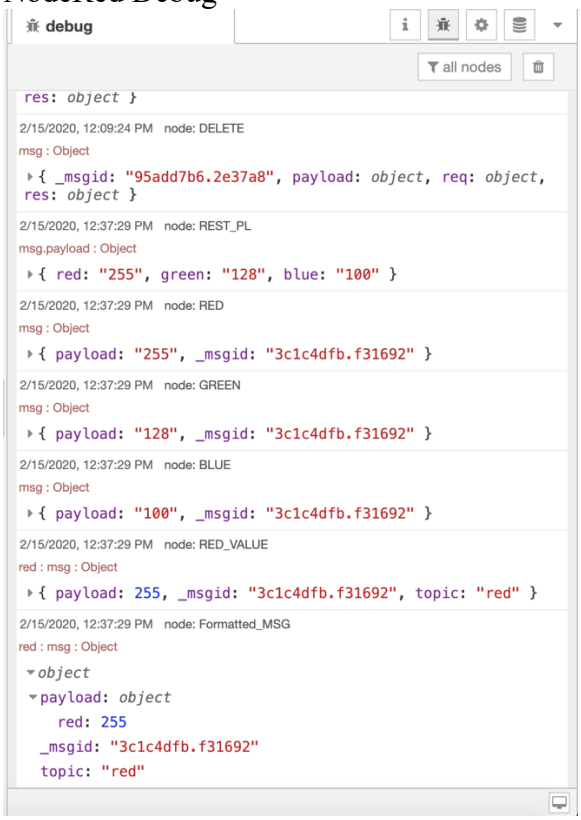

IoT - Module-3 Temperature a
IOT-697-MOD-4
REST_PL
RED
RED_VALUE
Red Slider
GetRestPayload
GREEN
CreateDiscreteData
Green Slider
GREEN_VALUE
BLUE
RestResponse
Blue Slider
BLUE_VALUE
GET_GLOBALS
DeletePayloadMsgId
GET_GLOBALS
DELETE
Edit slider node
Delete
Cancel
Done
Properties
Tab
Home
Name
Red Slider
Topic
red
Group
LEDS
Order
1
Min
0
Max
255
Topic based on discrete
Enabled


IoT - Module-3 Temperature a
IOT-697-MOD-4
REST_PL
RED
RED_VALUE
Red Slider
GetRestPayload
GREEN
CreateDiscreteData
Green Slider
GREEN_VALUE
BLUE
RestResponse
Blue Slider
BLUE_VALUE
GET_GLOBALS
DeletePayloadMsgId
GET_GLOBALS
DELETE
Edit function node
Delete
Cancel
Done
Properties
Name
CommonMsgFmtr
Function
var LEDS = global.get('LEDS')||{};
var topic = msg.topic;
switch(topic){
case "red":
msg.payload = {red : msg.payload};
LEDS.red = msg.payload.red;
global.set('LEDS', LEDS);
break;
case "green":
msg.payload = {green : msg.payload};
LEDS.green = msg.payload.green;
global.set('LEDS', LEDS);
break;
case "blue":
msg.payload = {blue : msg.payload};
LEDS.blue = msg.payload.blue;
global.set('LEDS', LEDS);
break;
}
return msg;
Outputs
1
Enabled

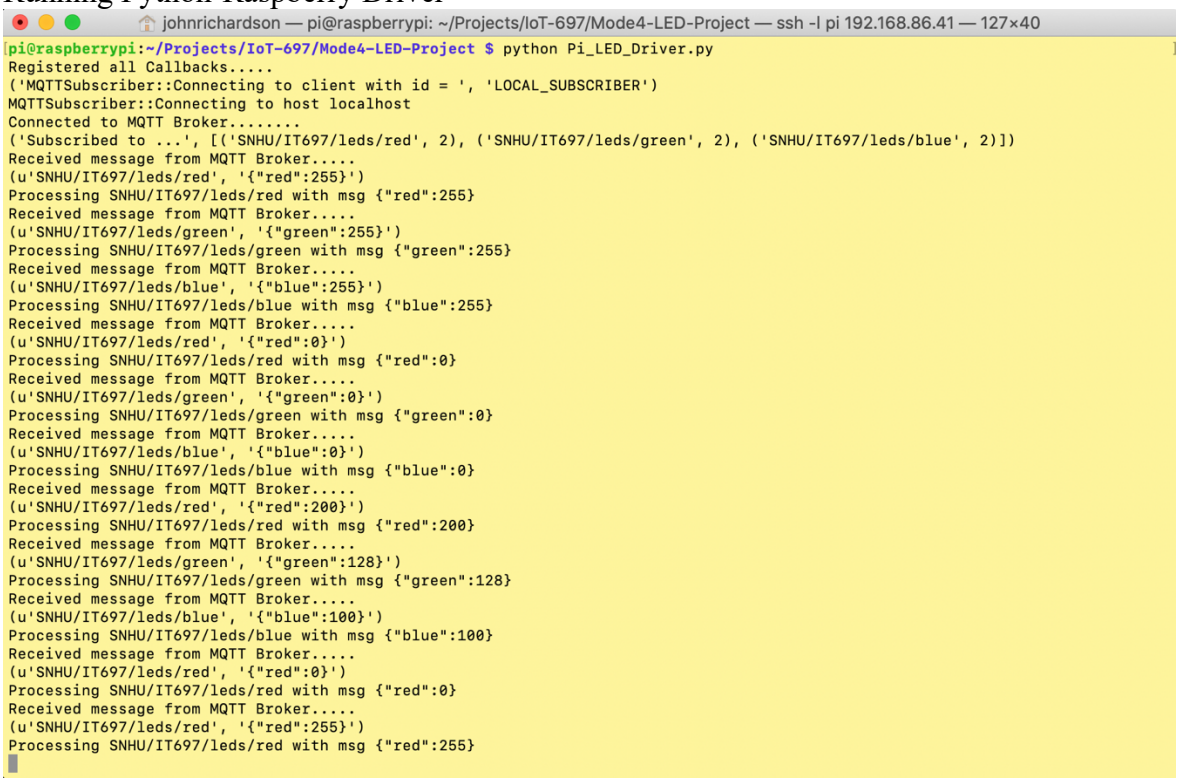

IoT - Module-3 Temperature a
IOT-697-MOD-4
REST_PL
RED
RED_VALUE
Red Slider
GREEN
CreateDiscreteData
Green Slider
GREEN_VALUE
BLUE
RestResponse
Blue Slider
BLUE_VALUE
GET_GLOBALS
DeletePayloadMsgId
GET_GLOBALS
DELETE
Edit switch node
Delete
Cancel
Done
Properties
Name
Seperator
Property
msg. topic
== red → 1
== green → 2
== blue → 3
+ add
stopping after first match
recreate message sequences
Enabled


Not Secure | 192.168.86.41:1880/ui/#/0
Apps
Bookmarks
KEYW
WORKSTUFF
WORK-STUFF
SOCSEC
Personal
Encryption
linux
ABAStuff
APPLE
NRES-SSDE
SEADOO
SannersLake
Other Bookmarks
Home
LEDS
Red Slider
Green Slider
Blue Slider

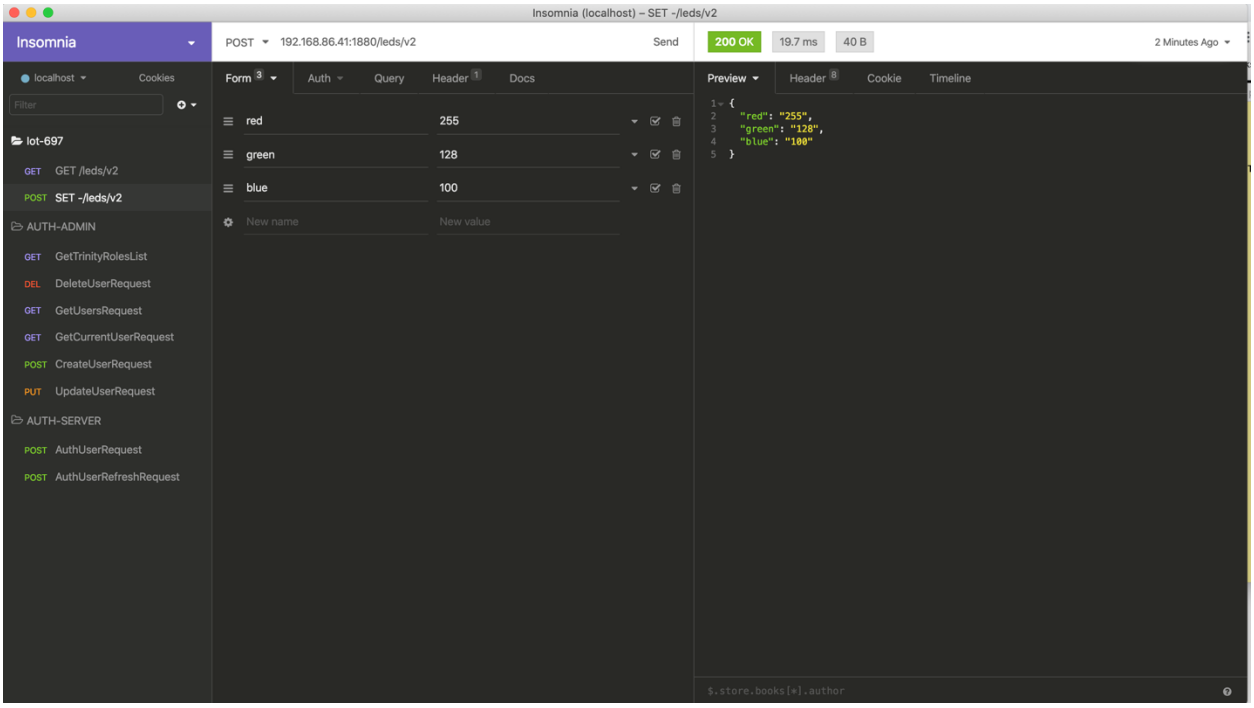
NodeRed Debug

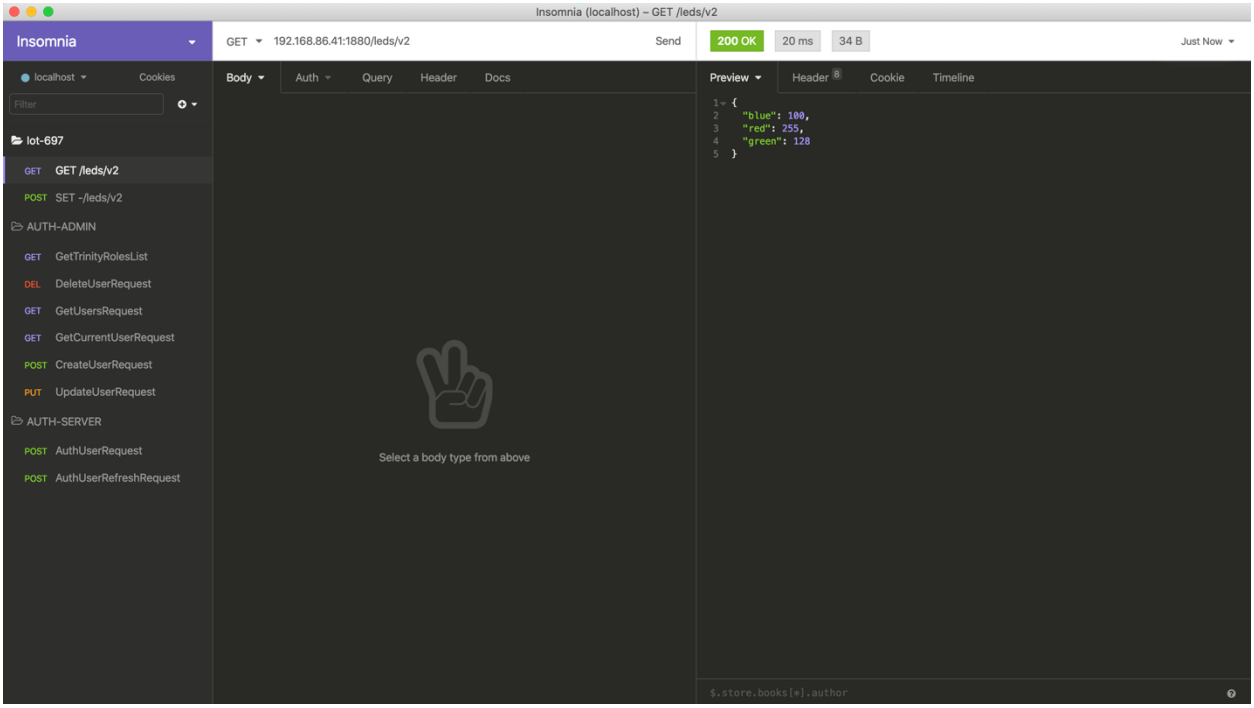


Running Python-Raspberry Driver



Insomnia Rest Client





- Created GitHub Gist to share Code with class:
  - https://gist.github.com/johnnyrich0617/a7cff6c6119d888b2d0f3878bccc3eac
- Created Discussion Post to shar Gist with class

- Sunday February 09, 2020,
  - Created Week 3 Timesheet
  - Created Week 3 Journal
  - Replied to class Module 4 Discussion Posts