CO326-2022

Computer Systems Engineering: Industrial Networks

Lighting

Group B

Lighting - Group B

E/17/352	Sensors to read data and send to MQTT Server
E/17/398	Read data from MQTT Server and take decisions and control actuators
E/17/201	Read data from MQTT Server display status of the system on SCADA
E/17/312	Inputs from the SCADA should send to the MQTT server
E/17/065	Process controller with operating and optimizing process and algorithms
E/17/240	MQTT Data, Commands and events should be stored in the database
E/17/122	Web interface or SCADA pages should display the data in the database
E/17/294	Team Leader
E/17/153	Data analytics: Prediction, Optimization, Correlation

Introduction

One of the largest areas of energy consumption in any building is its lighting system. This can have a significant effect on operating costs and, hence, the financial well-being of businesses. Intelligent lighting systems use IoT sensors and advanced analytics to automatically adjust lighting levels to create healthy illumination according to individuals' preferences, occupancy, availability of daylight, and other factors. This isn't just convenient, it can affect human health.

As the team **Lighting** of the project **Smart Building** mainly our purposes are

- Controlling the lighting intensity of the lights according to the environment
- Switching on/off lighting according to the occupancy
- Switching on/off lighting according to the time of the day

The lighting control system will allow the user to dim, zone and turn off lights as needed to reduce wasted energy expenditures or illuminate areas when occupied.

We are implementing a prototype model which we can use to build the lighting system.

Tasks

The main tasks comes under lighting control is as follows

1 - Read data from sensors and send to MQTT server
2 - Read data from MQTT Server and take decisions and control actuators
3 - Read data from MQTT Server display status of the system on SCADA
4 - Inputs from the SCADA should send to the MQTT server
5 - Process controller with operating and optimizing process and algorithms
6 - MQTT Data, Commands and events should be stored in the database
7 - Web interface or SCADA pages should display the data in the database
8 - Data analytics: Prediction, Optimization, Correlation

1 - Read data from sensors and send to MQTT server

As the initial step of the lighting system of the smart building, data must be taken from the lighting sensors and sent (published) to the MQTT broker. In our system, the inputs needed to make decisions on controlling the light system are taken in three ways.

1. Light sensor data - to control lighting based on the light intensity of the surrounding
2. Occupancy sensor data - to control lighting based on the presence of a person
3. The on/off states of the switches

As occupancy sensor data will be published by a separate team, we will focus on reading light sensor data and publishing the data to the MQTT broker in this section.

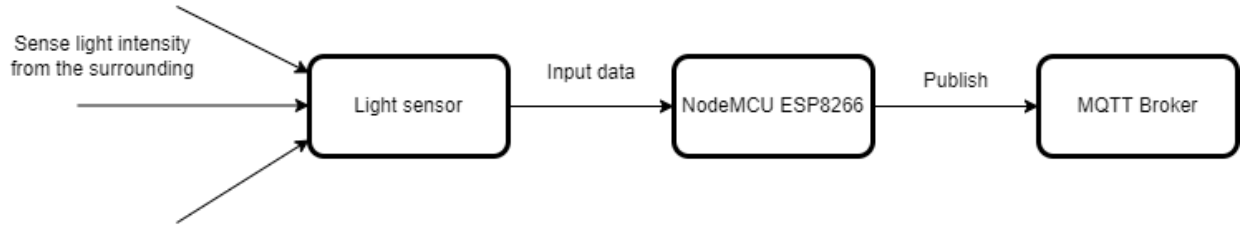
We considered the options available to get light sensor data. One of the options we considered was to use BH1750 light sensor. But based on the availability, for the prototype design LDR was used instead of BH1750 sensor.

LDR sensor

Photoresistors, also known as light dependent resistors (LDR), are light sensitive devices most often used to indicate the presence or absence of light, or to measure the light intensity. It is a component that has a (variable) resistance that changes with the light intensity that falls upon it. This allows them to be used in light sensing circuits. This variation of resistance is converted into a voltage variation.

Making an MQTT enabled IoT device using NodeMCU ESP8266

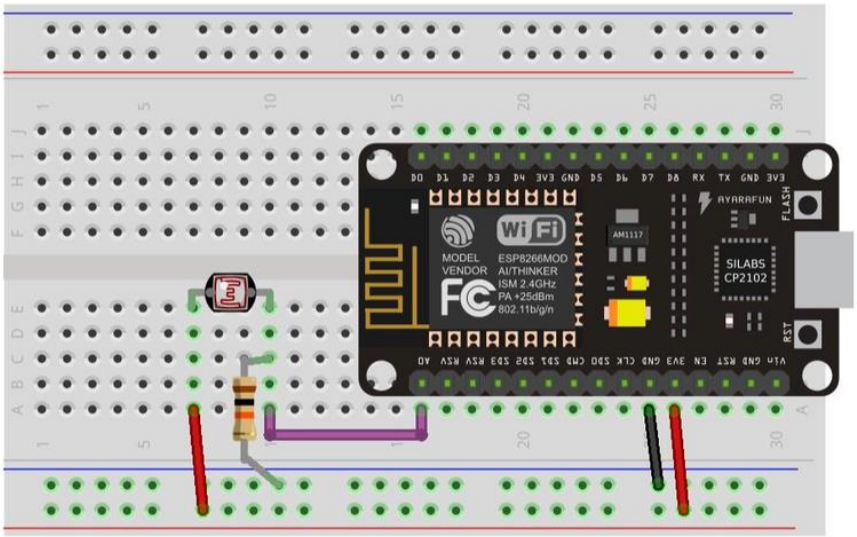
To get data from the light sensor and publish it to the MQTT broker set up at the department, an MQTT enabled IoT device will be set up using NodeMCU ESP8266. The light sensor will output the light intensity data and the controller will publish the data to the MQTT broker using the ESP8266 module which is a wifi module that serves as a microcontroller enhancement to connect directly to wifi and create TCP / IP connections.



Hardware implementation

Items used:
NodeMCU ESP8266
LDR
Resistor (220 Ohm)
Jumper wires

The light sensor will be connected to the controller's Analog inputs and then, the code to constantly (every 2 seconds) read the sensor data and publish to the relevant topic via the MQTT protocol will be uploaded to the controller. Once the connection between the controller and the MQTT broker is established, the broker will receive the light-sensing data continuously. According to the system design, LDRs will be categorized as inside and outside based on where they are placed.
We implemented the following hardware circuit which takes light sensor data from one LDR and publish it to the MQTT broker. It includes a NodeMCU ESP8266, LDR sensor, and a resistor as main components. By changing the topic that is published, we are planning to demonstrate the light controlling system.



Publishing to MQTT topics

When publishing data to the MQTT broker the following topic/tag naming convention will be used.

326project/smartbuilding/lighting/<floor_number>/<room_number>/lightsensor/inside/<sensor_number>/

326project/smartbuilding/lighting/<floor_number>/<room_number>/lightsensor/outside/<sensor_number>/

For the demonstration sensor data will be published to the below topic.

326project/smartbuilding/lighting/0/1/lightsensor/inside/1/

2 - Read data from MQTT Server and take decisions and control actuators

Lightning system control can be centralized or distributed. In our case, a distributed control system will be used. Therefore each sensing module has a controller and communicates with neighbors to calculate dimming levels.

After each sensor module sends required information to the main controller(NodeMCU) for data processing, the main controller does data processing and drives desired lighting conditions by controlling the LED drivers. So depending on data received from sensor modules, Arduino is programmed to control the lighting conditions in the building. It transmits commands to the ESP8266 depending on LDR sensor data and occupancy sensor data to turn them on or off and adjust the brightness of the lights.

Data read from MQTT server uses (to mainly take two control decisions)

1. To transmit a command to turn the light on or off

2.To transmit a command to adjust the brightness of the light

In our design we have to subscribe from MQTT server on following topics convention to get the command to whether light is on or off. → A

326project/smartbuilding/lighting/<floor_number>/<room_number>/switch/inside/<led_number>/

326project/smartbuilding/lighting/<floor_number>/<room_number>/switch/outside/<led_number>/

In our prototype design there are 3 inside bulbs and 1 outside bulbs.

After lighting the bulb, the status of the bulb will publish in MQTT server on following topic convention. → B

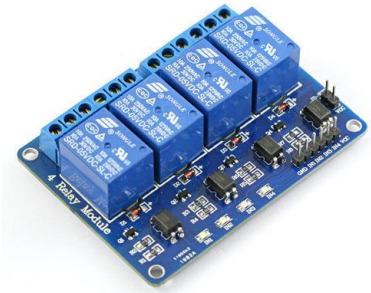
326project/smartbuilding/lighting/<floor_number>/<room_number>/led/inside/<led_number>/

326project/smartbuilding/lighting/<floor_number>/<room_number>/led/outside/<led_number>/

Equipments used:

1. NodeMCU ESP8266

2. 4-channel 5V Relay Module



3. 3 Bulbs and 1 LED

4. Arduino UNO board

Here we programmed NodeMCU using Arduino IDE to control 3 digital pins as ‘High’ or ‘Low’ and 1 digital pin to change brightness value depending on the result of subscribed

mqtt topics (A). Then to transfer that information to 3 light bulbs we used 4-channel 5V relay module. So we can control high voltage light bulbs.

3 bulbs - used for on and off

1 LED - used for change the brightness value

This is the hardware circuit for control 3 light bulbs. (Prototype design)

3 - Read data from MQTT Server display status of the system on SCADA

Source of data - MQTT server

SCADA is considered as a MQTT client which subscribed to relevant topics, so that the published data of sensor and actuators can be read.

Output - Status of the system

Displayed through node red dashboard.

Components status and other relevant details of the lighting subsystem

Initially, in order to communicate between the SCADA system(MQTT Client) and the MQTT server, they subscribed to the same topic so that they could exchange data.The data published by the publishers are received by the relevant subscribers. The status of the system components are displayed in the SCADA using node-red dashboard. It also facilitates both manual and auto configuration of system actuators.

Status of the sensors used are displayed as follows and the data is obtained through MQTT server and the topic used is mentioned below,

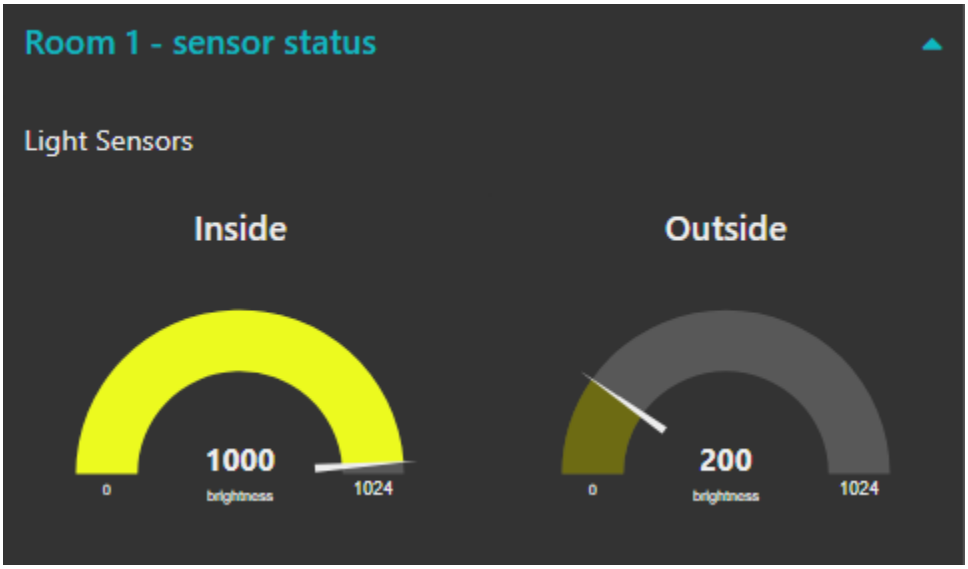
Status of sensors

This displays the sensor data that is obtained by two light sensors used to sense light intensity in a room. The two sensors are placed outside and inside of the room, the intensity is sensed and data is published as a value between 0 and 1024, where 1024 means that there is enough light.

- Inside light sensor
- Outside light sensor

326project/smartbuilding/lighting/<floor_number>/<room_number>/lightsensor/inside/<sensor_number>/

326project/smartbuilding/lighting/<floor_number>/<room_number>/lightsensor/outside/<sensor_number>/



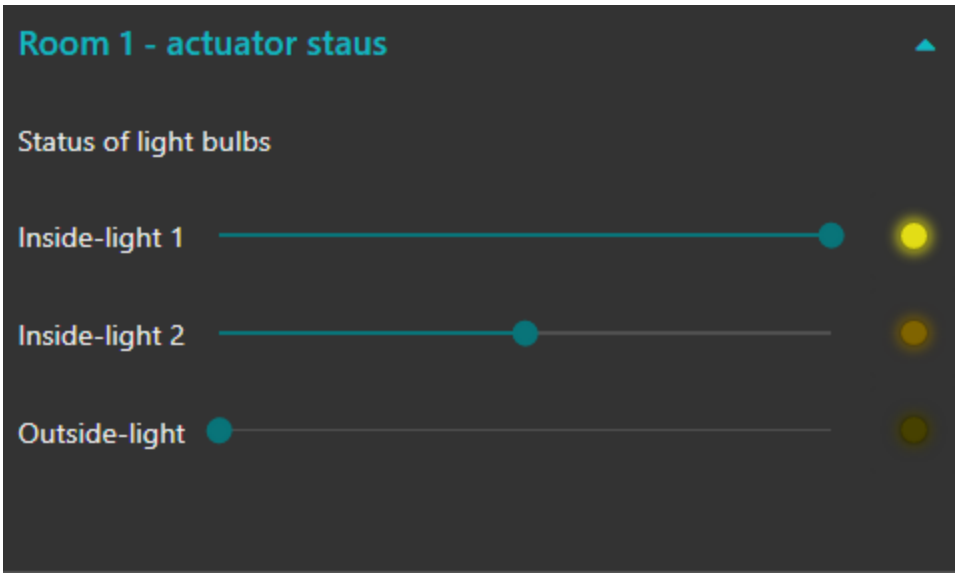
Status of actuators

The actuators used were light bulbs, their status is displayed with regard to brightness of the bulbs using a horizontal line of axis that correspond to the brightness level.

- Inside light bulb -1
- Inside light bulb - 2
- Outside light bulb

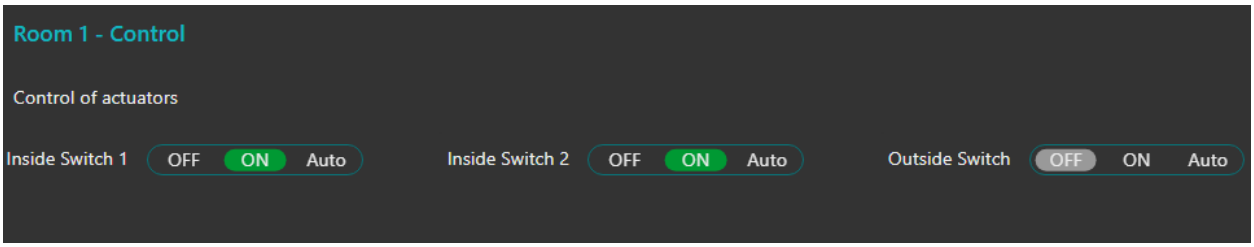
326project/smartbuilding/lighting/<floor_number>/<room_number>/switch/inside/<led_number>/

326project/smartbuilding/lighting/<floor_number>/<room_number>/switch/outside/<led_number>/



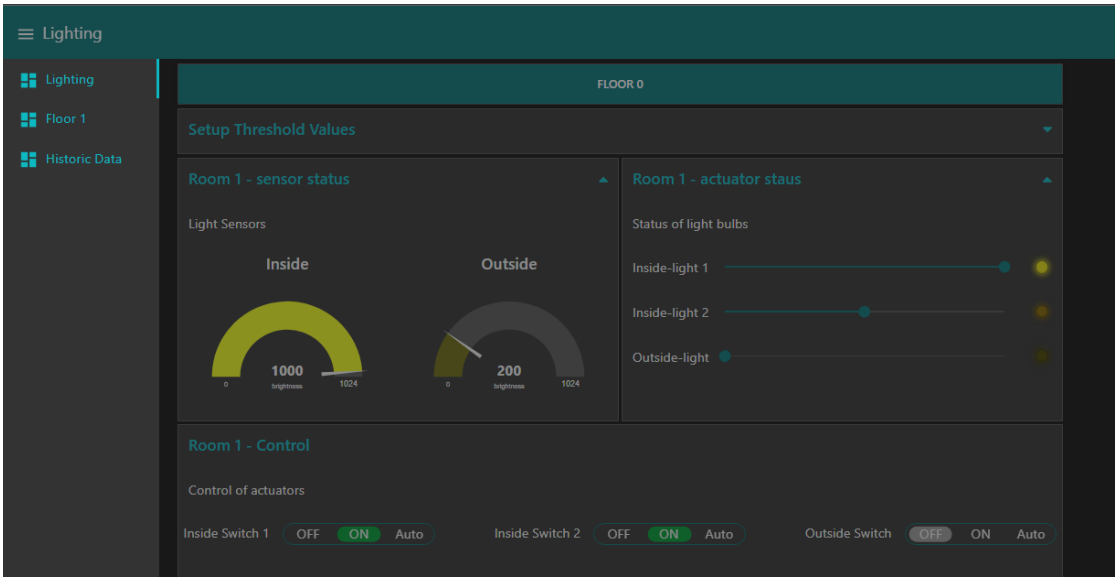
Other than components status, the mode which facilitate the functioning of actuators(state of light bulbs) is also displayed and allowed users to select among,

Manually ON
Manually OFF
AUTO



In addition to that, default threshold values used for process controlling are displayed on the dashboard,

How the status is displayed on node red dashboard



4 - Inputs from the SCADA should send to the MQTT server

SCADA system involves an easy-to-use, integrated development environment for a human–machine interface, database, data acquisition, alarm monitors, logic control etc. When the login status is with guest permission, the SCADA system is only allowed to monitor the system and not allowed to be controlled. As for control operation, it is necessary to be logged in using an authorized account. Then, the user can click on the button of the device control and access the control screen.

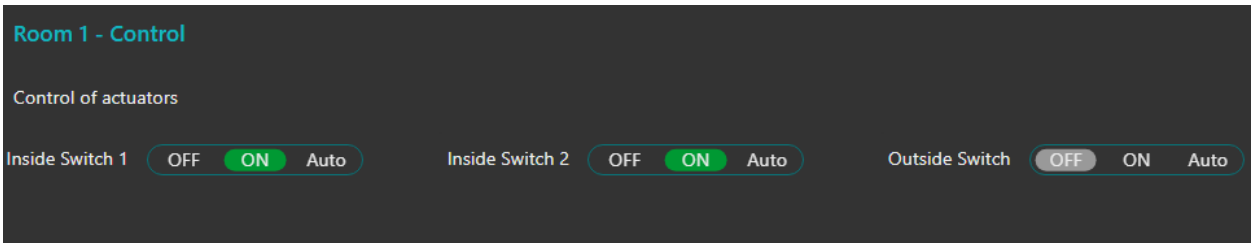
Depending on the sensor readings and other information received by the SCADA, it makes required control decisions to control the lighting. Then those control signals generated from the SCADA(inputs from the SCADA) should be sent to the MQTT server.

Inputs from the SCADA

- Manual ON/OFF signals for Lights.
- Automatic mode selection.
- Increase/Decrease light intensity for dimmable lights.
- Threshold values for LDRs.

In the figure below, it shows how the inputs Manual ON/OFF signals for lights and Automatic mode selection is displayed on the dashboard.

When the user selects ON, the bulbs will turn ON and when selects OFF, the bulbs will be turned OFF. If user Auto, the bulbs will be ON/OFF according to the algorithms provided in process control section (According to the values of LDR sensor data, occupancy based data and time scheduling)



Threshold values for LDRs to identify whether the background is dark or light can also be input by the user as follows.

5 - Process controller with operating and optimizing process and algorithms

In lighting controllers, mainly the following processes happen with following functionalities.

Process → • Turn the lights ON and OFF

• Adjust light output up and down using a dimmer.

Functionality → • Flexibility to satisfy user visual needs

• Automation to reduce energy costs and improve sustainability.

• Adjust light source color, including shade of white light.

• Generate data via measuring and/or monitoring.

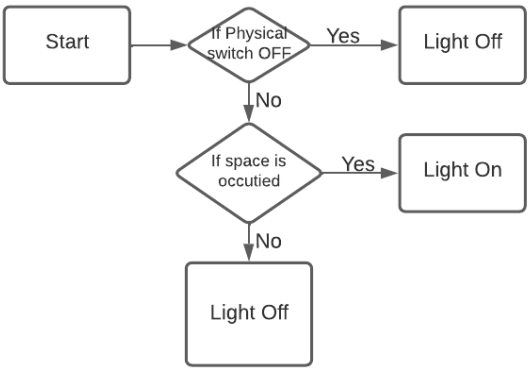
Main Control Strategies

• Manual control

Providing users the capability of choosing light levels either in steps (switching) or over a wide range with smooth transitions between levels (dimming). Switching may be ON/OFF or multilevel.

• Occupancy sensing

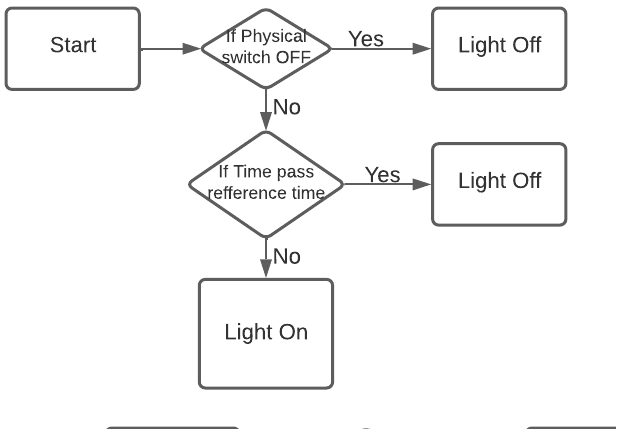
Occupancy sensors are devices that automatically turn the lights ON and OFF based on whether the space is occupied. By ensuring the lights are ON only while the space is occupied.



• Time scheduling

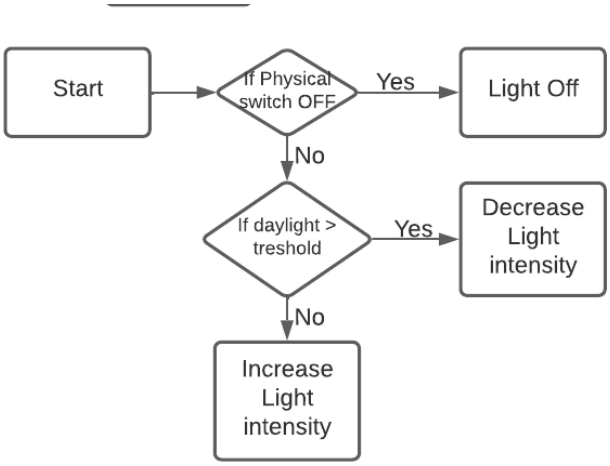
Scheduling adjusts the output of the lighting system based on a time event implemented using a time-clock. At certain times, controlled lights will turn ON, OFF or dim to either

save energy or support changing space functions.



• Daylight response

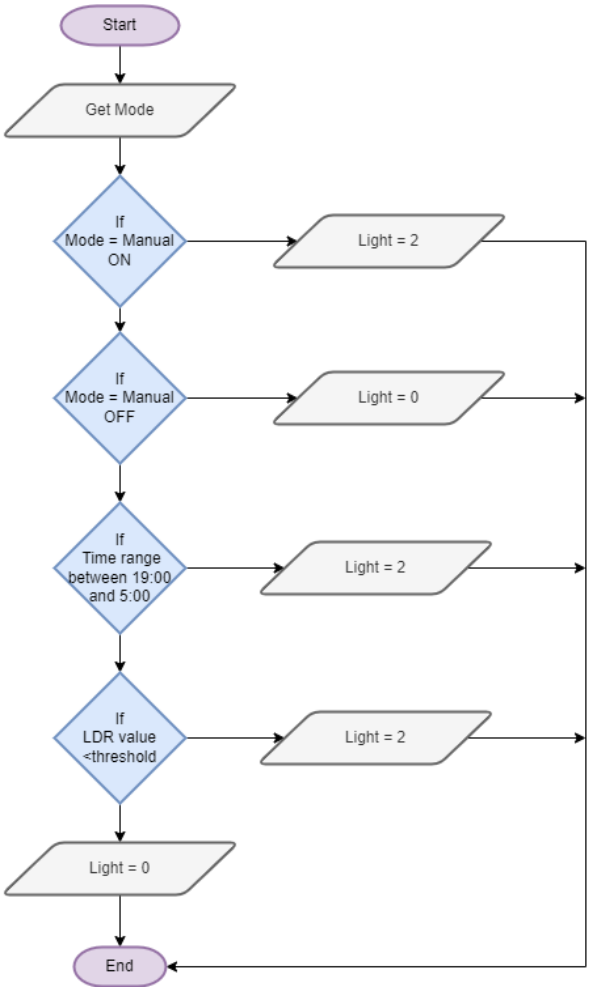
Power controller to switch or dim lighting in response to available daylight. As light levels rise above a target threshold due to daylight contribution, the sensor signals the controller to reduce light output.



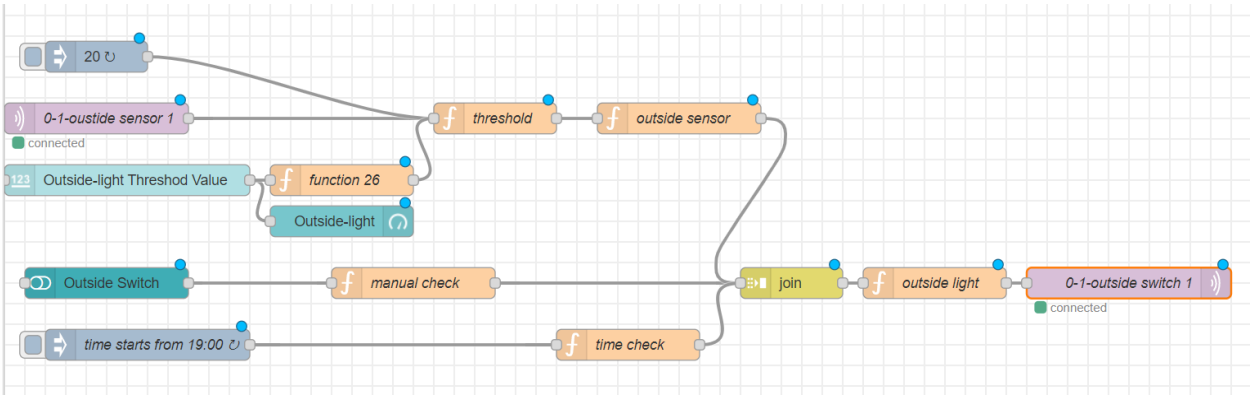
Combining the control mechanisms Manual control, Occupancy sensing, Time scheduling and daylight response from LDRs four types of process control techniques was implemented for different kinds of lights in the building.

1. Considering Manual control, Time scheduling and Daylight response - For outside

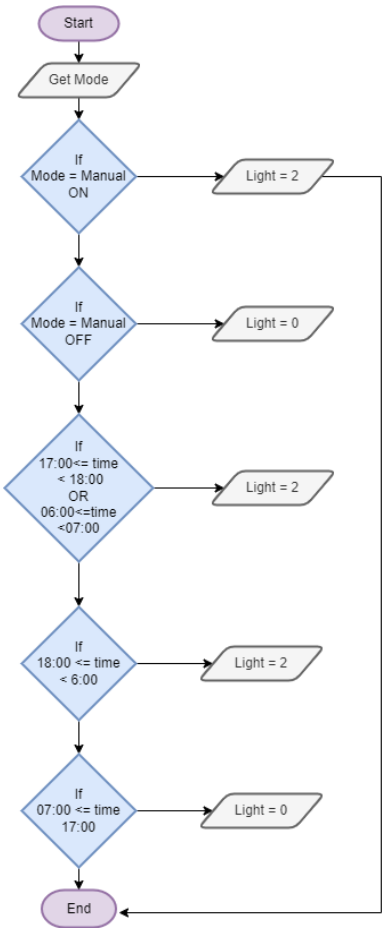
lights.



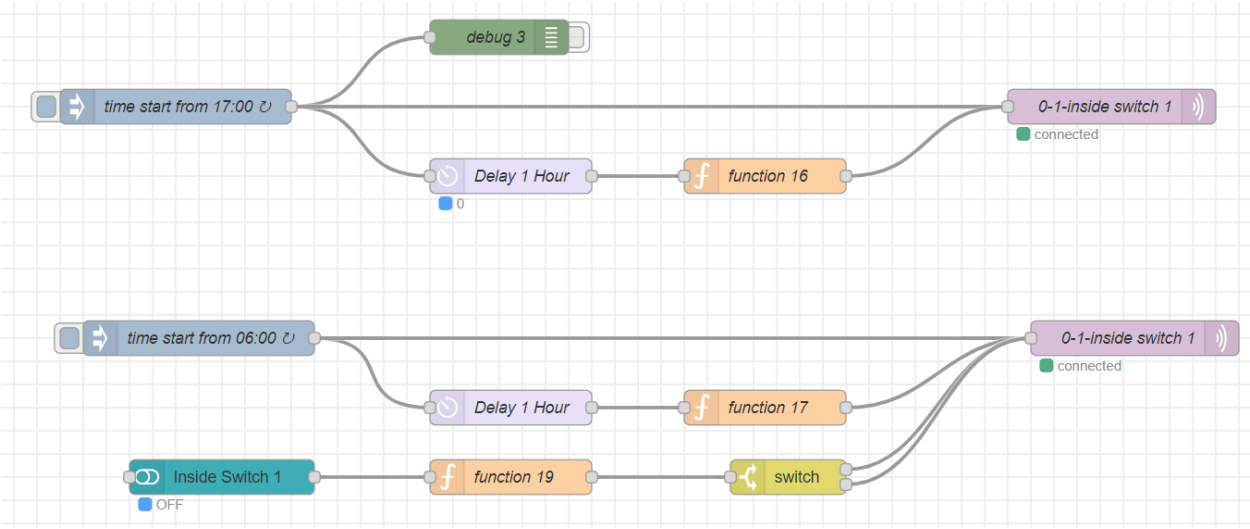
Node-RED implementation



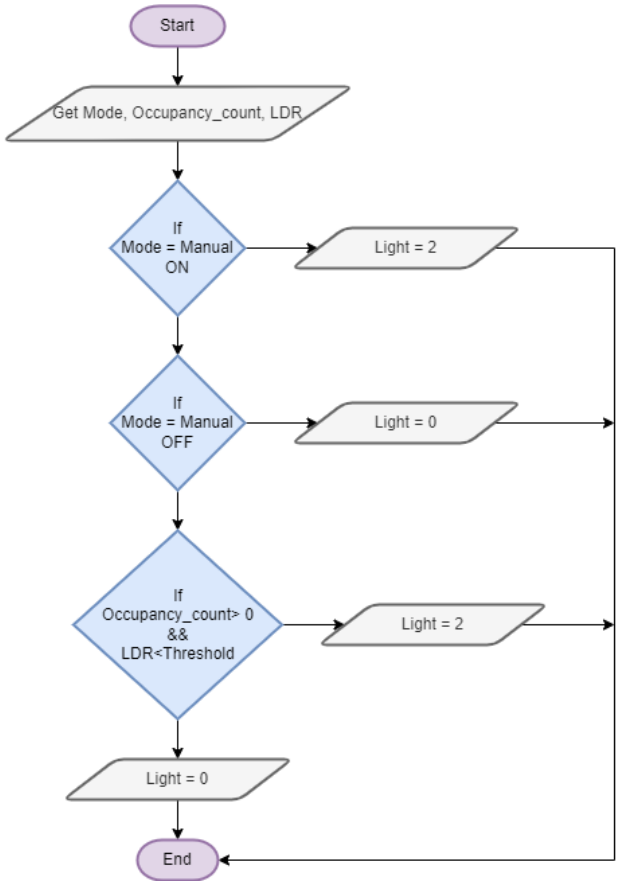
2. Considering Manual control and Time scheduling - For inside lights.



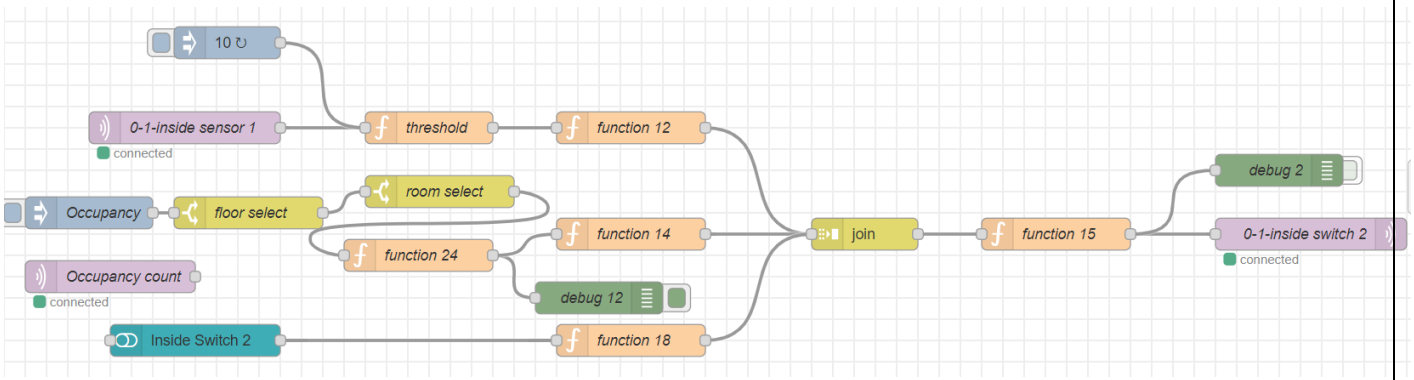
Node-RED implementation



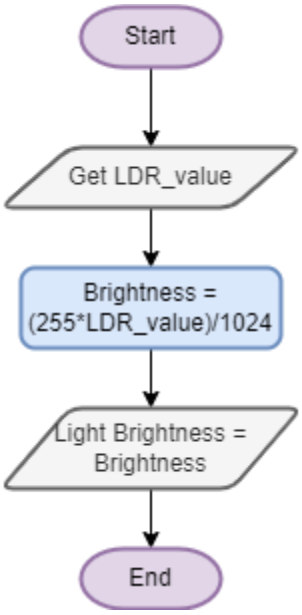
3. Considering Manual control, Occupancy data and Daylight response - For inside lights.



Node-RED implementation

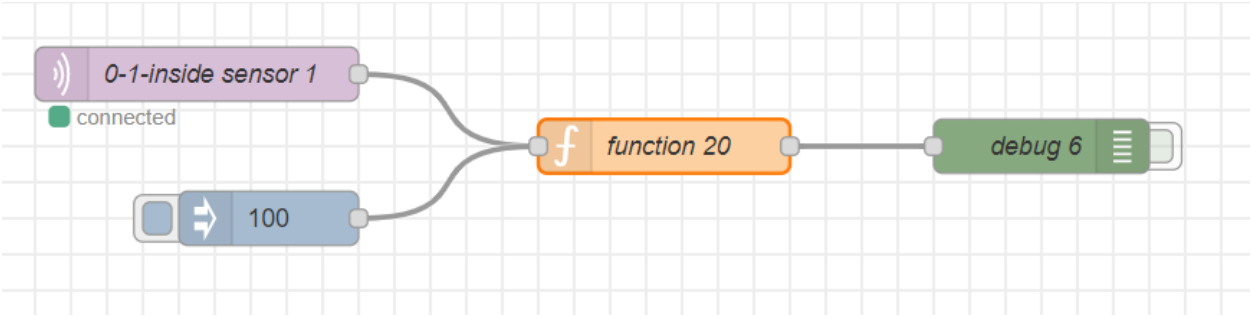


4. Considering Daylight response - For inside dimmable lights.



Here the LDR value is taken from the LDRs mounted outside the building.

Node-RED implementation



6 - MQTT Data, Commands and events should be stored in the database

Introduction :

In this project various kinds of data generation happens at different endpoints. Those data then will be published to a server as the system is interconnected, using MQTT protocol. Based on that data, the control decisions (aka commands) are made in order to automate the system. Therefore, keeping a copy of those generated data (by the sensors and other system end points) would be needed to operate the system with expected functionalities.

The Process :

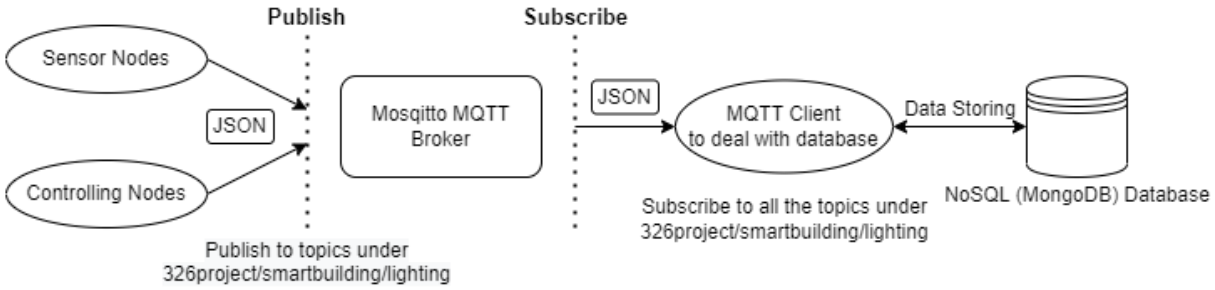
To accomplish this task, the work can be divided into two sub tasks, data retrieving and data storing.

- Data Retrieval:

  Since the use of MQTT protocol as the data exchange protocol is in the middle, it needs to be subscribed to all the necessary topics that are going to be published by the endpoints. For that subscribing purpose, there has to be MQTT client to subscribe all those topics under `326project/smartbuilding/lighting/` in Mosquitto broker that is installed in the Dept server.

- Data Storing:

  To store all the generated data that has been retrieved, the used database is MongoDB. Several different collection has created on the database (MongoDB) to store the relevant messages that are published to the MQTT message broker.



The used data format that is to be exchanged between the nodes is JSON. Therefore, those JSON data will be preprossed before storing into the database.

Before storing those data to the database, first it is needed to clean the received message from the broker and then, system will add the timestamp to the message after that, the data will be stored to the database.

7 - Web interface or SCADA pages should display the data in the database

Introduction :

The data which are stored in the database is visualized in this section.

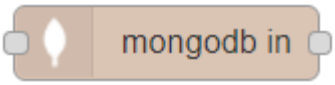
The technology used to display the data in the database is Node RED.

The Node-RED-Dashboard is a module that provides a set of nodes in Node-RED to

create a data dashboard. The MongoDB database is used to store data about lightning. MongoDB node connects to the Mongo DB database to the node RED dashboard.

The Process :

The connection to MongoDB database is done via ‘MongoDB in’ node.



Then to select the relevant data from the database ‘template’ node is used.


```
var floor = "0"
var room = "1"

async function sendSingleObj(payload) {
    var data = []
    for (let i = 0; i < payload.length; i++) {
        var floorTemp = payload[i].floor
        var roomTemp = payload[i].room
        if (floor == floorTemp && room == roomTemp) {
            let dataTemp = payload[i].data
            let time = payload[i].time
            data.push({ "x": time, "y": dataTemp })
        }
    }
    return data;
}
var data = await sendSingleObj(msg.payload)
msg.payload = [{
    "series": ["Control Data reading"],
    "data": [data],
    "labels": [""]
}]
return msg
```

Finally, the data is visualized in the dashboard as charts and tables.

What is shown in the historical visualization?

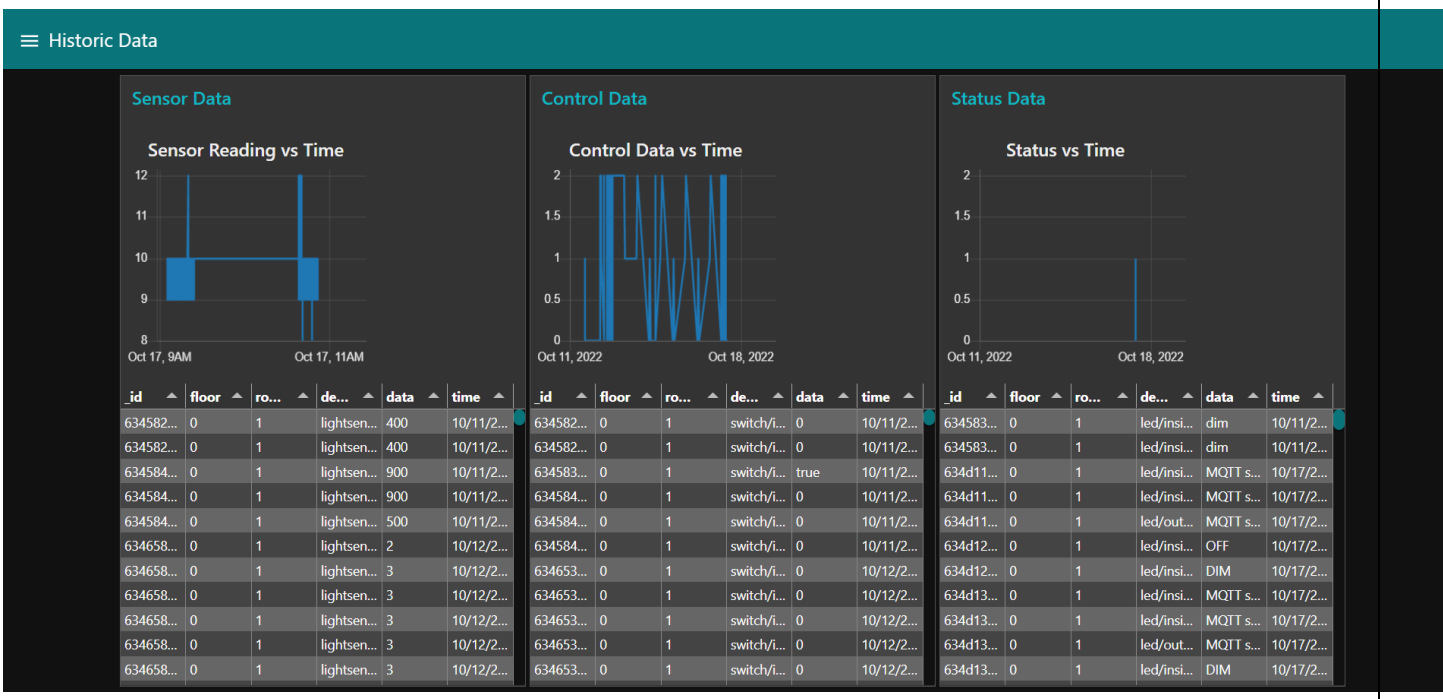
The historic data that are visualized are,,

- Sensor Reading vs Time (LDR Sensor readings)

- Control Data vs Time
    - 0: Manual Off
    - 1: Auto
    - 2: Manual On
- Status Data vs Time
    - Light Intensity
        - 0 : Light Off
        - 1 : light On

(Taken from the database.)

❖ These data are shown in the node red dash board as charts and tables.



8 - Data analytics: Prediction, Optimization, Correlation

Data Analytics

Since a number of data is continuously stored in the database, it can be used for further analytics of data and make it an aid for predictions,optimizations and correlations. When we use those predicted data, it’s not always necessary to rely on sensor data which can be wrong/un reliable sometimes due to several conditions. A machine learning model is used here to analyse data and this will save the unnecessary energy consumption as well as reduce the sensor failures.

From the database, some attributes can be used as data for the model such as:

- ➢ Data collected from LDR sensor
- ➢ Data collected from Ocupancy sensor
- ➢ Control decisions

Here in the first two attributes we need to have the date/time/duration data as well, in addition to the numeric value of the sensor reading.

Predictions:

After populating the database, we can use those data to implement visual data representations and do predictions. Some predictions we can build up are:

- ➢ Normal time/duration in a day where the light intensity is lesser than a particular limit.
- ➢ Identify usually more occupied rooms/apartments in the building.
- ➢ Occupancy sensor data variation in weekdays/weekends.
- ➢ Control decision that might take for a particular sensor reading with respect to past data.

Optimization:

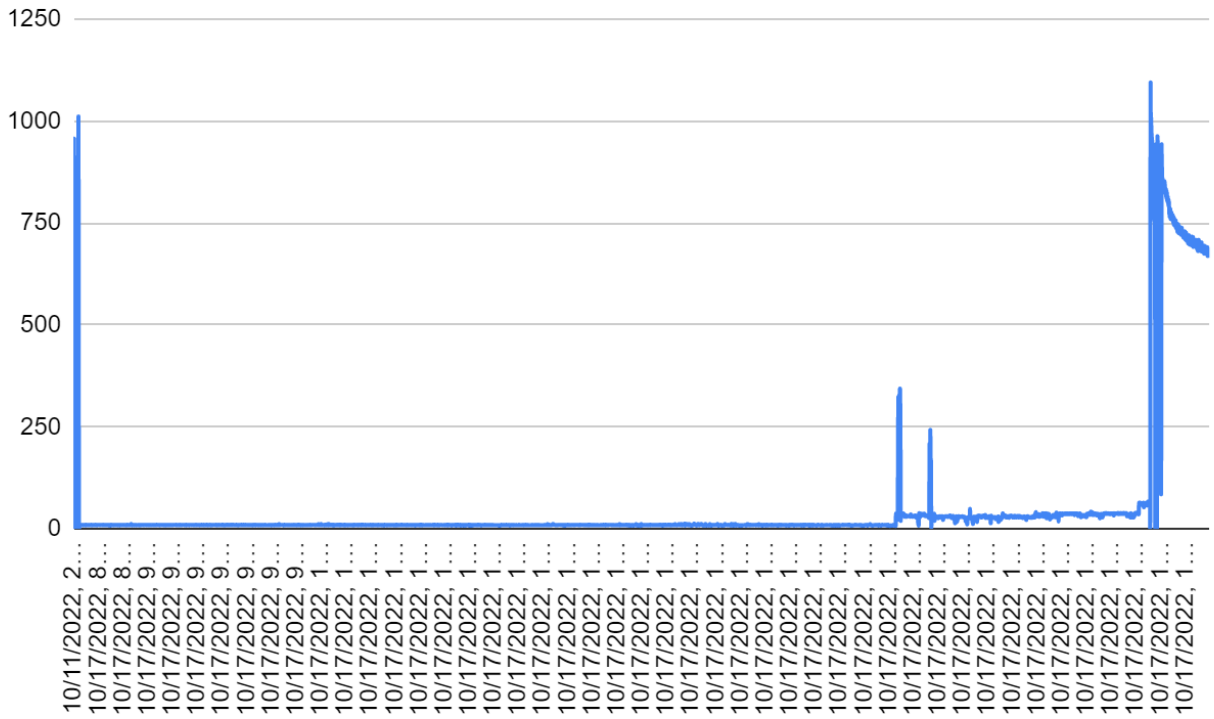
Using the predictions we can optimize the energy consumption of the building in a sustainable manner.

Correlations:

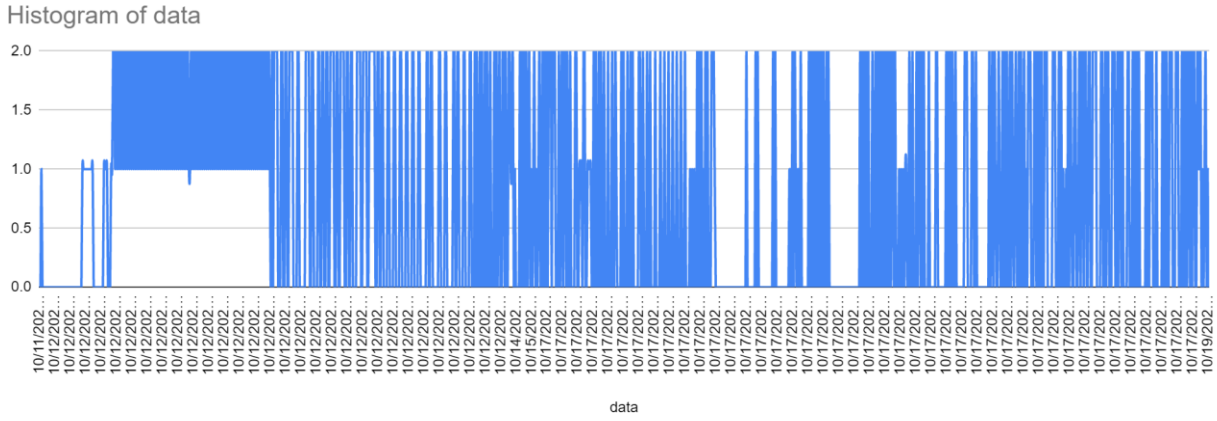
By analyzing

the graphs, we can find correlations between some attributes. This will aid to fill in the missing data points happened to have due to connection/server issues or sensor failures. By this, we can further increase the efficiency of the model.

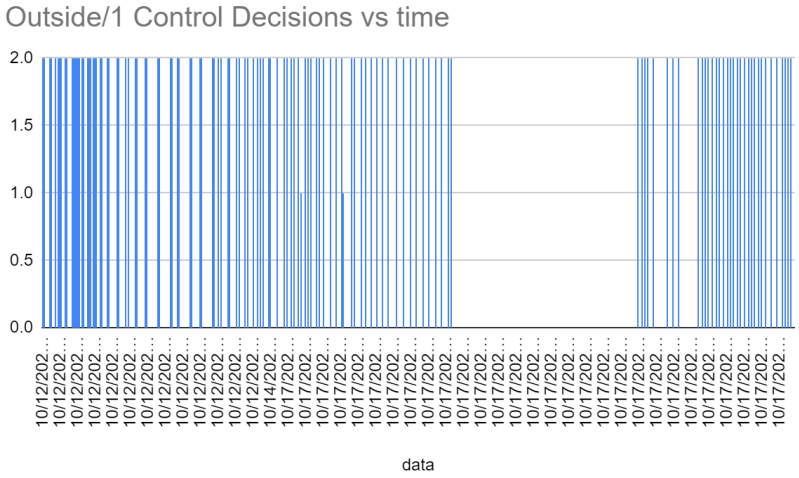
Graphs Obtained

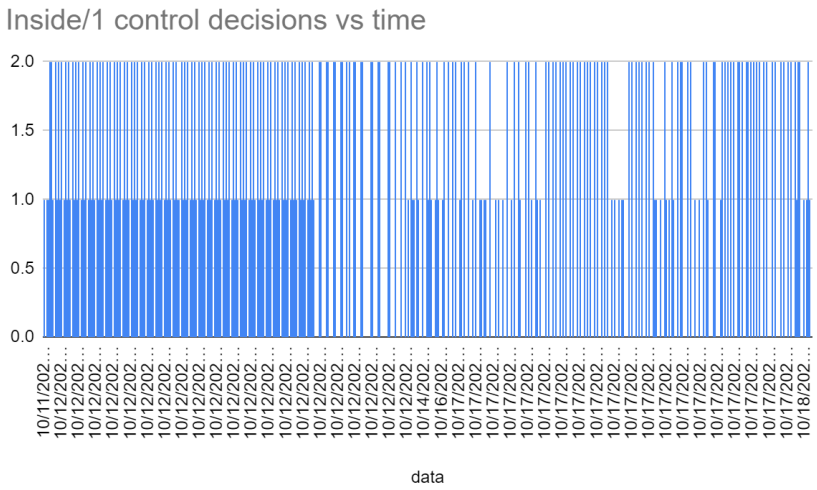


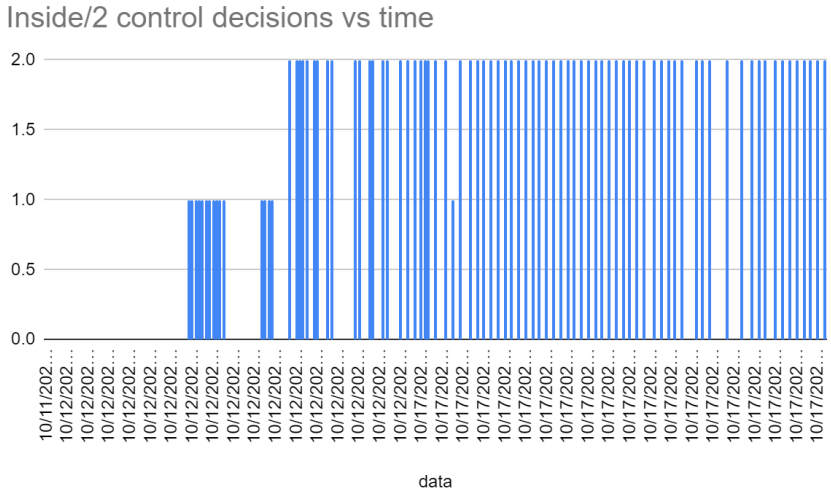
LDR sensor reading variation with respect to time for a dummy data set is shown in the above graph. By analysing this, it can be found an approximate threshold time where the LDR sensor reading is above some value. This will be used for future predictions after checking and validating for reasonable amount of data.



Above graph shows the control decision made (0,1 or 2) with respect to the time series. From this it can be predicted usual time slots that the light is needed to be ON and timeslots that usually don't need to turn ON the lights.







By analysing above graphs it can be observed the time periods where the inside/outside lights are ON and OFF.