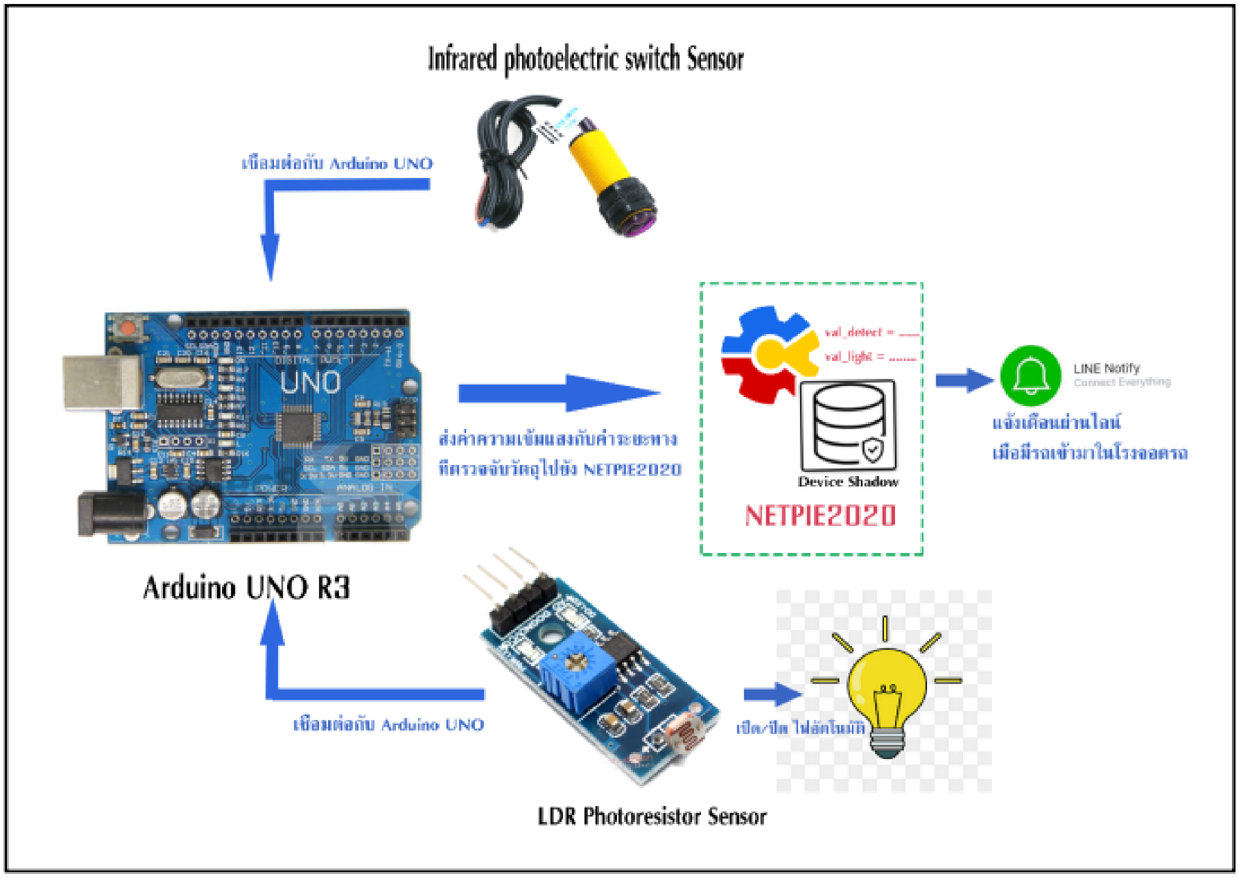
Smart Garage System Using Arduino UNO Connected to the NETPIE2020

Project Description

This example demonstrates how to implement a smart garage system using Arduino UNO and NETPIE2020. A Infrared photoelectric switch sensor which detects the object within its range and a LDR photoresistor sensor which measures the light intensity are connected to the Arduino UNO and their values are pushed to the NETPIE2020. Users get the notification via LINE notify when a car approaches the garage and parking lights get turned on/off.



Working model

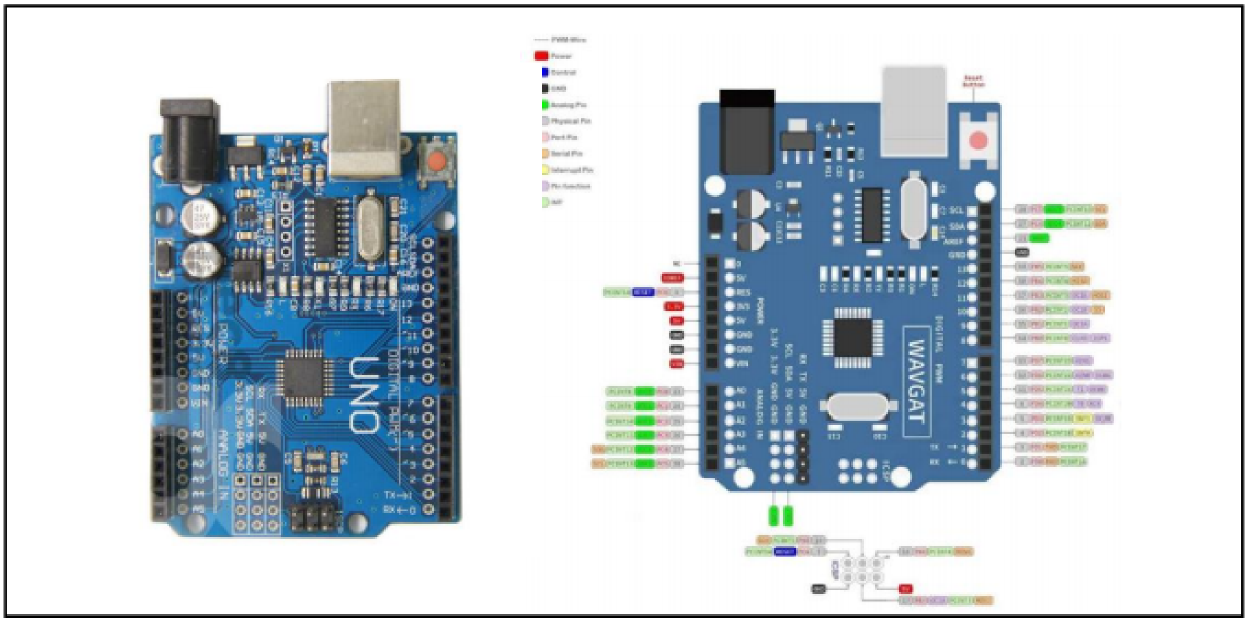
Require Basic Knowledge of

1.Using NETPIE2020
2.Using NETPIE Freeboard

Equipment Used

1.Arduino UNO

Arduino Uno is the most popular board among the Arduino family, because it's easy to learn. The Uno board comes with different versions such as R2 and R3 depending on the type of the microcontroller used. The code can be uploaded to it via usb and can control various sensors.



Arduino UNO pinout

2.LDR Photoresistor Sensor

It is used to detect the intensity of the light. It changes its resistance when the light strikes on it. The resistance change depends on the intensity of the light. The signal is both analog and digital.

Pin Description

Pin1 : Power +Ve (5V Max wrt. GND)
Pin2 : Power Ground or Power –Ve
Pin3 : Digital I/O
Pin4 : Analog I/O

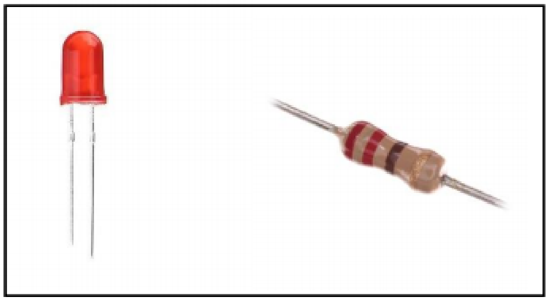
3.Infrared Photoelectric Switch Sensor

It is used to detect the absence or presence of an object based on reflection of infrared waves. The distances at which the sensor can detect the objects can be set by adjusting the potentiometer.

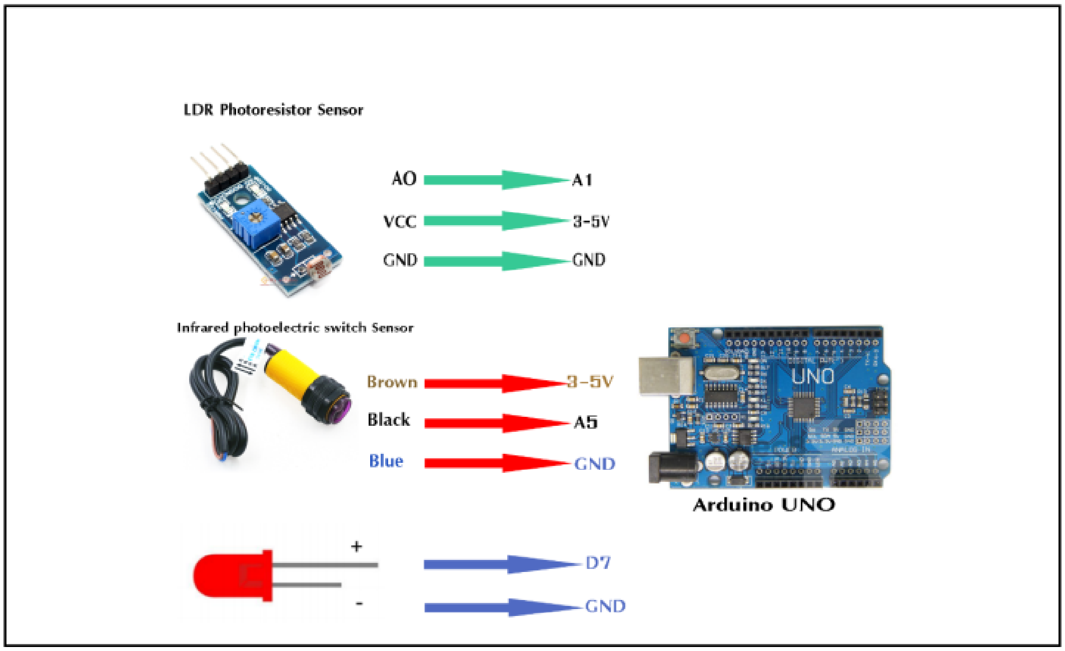
Pin Description

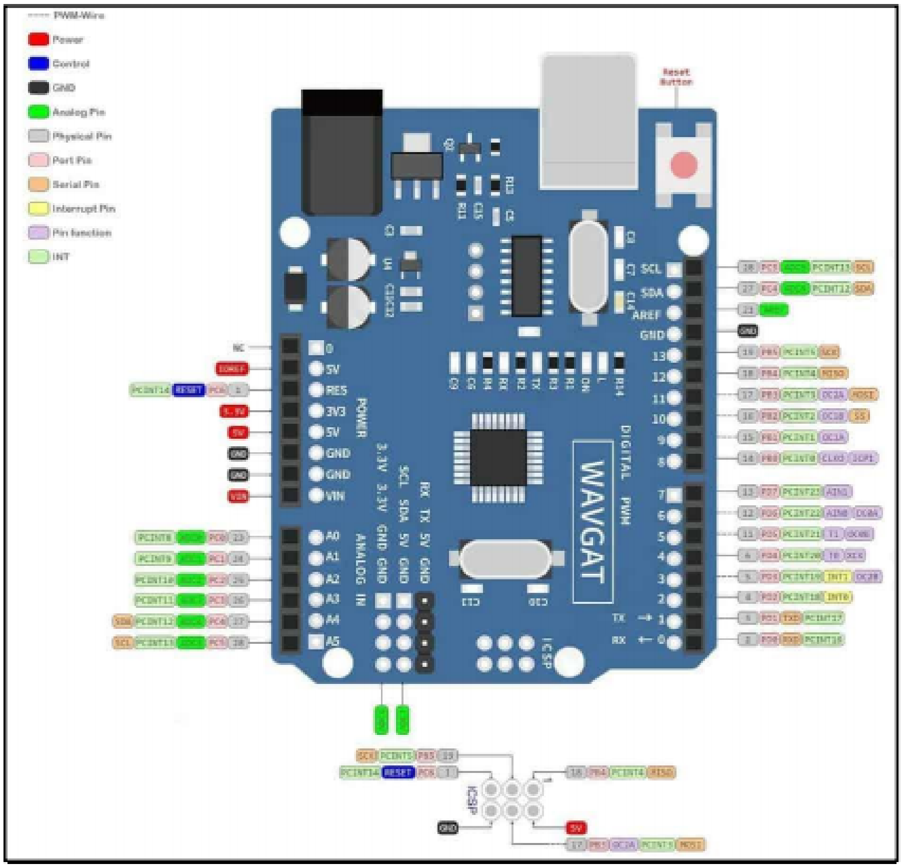
Pin1(brown) : Power +Ve (5V Max wrt. GND)
Pin2(Black) : Digital I/O interface connected to microcontroller
Pin3(Blue) : Power Ground or Power –Ve

4.LED and 220 Ohm Resistor



Arduino UNO Circuit Connection

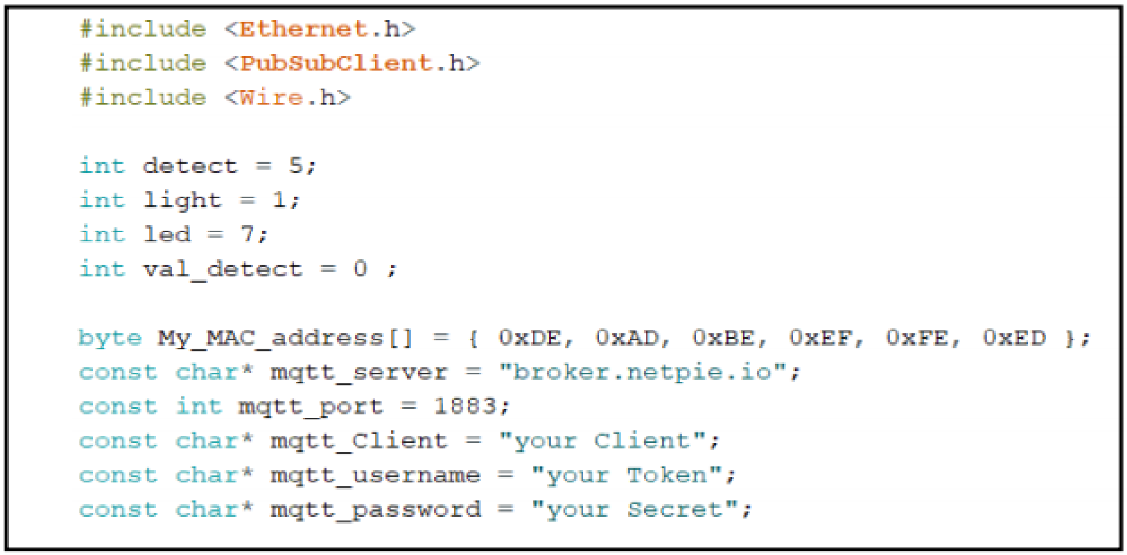




Arduino UNO pinout

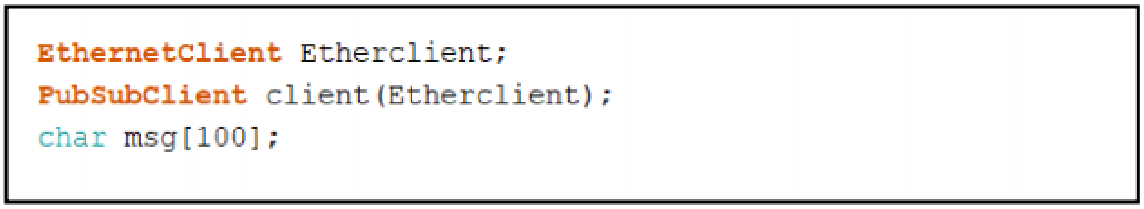
Description of Arduino Program

First, the required libraries for executing the program should be imported, followed by defining the program parameters like LED PIN. Next, configure the network parameters like server, port, client ID, username, and password for connecting to the NETPIE2020.



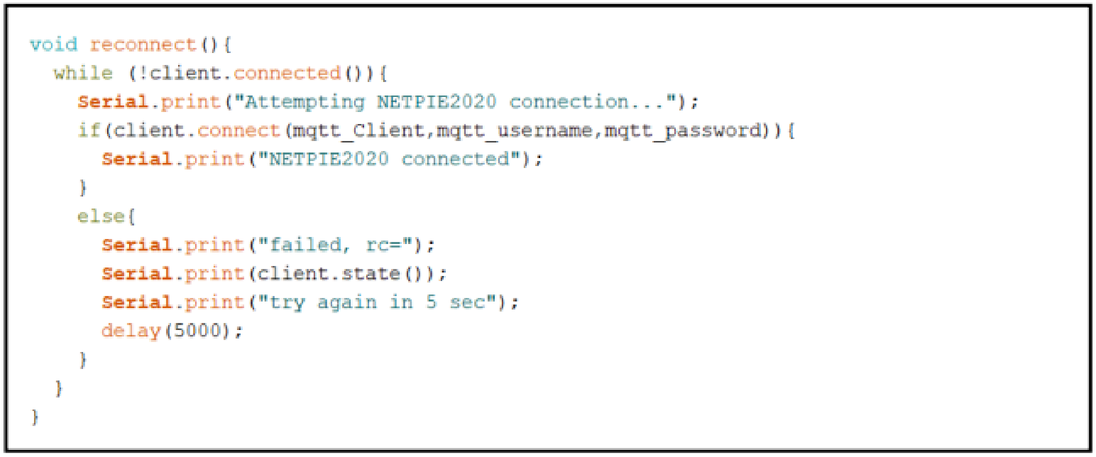
```
#include <Ethernet.h>
#include <PubSubClient.h>
#include <Wire.h>

int detect = 5;
int light = 1;
int led = 7;
int val_detect = 0 ;

byte My_MAC_address[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
const char* mqtt_server = "broker.netpie.io";
const int mqtt_port = 1883;
const char* mqtt_Client = "your Client";
const char* mqtt_username = "your Token";
const char* mqtt_password = "your Secret";
```

The second part is to run the MQTT commands.



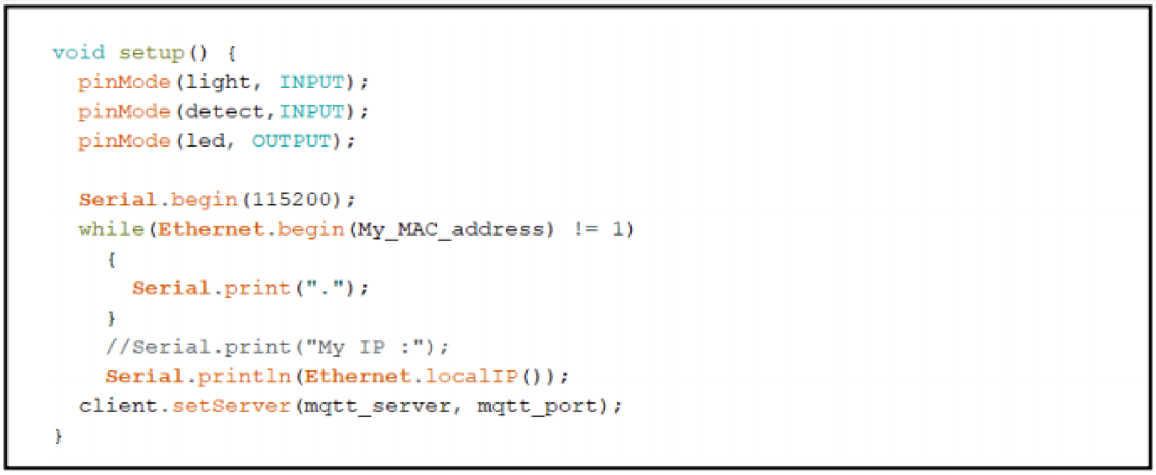
```
EthernetClient Etherclient;
PubSubClient client(Etherclient);
char msg[100];
```

The MQTT connect function is used to connect to the MQTT server. If the connection is successful, it will display the message saying ‘connected’. But, if the connection is unsuccessful, ‘failed’ message is displayed and will try to reconnect automatically.



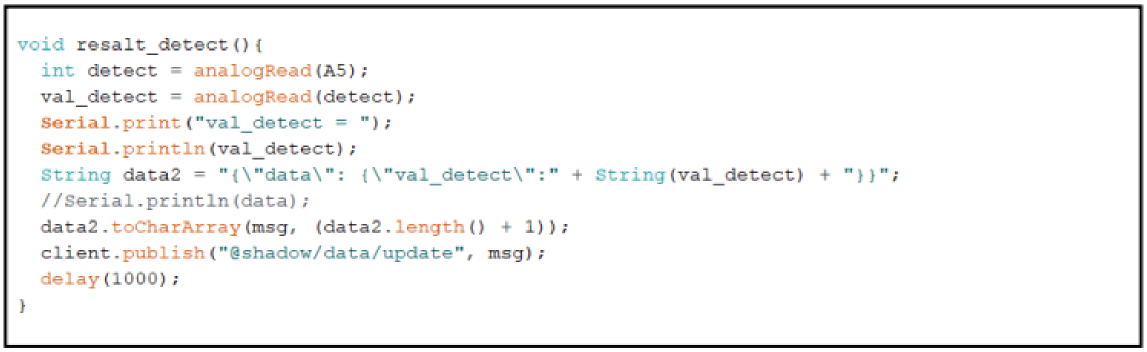
```
void reconnect(){
  while (!client.connected()){
    Serial.print("Attempting NETPIE2020 connection...");
    if(client.connect(mqtt_Client,mqtt_username,mqtt_password)){
      Serial.print("NETPIE2020 connected");
    }
    else{
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.print("try again in 5 sec");
      delay(5000);
    }
  }
}
```

The setup function attempts to use the internet by running the command Ethernet.begin command. Assign the pin configurations where light intensity and distances are given as INPUT and LED is configured as OUTPUT.



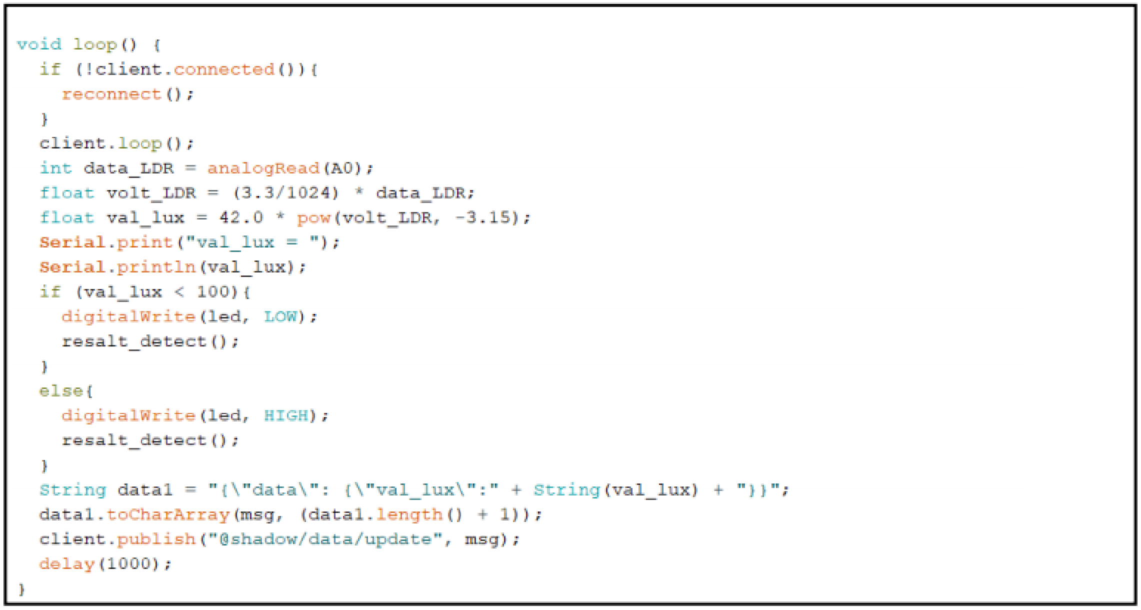
```
void setup() {
  pinMode(light, INPUT);
  pinMode(detect,INPUT);
  pinMode(led, OUTPUT);

  Serial.begin(115200);
  while(Ethernet.begin(My_MAC_address) != 1)
    {
      Serial.print(".");
    }
    //Serial.print("My IP :");
    Serial.println(Ethernet.localIP());
  client.setServer(mqtt_server, mqtt_port);
}
```

The infrared photoelectric switch sensor functionality is defined in the resalt_detect function. Whenever the object is detected by the sensor it will publish the data on the Topic: @shadow/data/update for every 2 seconds to update the val_detect values in Shadow.



```
void resalt_detect(){
  int detect = analogRead(A5);
  val_detect = analogRead(detect);
  Serial.print("val_detect = ");
  Serial.println(val_detect);
  String data2 = "{\"data\": {\"val_detect\":" + String(val_detect) + "}}";
  //Serial.println(data);
  data2.toCharArray(msg, (data2.length() + 1));
  client.publish("@shadow/data/update", msg);
  delay(1000);
}
```

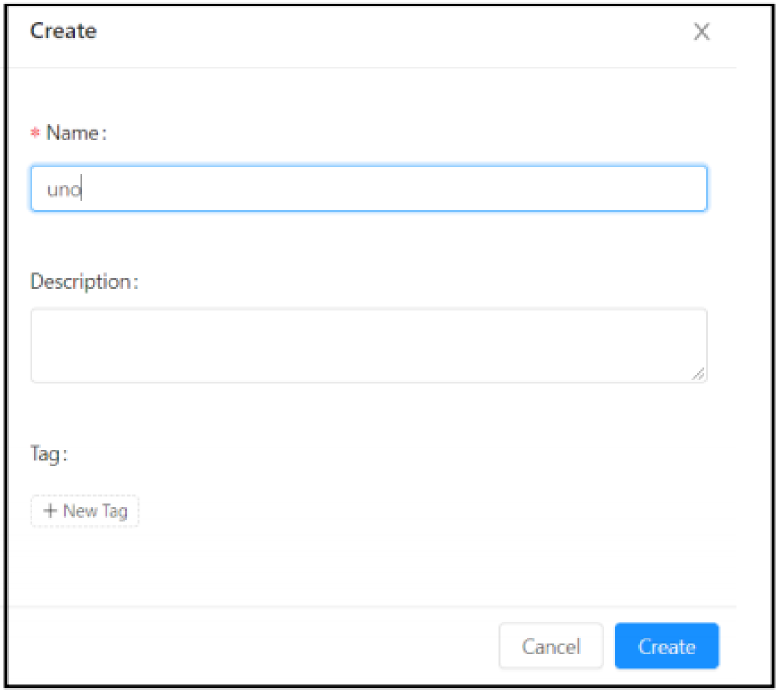
The last part is the loop function. The LDR sensor functionality is defined in this part. If the light intensity is found to be less than 100 Lux then the LED is turned off. But, if the light intensity is greater than 100 Lux the led is turned on automatically. The data is published on the Topic: @shadow/data/update for every 2 seconds to update the val_lux values in the Shadow.



```
void loop() {
  if (!client.connected()){
    reconnect();
  }
  client.loop();
  int data_LDR = analogRead(A0);
  float volt_LDR = (3.3/1024) * data_LDR;
  float val_lux = 42.0 * pow(volt_LDR, -3.15);
  Serial.print("val_lux = ");
  Serial.println(val_lux);
  if (val_lux < 100){
    digitalWrite(led, LOW);
    resalt_detect();
  }
  else{
    digitalWrite(led, HIGH);
    resalt_detect();
  }
  String data1 = "{\"data\": {\"val_lux\":" + String(val_lux) + "}}";
  data1.toCharArray(msg, (data1.length() + 1));
  client.publish("@shadow/data/update", msg);
  delay(1000);
}
```

Creating a device on NETPIE2020

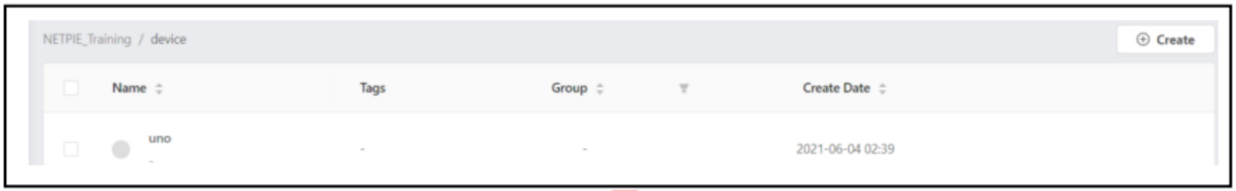
1.Start by selecting the menu Device List > Create and name the device. You can then use the ClientID, Token, and Secret of the newly created device.



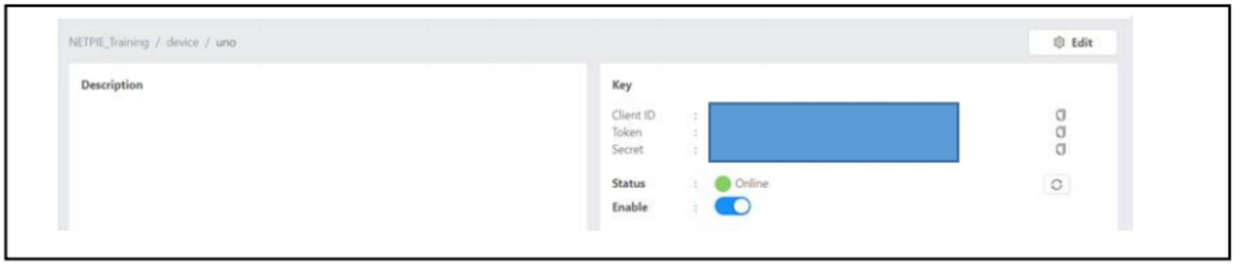
* Name : uno

Description :

Tag :



	Name	Tags	Group	Create Date
□	uno -	-	-	2021-06-04 02:39



Description

Key

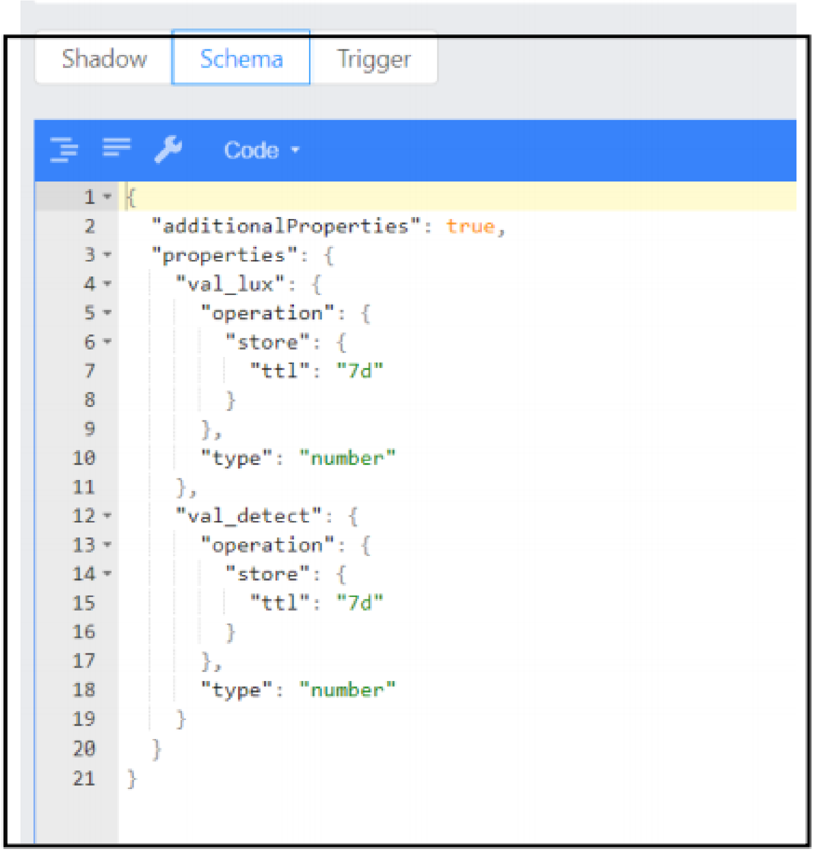
Client ID :
Token :
Secret : [redacted]

Status : ● Online

Enable :

Defining Device Schema on NETPIE2020

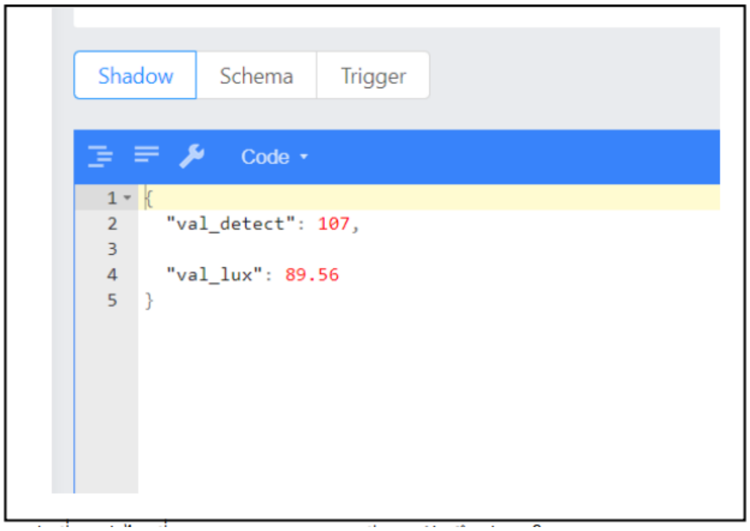
Device Schema is the data structure for the devices generating data. The server checks the data structure defined in the device schema and performs the required actions before storing the data in the Timeseries database. These actions include converting the data units and data validation.



```
{
  "additionalProperties": true,
  "properties": {
    "val_lux": {
      "operation": {
        "store": {
          "ttl": "7d"
        }
      },
      "type": "number"
    },
    "val_detect": {
      "operation": {
        "store": {
          "ttl": "7d"
        }
      },
      "type": "number"
    }
  }
}
```

Description of Schema

Stores the first variable val_light of type number in the Timeseries database for a retention period of 7 days.
Stores the second variable val_detect of type number in the Timeseries database for a retention period of 7 days.



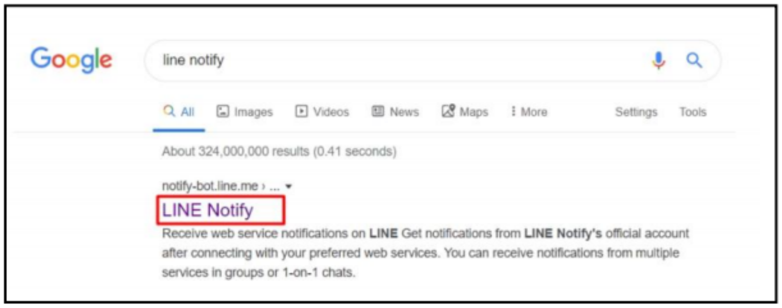
```
{
  "val_detect": 107,

  "val_lux": 89.56
}
```

Values sent to the NETPIE2020 and are stored in Shadow

<u>Connecting to the LINE Notify</u>

Connect to the LINE by using Line Token and send the data of interest.

1.Search for LINE Notify



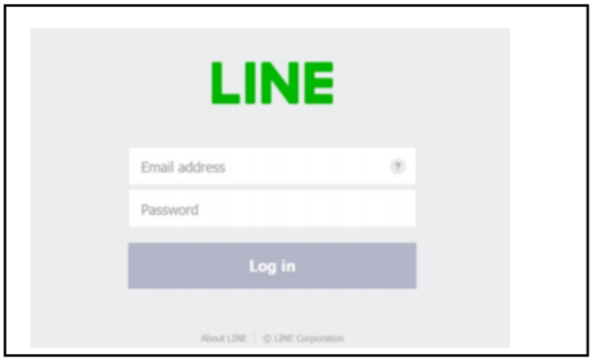
About 324,000,000 results (0.41 seconds)

notify-bot.line.me › ... ▾

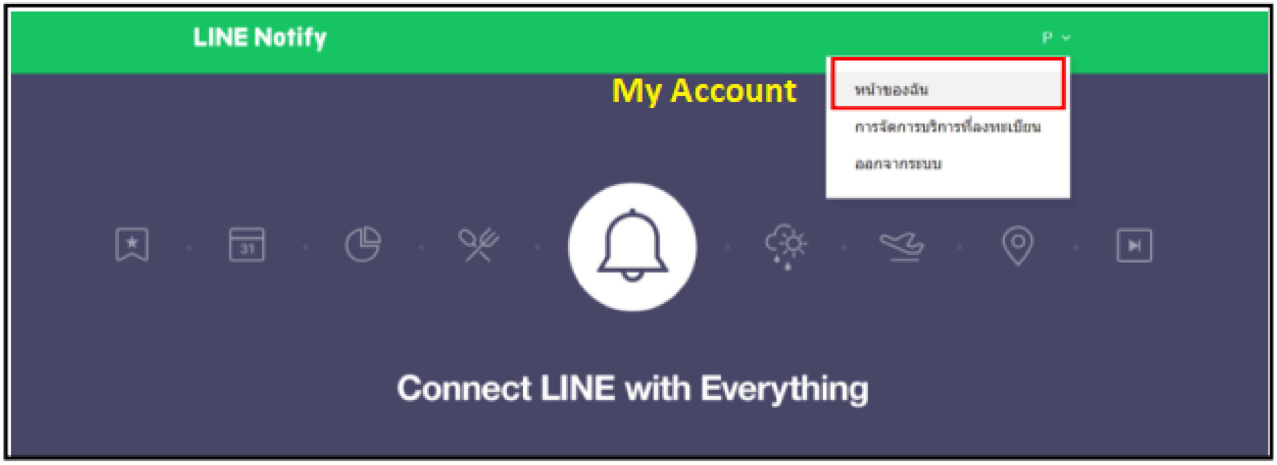
LINE Notify

Receive web service notifications on **LINE** Get notifications from **LINE Notify's** official account after connecting with your preferred web services. You can receive notifications from multiple services in groups or 1-on-1 chats.

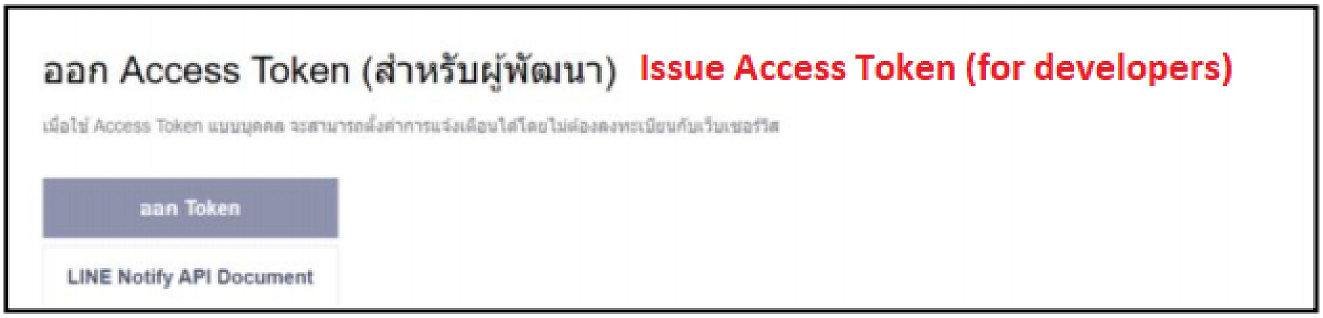
2.LINE login



3.Click on the Mypage



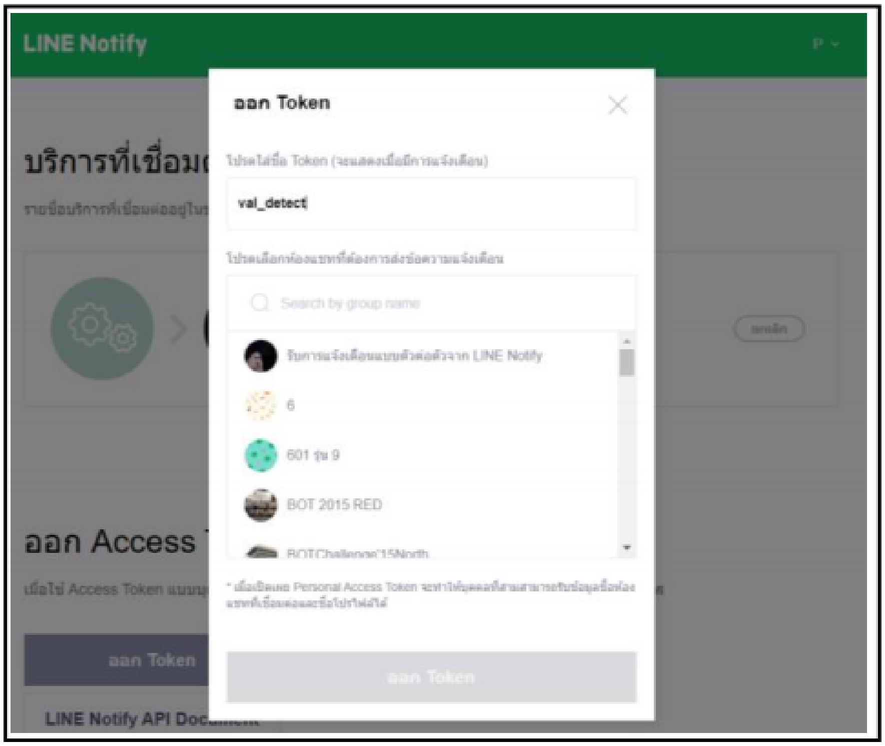
4.Click on Token generation



ออก Access Token (สำหรับผู้พัฒนา) Issue Access Token (for developers)

เมื่อใช้ Access Token แบบบุคคล จะสามารถตั้งค่าการแจ้งเตือนได้โดยไม่ต้องลงทะเบียนกับเว็บเซอร์วิส

5.Enter the message you want to send as default and select the account to which the message needs to be sent.



LINE Notify

บริการที่เชื่อม

รายชื่อบริการที่เชื่อมต่ออยู่ใน

ยกเลิก

ออก Access

เมื่อใช้ Access Token แบบบุ

ออก Token

LINE Notify API Doc

ออก Token

โปรดใส่ชื่อ Token (จะแสดงเมื่อมีการแจ้งเตือน)

val_detect

โปรดเลือกห้องแชทที่ต้องการส่งข้อความแจ้งเตือน

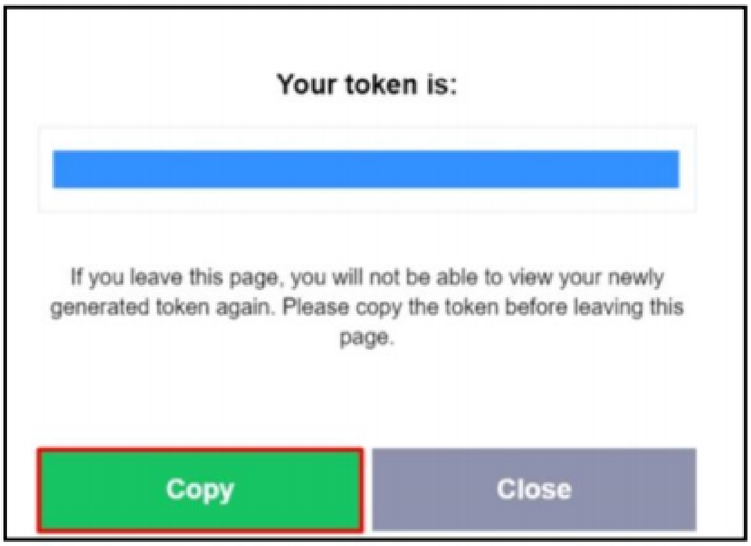
Search by group name

- รับการแจ้งเตือนแบบตัวต่อตัวจาก LINE Notify
- 6
- 601 รุ่น 9
- BOT 2015 RED
- BOTChallenge'15North

* เมื่อเปิดเผย Personal Access Token จะทำให้บุคคลที่สามสามารถรับข้อมูลชื่อห้องแชทที่เชื่อมต่อและชื่อโปรไฟล์ได้

ออก Token

6.Copy the Token code and paste it in the Trigger code section.



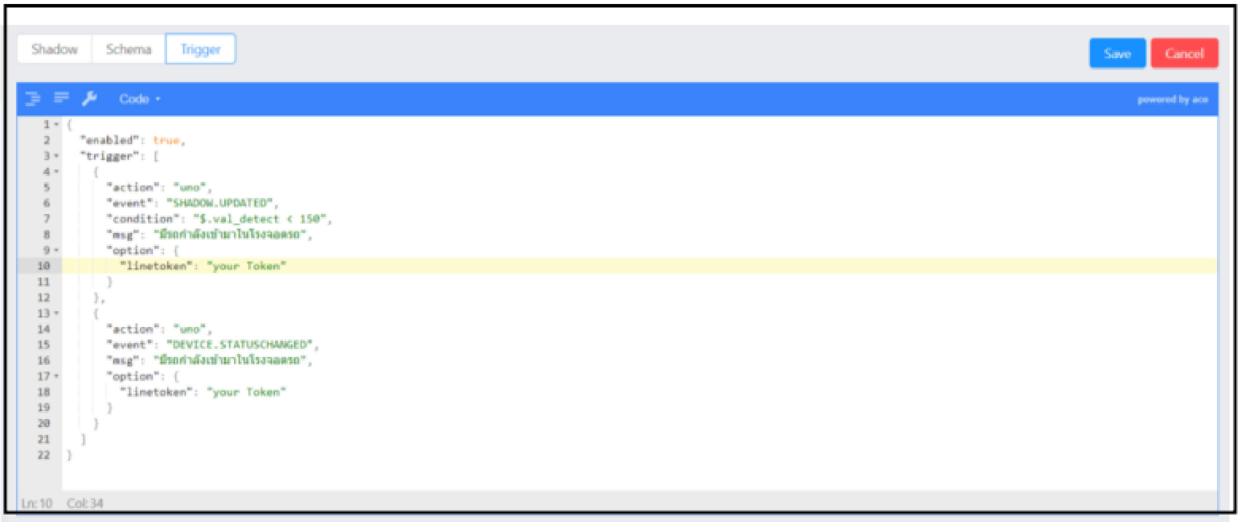
Your token is:

If you leave this page, you will not be able to view your newly generated token again. Please copy the token before leaving this page.

Copy

Close

7.Enter the code in the Trigger section.



```
{
  "enabled": true,
  "trigger": [
    {
      "action": "uno",
      "event": "SHADOW.UPDATED",
      "condition": "$.val_detect < 150",
      "msg": "มีรถกำลังเข้ามาในโรงจอดรถ",
      "option": {
        "linetoken": "your Token"
      }
    },
    {
      "action": "uno",
      "event": "DEVICE.STATUSCHANGED",
      "msg": "มีรถกำลังเข้ามาในโรงจอดรถ",
      "option": {
        "linetoken": "your Token"
      }
    }
  ]
}
```

Code in Trigger

"enabled": true connects to Line to update the desired value.

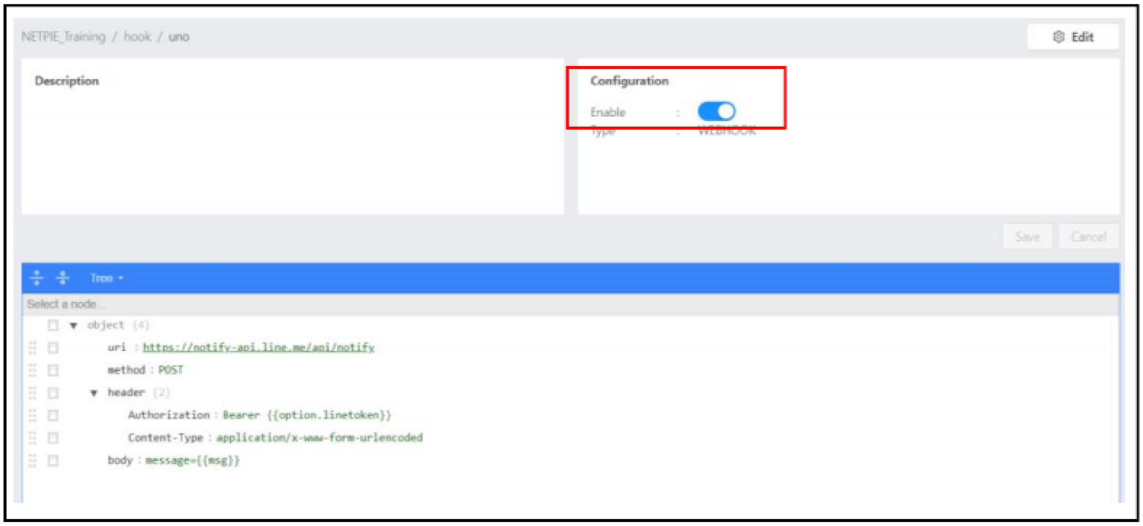
"condition": "$.val_detect < 150" defines a condition to be notified when an object is detected. Yes(Sensor detects at approx. 3-106)

"msg": "A car is entering the garage."

"action": "Name of Event Hook ", "event": "DEVICE.STATUSCHANGED",

"msg": "A car is entering the garage.”

8. Enable the Webhooks to receive the messages.

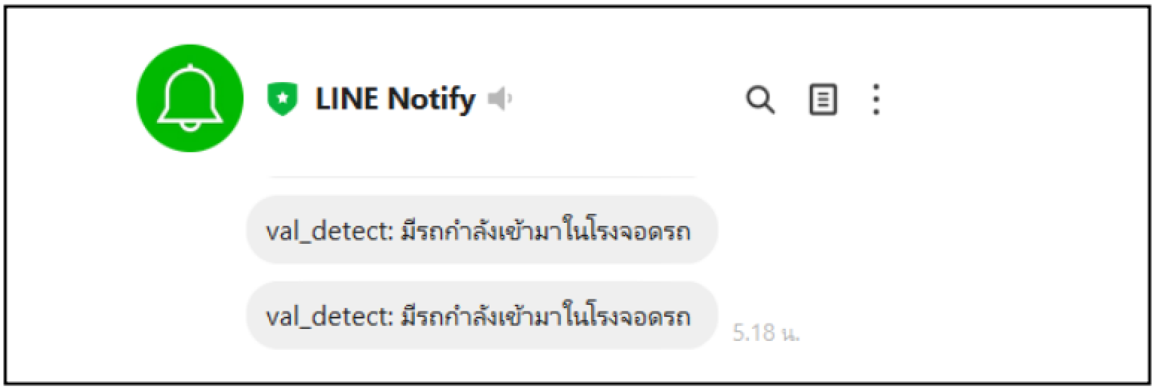


Enable : ●

Type : WEBHOOK

```
object (4)
    uri : https://notify-api.line.me/api/notify
    method : POST
    header (2)
        Authorization : Bearer {{option.linetoken}}
        Content-Type : application/x-www-form-urlencoded
    body : message={{msg}}
```

A notification will be sent to the LINE Notify if the value is greater than the defined value.



val_detect: มีรถกำลังเข้ามาในโรงจอดรถ

val_detect: มีรถกำลังเข้ามาในโรงจอดรถ

5.18 น.