捷運驗票閘門

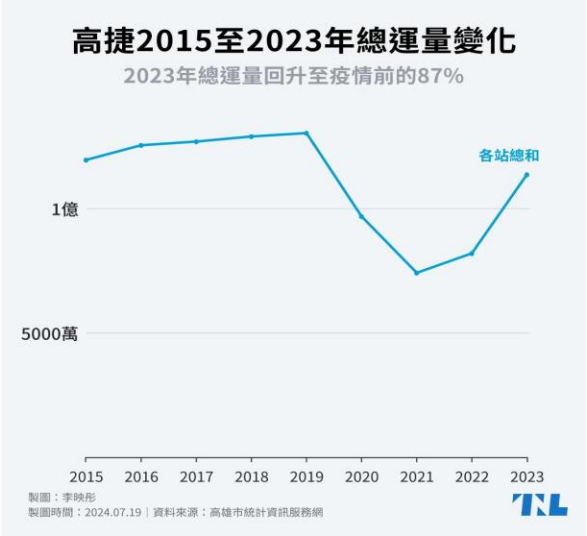
第十一組

11103021A 黃敬霖
11103026A 嚴偉綸
11103104A 林睿宏
11103110A 沈　翔

目錄

一、專案介紹

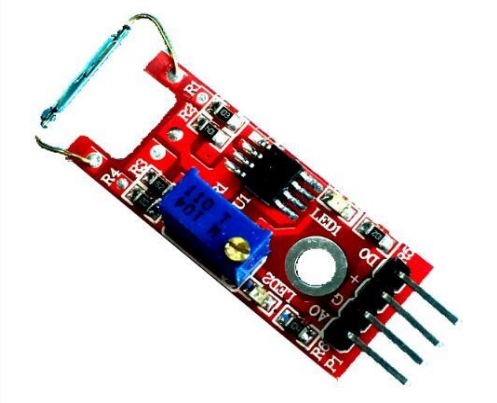
本專案將模擬捷運閘門系統，前端具備感應判斷開關門、強闖偵測、資訊顯示與提示音播放功能，後端則使用MQTT與Node-Red來模擬監控資料庫與實現遠端開關門之功能。



二、使用物件

ESP32

RC522 IC卡感應模組



KY-025 磁簧開關模組

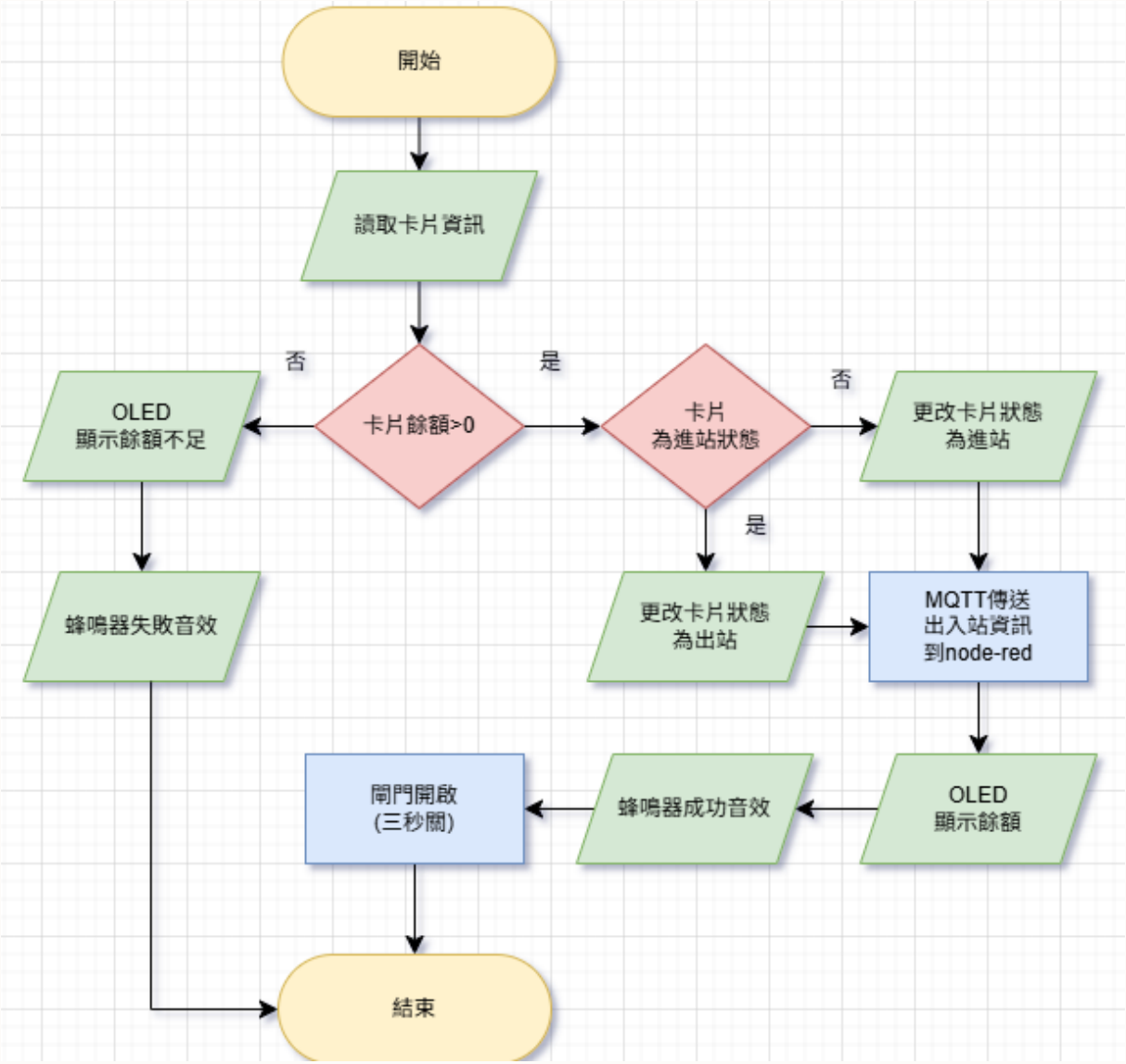
伺服馬達(閘門)

LED(指示燈)

蜂鳴器(音效)

OLED顯示模組

三、閘門系統


開始
讀取卡片資訊
卡片餘額>0
否
是
OLED
顯示餘額不足
卡片
為進站狀態
否
是
更改卡片狀態
為進站
蜂鳴器失敗音效
更改卡片狀態
為出站
MQTT傳送
出入站資訊
到node-red
閘門開啟
(三秒關)
蜂鳴器成功音效
OLED
顯示餘額
結束

三、閘門系統

內部空間：1KB

16 部門(sectors)每個4Blocks

1Block可存16Bytes

第 0 Block：UID跟一些開發資訊

部門最後Block儲存金鑰或密碼，

它控制磁區中其餘區塊的存取

-RC522 IC卡感應模組

PICC type: MIFARE 1KB

Sector	Block	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
15	63	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF
	62	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	61	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
14	59	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF
	58	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	57	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	56	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	...																
3	15	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF
	14	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	13	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	12	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2	11	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF
	10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	9	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	8	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1	7	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF
	6	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	5	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	4	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0	3	00	00	00	00	00	00	FF	07	80	69	FF	FF	FF	FF	FF	FF
	2	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	1	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	0	82	72	9F	0B	64	08	04	00	01	1E	C3	C1	52	E6	F5	1D

三、閘門系統

我們將資訊存在第1 block 前4個Bytes

```
writeBuffer[0] = stationCode[0];
writeBuffer[1] = stationCode[1];
writeBuffer[2] = stationCode.length() > 2 ? stationCode[2] : 0;
writeBuffer[3] = balance;
```

範例：R19 餘額：196

線別(R)、站號(1)、站號(9)、餘額(196)

我們卡片餘額儲存範圍是-59~196

```
int readBalance(byte *data) {
  int balance = data[3];
  if (balance > 196) {
    balance -= 256;  // 將197
  }
  return balance;
}
```

三、閘門系統

1. 卡片的讀取判斷餘額

```
String storedStation = getStationCodeFromData(readBuffer);
int balance = readBalance(readBuffer);

if (balance <= 0) {
  displayMessage("Insufficient balance","Please top up!");
  playInsufficientBalanceSound();
  //儲值196
  //int balance = 196;
  //writeExitData(balance);

  delay(3000);
  return;
}
```

餘額不足時，顯示訊息與發出音效後return

2. 判斷進出站

```
if (storedStation == "") {
  writeEntryData(currentStation, balance);
  displayMessage("Entry @ " + currentStation, "Balance: " + String(balance));
  inCount++;  // 門關著時有人通過視為進入
  peopleCount++;
  trust=true;
  playEntrySound();
  toggleDoor();
  delay(3000);
  toggleDoor();
} else {
  int fare = calculateFare(storedStation, currentStation);
  int newBalance = balance - fare;
  writeExitData(newBalance);
  displayMessage("Exit @ " + currentStation, "Fare: " + String(fare) + " Rema
```

若卡片內站別為空則進站；反之出站並清空站別

三、閘門系統

3. 定時將訊息經MQTT傳到node-red做紀錄

```
void publishSensorData() {
  if (!client.connected()) return;

  // 創建人流感測器JSON資料
  String sensorJson = "{";
  sensorJson += "\"count\":" + String(peopleCount) + ",";
  sensorJson += "\"in_count\":" + String(inCount) + ",";
  sensorJson += "\"out_count\":" + String(outCount) + ",";
  sensorJson += "\"timestamp\":\"" + String(millis()) + "\",";
  sensorJson += "\"door_open\":" + String(doorOpen ? "true" : "false") + ",";
  sensorJson += "\"authorized\":" + String(trust ? "true" : "false");
  sensorJson += "}";

  publishMqttMessage(topic_sensor, sensorJson.c_str());
}
```

欄位	含義
count	當前總人數
in_count	進入人數累積
out_count	離開人數累積
timestamp	系統時間戳
door_open	門是否開啟
authorized	是否授權通行

三、閘門系統

4. 開關門、顯示資訊、音效

```
void toggleDoor() {
  doorOpen = !doorOpen;
  Serial.print("🚪 門控制: ");
  bool wasAlarming = isAlarming;

  if (wasAlarming) {
    noTone(buzzerPin);
    digitalWrite(buzzerPin, LOW);
  }

  doorServo.attach(doorServoPin);
  doorServo.write(doorOpen ? 90 : 0);
  Serial.println(doorOpen ? "開啟 (90°)" : "關閉 (0°)");

  delay(500);

  if (wasAlarming) buzzerPhaseStart = millis();
}
void displayMessage(String line1, String line2) {
  display.clearDisplay();
  display.setCursor(0, 0);
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);
  display.println(line1);
  display.println(line2);
  display.display();
}
```

```
void playEntrySound() {
  // 第一聲
  tone(buzzerPin, 1000, 150);  // 1000
  delay(200);
  // 第二聲
  tone(buzzerPin, 1200, 150);  // 1200
  delay(300);
}
// 出站音效 - 一聲長響
void playExitSound() {
  tone(buzzerPin, 800, 500);   // 800H
  delay(600);
}
void playInsufficientBalanceSound() {
  tone(buzzerPin, 300, 300);   // 長響
  delay(350);
  tone(buzzerPin, 500, 100);   // 短響
  delay(150);
  tone(buzzerPin, 300, 300);   // 長響
  delay(400);
}
```

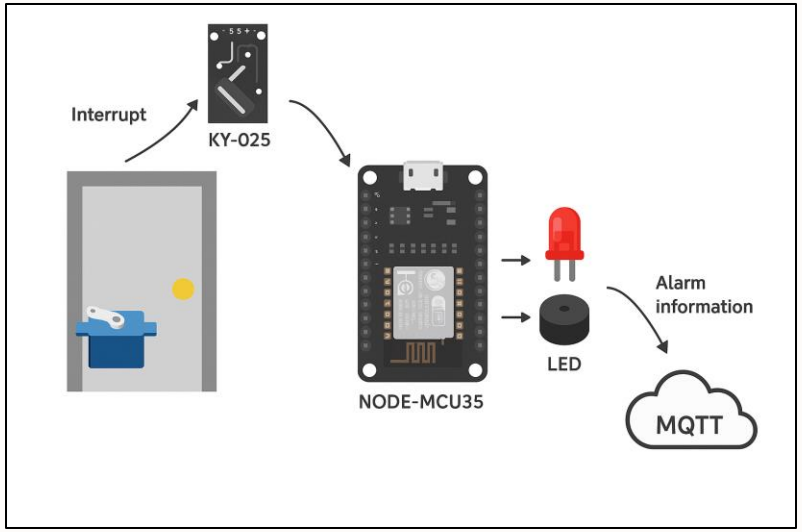
5. 遠端操控開關門(特殊情況使用)

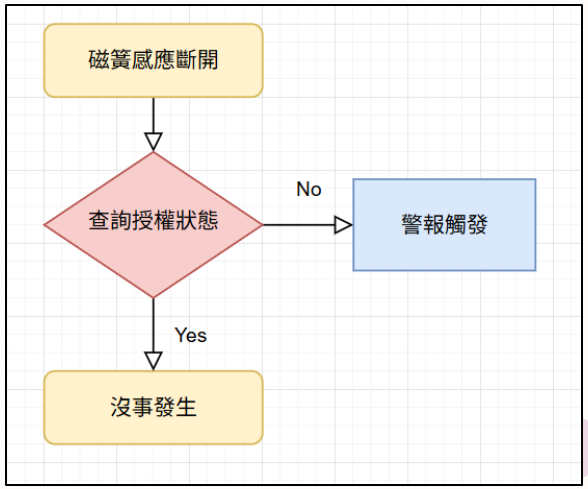
```
void mqttCallback(char* topic, byte* payload, unsigned int length) {
  String message = "";
  for (int i = 0; i < length; i++) {
    message += (char)payload[i];
  }

  Serial.print("收到MQTT訊息 [");
  Serial.print(topic);
  Serial.print("]: ");
  Serial.println(message);

  // 處理門控制命令
  if (String(topic) == topic_door_cmd) {
    if (message == "open") {
      if (!doorOpen) toggleDoor();
    } else if (message == "close") {
      if (doorOpen) toggleDoor();
```

四、強闖偵測 -KY-025 磁簧開關模組

原理：內部兩片金屬簧片靠近磁鐵時會接通，離開磁鐵則斷開，藉此來偵測門的開關

偵測方法：每當有未經授權的斷開時，觸發蜂鳴器與LED運行警報系統，並上傳後端紀錄各項警報資訊





四、強闖偵測

將訊息經MQTTX傳到:

1. Node-RED顯示　　2. HeidiSQL紀錄

```
void publishIntrusionAlert(bool alertStatus) {
  if (!client.connected()) return;

  // 創建強闖偵測JSON資料
  String intrusionJson = "{";
  intrusionJson += "\"alert\":" + String(alertStatus ? "true" : "false") + ",";
  intrusionJson += "\"location\":\""+ currentStation + "\",";
  intrusionJson += "\"timestamp\":\"" + String(millis()) + "\",";
  intrusionJson += "\"door_status\":\"" + String(doorOpen ? "open" : "closed") + "\",";
  intrusionJson += "\"authorized\":" + String(trust ? "true" : "false");
  intrusionJson += "}";

  publishMqttMessage(topic_intrusion, intrusionJson.c_str());
}
```

欄位	含義
alert	警報狀態
location	發生地點
timestamp	系統時間戳
door_status	門是否開啟
authorized	是否授權通行

五、後端介紹

資料庫: HeidiSQL

記錄不同時間段的人流狀況，以及所有的強闖紀錄

通訊端: MQTT

收發所有相關的json訊息與開關門的相關指令

資料流: Node-RED

以流程化的方式獲取資料並轉發給對應的物聯網設備

五、後端介紹

- 資料表架構

```sql
CREATE DATABASE access_control;
USE access_control;

-- 人流資料表
CREATE TABLE people_flow (
    id INT AUTO_INCREMENT PRIMARY KEY,
    count INT,
    in_count INT,
    out_count INT,
    timestamp DATETIME DEFAULT CURRENT_TIMESTAMP
);

-- 強闖記錄表
CREATE TABLE intrusion_log (
    id INT AUTO_INCREMENT PRIMARY KEY,
    alert BOOLEAN,
    location VARCHAR(100),
    timestamp DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

- 人流表

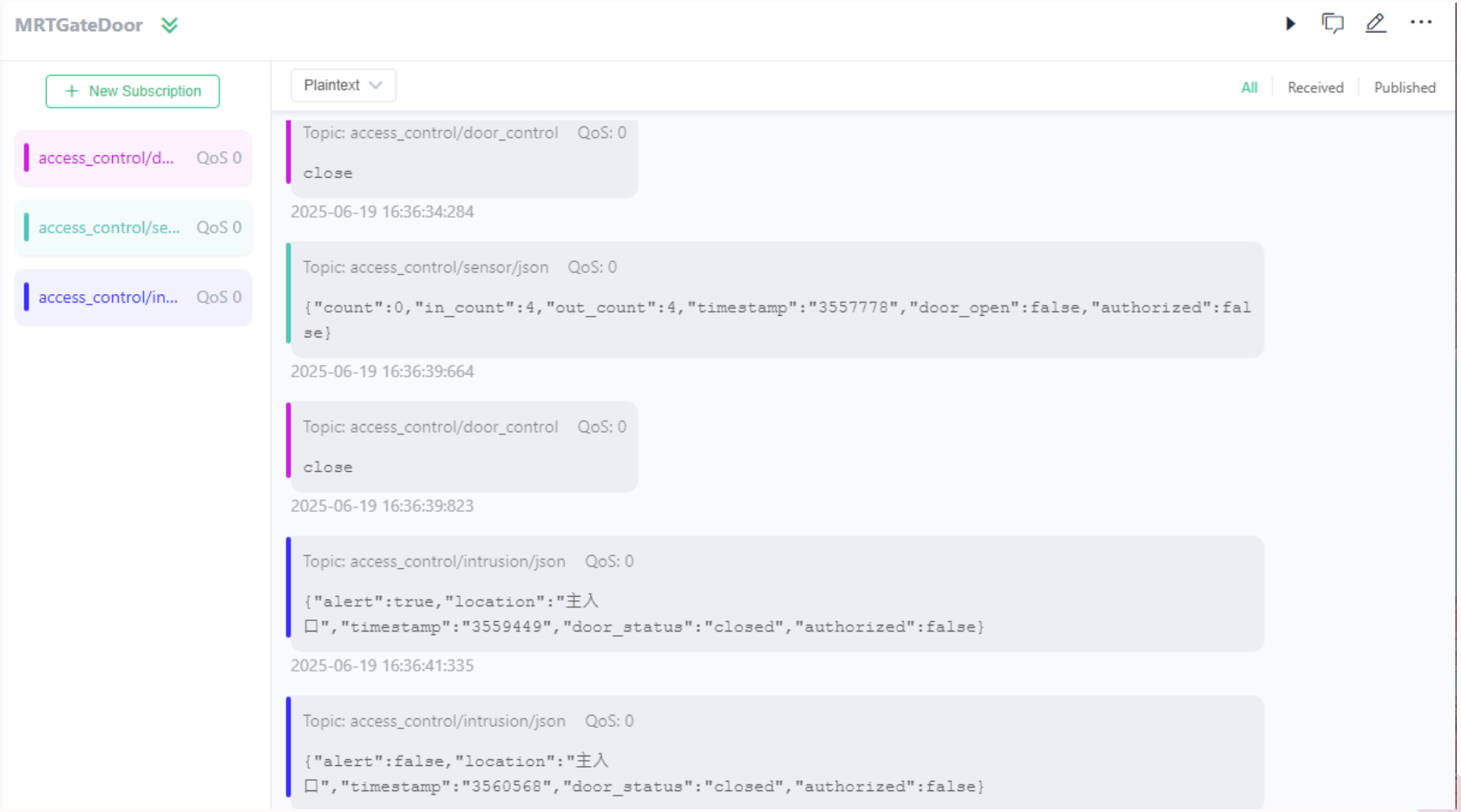
id	count	in_count	out_count	timestamp
683	2	2	0	2025-06-17 18:53:51
684	2	2	0	2025-06-17 18:53:56
685	3	3	0	2025-06-17 18:54:01
686	3	3	0	2025-06-17 18:54:06
687	2	3	1	2025-06-17 18:54:11
688	2	3	1	2025-06-17 18:54:16

- 警報表

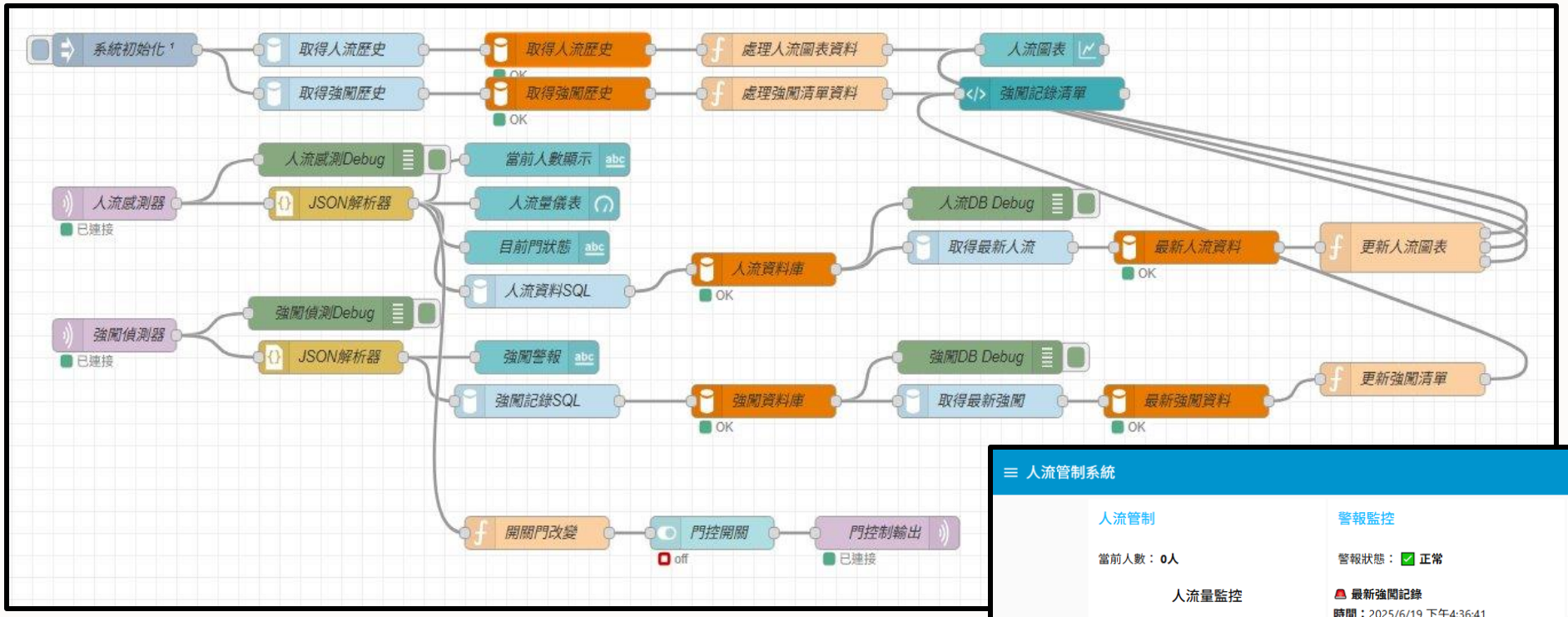
id	alert	location	timestamp
1	1	主入口	2025-06-17 18:05:33
2	0	主入口	2025-06-17 18:05:36
3	1	主入口	2025-06-17 18:05:39
4	0	主入口	2025-06-17 18:05:42
5	1	主入口	2025-06-17 18:06:02

五、後端介紹

- MQTTX訂閱主題與收到資訊



五、後端介紹



- Node-RED流程圖(上)與UI介面(右)

感謝聆聽