

專案主題：

應用樹莓派 IOT-server 解析捷運轉轍器馬達電流偵測數據，並達成行動手機遠端監看轉轍器電流，提升保養效率

緣起：

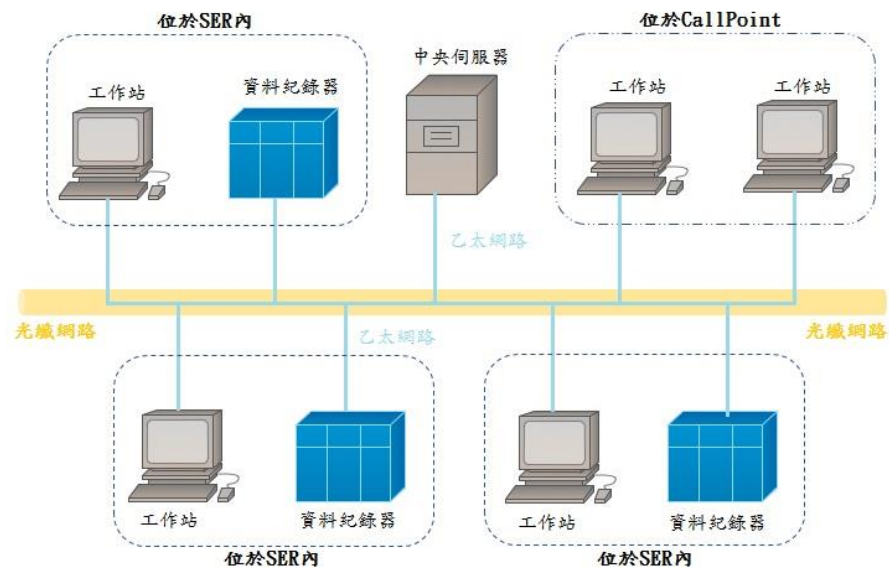
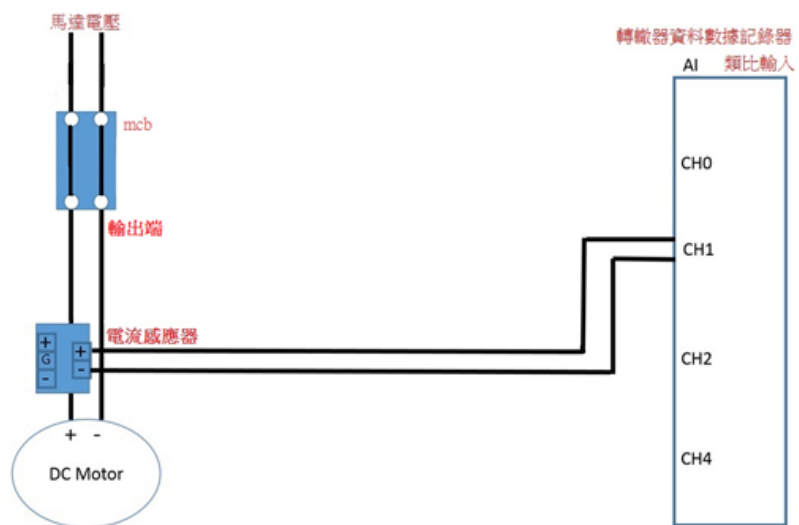
捷運網路為封閉型，在場域存取點才可以進行連網讀取相關資訊，捷運軌道區域重要工作皆於夜間斷電後下軌進行維修，本專題號誌系統轉轍器先前自行發包建置「Turnout 監視系統」，監看其一之功能可以調閱系統中馬達運行電流圖，從中解讀轉轍器正、反位板動時馬達電流變化，當遇阻力時電流變化會呈現較高電流，此時為異常點；當然，必須經過長期經驗累積才可以讓我們對於異常電流運作圖作出正確狀態判讀。

因此，軌道轉轍器保養後能從工務手機取得動態測試時電流運作圖，可以幫助維修人員清楚了解板動作動點位阻力狀況，使得保養品質維持標準以上之水平，並且可以提早防止軌床滾輪與可動軌異常磨擦；不需回設備室開監視系統，軌道保養位置與設備室之間是有很長距離並且有進場程序。

以樹莓派作為 IOT 伺服器，讀取另一端 Turnout 監視系統(Windows7)轉轍器馬達電流偵測數據檔案，檔案傳輸使用 RS232 傳輸協定，在 Windows 主機和樹莓派伺服器間傳輸檔案，能不影響原系統運作及確保網路資訊安全，實現手機遠端連線需求須配合 4G-Router 轉址功能設定來完成，能讓行動手機進行遠端監看轉轍器電流 dashboard 圖。

從感測器取得的資料轉儲存在 MariaDb 資料庫，可以建立來大數據(Big Data)來進行分析，並且從取得的數據分析結果來重新設計預檢保養排程更有效安排人力，數據分析也可有效作為故障排除分析判斷。

## Turnout 監視系統



霍爾電流感應器



PAC 資料記錄器(CPU 模組、數位輸入、數位輸出、類比輸入)

Turnout 監視系統工作站網路架構



Raspberry Pi IOT-Server

Docker-Container :

Grafana

MariaDB(資料庫管理軟體 phpMyAdmin)

Node-Red

注意事項：

Container-系統資料夾放的路徑位置與實際放的位置不同。

以 Node-Red 為例：

Host/volume

/home/thomas/IOTstack/volumes/nodered/data

Path in container

/data

4G-Router  
固定 IP 轉址

捷運軌道旁  
行動基地台

捷運軌道旁保養人  
員工務手機連線

系統架構圖

專題實作解說：

壹、樹莓派 IOT-server(Grafana、MariaDB、NodeRed、Portainer-CE、phpMyAdmin)程式安裝

參考資料：<https://www.youtube.com/watch?v= DO2wHI6JWQ>

<https://learnembeddedsystems.co.uk/easy-raspberry-pi-iot-server>

1.使用 Raspbian OS 64-bit 作業系統，並更新

```
sudo apt update
```

```
sudo apt upgrade
```

2.使用 IOT-stack 安裝程序

```
curl -fsSL https://raw.githubusercontent.com/SensorsIot/IOTstack/master/install.sh | bash
```

3.安裝後，重啟

```
sudo shutdown -r now
```

4.創立 IOTstack 資料夾

```
cd IOTstack/
```

5.利用 menu 安裝需要程式：Grafana、MariaDB、Node-RED、Portainer-CE

使用多人喜好之 MariaDB 資料庫，所以，我們要記得在 menu 選表中將 influxdb 點掉不安裝

6.可在 menu 中 Start Stack 啟動所有容器，並在終端機執行下方指令，來確認容器是否正在運行

```
docker-compose ps
```

7.在 Portainer.io 安裝 MariaDB 管理工具 phpmyadmin，Add container → Image:phpMyAdmin

參考：<https://www.youtube.com/watch?v= 3qPI83Acn8>

The screenshot displays the Portainer.io Community Edition web interface. On the left is a dark blue sidebar with navigation links: Home, local (selected), Dashboard, App Templates, Stacks, Containers, Images, Networks, Volumes, Events, and Host. Below these are Settings (Users, Environments, Registries) and version information (Community Edition 2.16.2 Upgrade). The main area is titled 'Containers' and 'Container list'. It features a search bar, a row of action buttons (Start, Stop, Kill, Restart, Pause, Resume, Remove, Add), and a table of running containers.

<input type="checkbox"/>	Name ↓↑	State ↓↑	Filter	Quick Actions	Stack ↓↑	Image ↓↑	Created ↓↑	IP Address ↓↑	GPUs	Published Ports
<input type="checkbox"/>	grafana	healthy			iotstack	grafana/grafana	2022-12-29 12:12:42	172.19.0.4	none	3000:3000
<input type="checkbox"/>	mariadb	healthy			iotstack	iotstack_mariadb	2022-12-29 12:12:42	172.19.0.3	none	3306:3306
<input type="checkbox"/>	nodered	healthy			iotstack	iotstack_nodered	2022-12-29 12:12:42	172.19.0.5	none	1880:1880
<input type="checkbox"/>	phpMyAdmin	running			-	phpmyadmin:latest	2022-12-30 08:44:02	172.19.0.6	none	8080:80
<input type="checkbox"/>	portainer-ce	running			iotstack	portainer/portainer-ce	2022-12-29 12:12:42	172.19.0.2	none	8000:8000  9000:9000

Items per page

貳、具資安機制之資料傳送

一、Windows 工作站-傳送方

1.設定 Com port，本機為例 com6

2.Windows 命令提示字元執行 main

main 檔案是 python 文件打包成 exe 文件






<https://www.youtube.com/watch?v=38cqMSqkrG0>

3.設定 windows 工作排程器

<https://www.youtube.com/watch?v=oKDGLr7d5-Q>

4. Turnout 監視工作站，產生之 rawdata 以 Pointswitchname\_direction\_YMDHms，存成逗點隔開檔案.csv

› 新增磁碟區 (D:) › Pointswitch\_rawdata

名稱	修改日期	類型	大小
 config	2023/1/10 下午 08:21	檔案	1 KB
 history	2023/1/10 下午 10:12	檔案	1 KB
 main.exe	2023/1/10 下午 09:57	應用程式	7,139 KB
 main.py	2023/1/10 下午 09:56	Python 來源檔案	9 KB
 P2103N_20211001051336.csv	2021/10/1 上午 05:15	Microsoft Excel 逗點...	14 KB
 P2103N_20211001072759.csv	2021/10/1 上午 07:29	Microsoft Excel 逗點...	14 KB
 P2103N_20211001081259.csv	2021/10/1 上午 08:14	Microsoft Excel 逗點...	14 KB
 P2103N_20211001091259.csv	2021/10/1 上午 09:14	Microsoft Excel 逗點...	14 KB
 P2103N_20211001110337.csv	2021/10/1 上午 11:05	Microsoft Excel 逗點...	14 KB
 P2103N_20211001171258.csv	2021/10/1 下午 05:14	Microsoft Excel 逗點...	14 KB
 P2103N_20211001181258.csv	2021/10/1 下午 06:14	Microsoft Excel 逗點...	14 KB
 P2103N_20211005033951.csv	2021/10/5 上午 03:55	Microsoft Excel 逗點...	15 KB
 P2103N_20211005044907.csv	2021/10/5 上午 04:50	Microsoft Excel 逗點...	14 KB
 P2103N_20211005072757.csv	2021/10/5 上午 07:29	Microsoft Excel 逗點...	14 KB
 P2103N_20211005081033.csv	2021/10/5 上午 08:12	Microsoft Excel 逗點...	14 KB
 P2103N_20211005091256.csv	2021/10/5 上午 09:14	Microsoft Excel 逗點...	14 KB
 requirements.txt	2022/12/23 下午 03:43	文字文件	1 KB

讓其設定「工作排程器」固定時間執行 main.exe(R232 資料傳送)，將現場轉轍器動作資料利用傳送程式傳送到樹莓派 IOTserver 資料夾/home/Thomas/IOTstack/volumes/nodered/data/Pointswitch

## 二、Raspberry Pi-接收方

- 1.設定/dev/ttyUSB0(通常為 0)
2. pip3 install -r requirements.txt，匯入相依函式庫(第 1 次執行)
- 3.終端機，執行 sudo python ./main.py

Python 程式碼：

```
import time
import os
import serial
import json
import yaml
import pprint
import hashlib
import threading as th

# {cmdline:, payload:, num:, hash:}
'''
# 建立 MD5 物件
m = hashlib.sha256()

# 要計算 MD5 雜湊值的資料
data = "G. T. Wang"
```



```
# 更新 MD5 雜湊值
m.update(data)

# 取得 MD5 雜湊值
h = m.hexdigest()
print(h)

cmdline
0 = resend command payload = miss num (option)
'''
```

```
def hash_file(filepath):
    m = hashlib.sha256()
    with open(filepath, 'rb') as f:
        m.update(f.read())
    h = m.hexdigest()
    return h
```

```
def hash_cmd(command):
    m = hashlib.sha256()
    m.update(command.encode())
    h = m.hexdigest()
```

```
return h
```

```
class transerver:
```

```
    def __init__(self, port, baudrate, retry_delay=20, timeout=None):  
        self.port = port  
        self.baud_rate = baudrate  
        self.timeout = timeout  
        self.retry_delay = retry_delay  
        self.ser = serial.Serial(self.port, self.baud_rate, timeout=self.timeout)  
        self.ser.flush()  
        self.p = th.Thread(target=self.get_command, daemon=True)  
        self.file_buff = b''  
        self.num = 0  
        self.rece_cmd = []  
        self.trans_cmd = []  
        self.read_flag = False  
        self.read_start()  
        self.write_flag = False
```

```
    def trans_file(self, file_path):  
        del self.file_buff  
        with open(file_path, 'rb') as f:  
            self.file_buff = f.read()
```

```
cmd = {'cmdline': 'write_file', 'payload': self.file_buff.hex(), 'filename': file_path,
      'sha256': hash_file(file_path)}
self.trans_handle(cmd)
print(f'傳送成功: {file_path}')
return True
```

```
def get_command(self):
    try:
        while self.read_flag:
            while self.ser.in_waiting > 0:
                st = time.time()
                cmd = self.ser.readline().decode().replace('\n', '')
                try:
                    if len(cmd) > 0:
                        print('reading cmd')
                        cmd = json.loads(cmd)
                    else:
                        continue
                except:
                    print(f'Error packet {cmd}')
                    continue
            if 'cmdline' in cmd.keys() and 'num' in cmd.keys():
                print(f'Get Command : {cmd["cmdline"]}')
    except:
```

```
self.rece_cmd.append(cmd)
if cmd['cmd_hash'] == hash_cmd(str(cmd['cmdline']) + str(cmd['payload']) + str(cmd['num'])):
    print('Command hash Conform')
    if not cmd['cmdline'] == 'Success Receive':
        re_cmd = {}
        re_cmd['cmdline'] = 'Success Receive'
        re_cmd['payload'] = hash_cmd(str(cmd['cmdline']) +
                                     str(cmd['payload']) + str(cmd['num']))

        re_cmd['num'] = self.num
        re_cmd['cmd_hash'] = hash_cmd(str(re_cmd['cmdline']) +
                                       str(re_cmd['payload']) + str(re_cmd['num']))

        self.ser.write('\n'.encode())
        self.ser.write(json.dumps(re_cmd).encode())
        self.ser.write('\n'.encode())
        self.num = self.num + 1

    self.run_cmd(cmd)
    #self.run_cmd(cmd)
else:
    print(cmd)
else:
    print(cmd)
print(f'用時{time.time() - st}')

except:
```

```
        print(f'Error with Serial Status is {self.ser.is_open}')
    return f'read stop'

def read_start(self):
    self.read_flag = True
    self.p = th.Thread(target=self.get_command, daemon=True)
    self.p.start()
    return True

def read_stop(self):
    self.read_flag = False
    self.p.join()
    return True

def trans_command(self, cmd):
    self.read_stop()
    if type(cmd) is dict:
        if 'cmdline' in cmd.keys() and 'payload' in cmd.keys():
            cmd['num'] = self.num
            cmd['cmd_hash'] = hash_cmd(str(cmd['cmdline']) + str(cmd['payload']) + str(cmd['num']))
            self.num = self.num + 1
            self.ser.write('\n'.encode())
            self.ser.write(json.dumps(cmd).encode())
```

```
        self.ser.write('\n'.encode())
        self.trans_cmd.append(cmd)
    self.read_start()
    return True, cmd
else:
    self.read_start()
    return False, cmd

def trans_handle(self, cmd):
    while not self.read_flag:
        time.sleep(0.1)
    count = 1
    handle = self.trans_command(cmd)
    while True:
        if count % self.retry_delay == 0:
            handle = self.trans_command(cmd)
        count = count + 1
        if handle[0]:
            if handle[1]['cmd_hash'] in [x["payload"] for x in self.rece_cmd if x["cmdline"] == 'Success Receive']:
                print(f'trans success: {cmd["cmd_hash"]}')
                print('Conform')
                break
        else:
```

```
        print(f'trans fail: {cmd["cmd_hash"]}')
        print(f'Retry in 5s')
        time.sleep(5)
        count = 0
        handle = self.trans_command(cmd)
        time.sleep(1)
    return True

def run_cmd(self, command):
    if command['cmdline'] == 'write_file':
        if 'filename' in command.keys():
            print(f'Start Write File : {command["filename"]}')
            with open(command['filename'], 'wb') as f:
                f.write(bytes.fromhex(command['payload']))
            if command['sha256'] == hash_file(command['filename']):
                return True, f'接收並驗證正確'
            else:
                return False, f'sha256 較驗錯誤'
        else:
            print(f'Start Write File : {command["sha256"]}')
            with open(str(command['sha256']), 'wb') as f:
                f.write(bytes.fromhex(command['payload']))
    elif command['cmdline'] == 'Success Receive':
```

```

        print(f'解析接收事件: {command["payload"] in [x["cmd_hash"] for x in self.trans_cmd]}')
    else:
        print(f'*****{command}*****')
    return True

if __name__ == '__main__':
    with open('config', 'r', encoding="utf-8") as f:
        config = yaml.full_load(f)
    finish_log = []
    if os.path.exists('history'):
        with open('history', 'r', encoding="utf-8") as f:
            finish_log = f.readlines()
            finish_log = [x.replace('\n', '') for x in finish_log]
    pprint.pprint(config)
    tr = transerver(config['Serial']['COM'], config['Serial']['baudrate'],
                    retry_delay=config['Serial']['Conform_timeout'])
    if config['Serial']['Mode'] == 1:
        print(f'Write Mode ON')
        trans_file_list = []
        for root, dirs, files in os.walk(".", topdown=False):
            for name in files:
                if root == '.':

```



```

        for i in config['File']['file_type']:
            if name.endswith(i):
                print(f'搜尋到新檔案: {name}')
                trans_file_list.append(name)

count_new = 0
count_old = 0
print(finish_log)
print(trans_file_list)
with open('history', 'w', encoding="utf-8") as f:
    for i in trans_file_list:
        f.write(i)
        f.write('\n')
        if i not in finish_log:
            tr.trans_file(i)
            count_new = count_new + 1
        else:
            count_old = count_old + 1
    print(f'任務完成，共掃描\n{count_new}個新檔案\n{count_old}個舊檔案')
else:
    print(f'couch potato')
    while True:
        time.sleep(0.5)

```

檢附：main.py

參考資料：

<https://pythonhosted.org/pyserial/>

Welcome to pySerial's documentation

<https://www.runoob.com/python/os-walk.html>

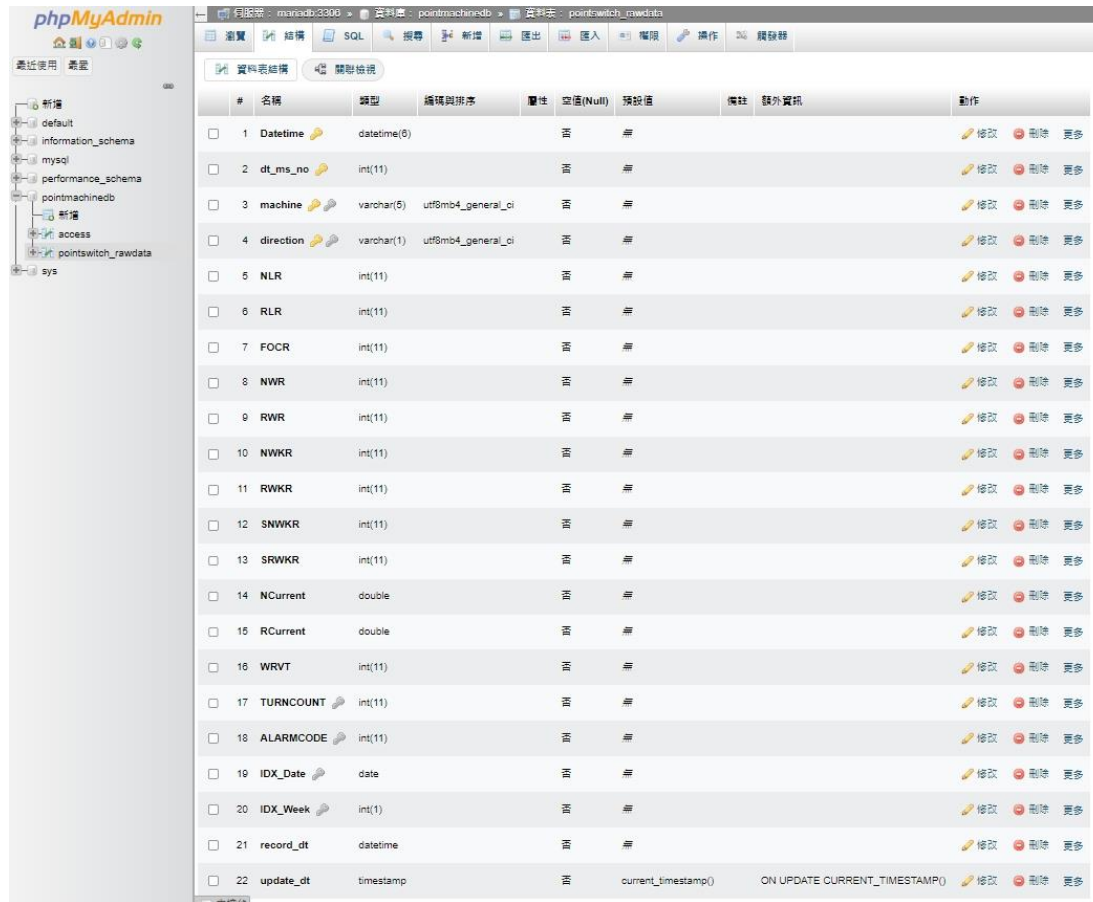
Python os.walk() 方法

<https://blog.gtwang.org/programming/python-md5-sha-hash-functions-tutorial-examples/>

Python 計算 MD5 與 SHA 雜湊教學與範例

叁、資料庫(使用 mariadb 名稱登入)

1.使用 phpMyAdmin 管理 MariaDb 資料庫，新增資料結構



The screenshot shows the phpMyAdmin web interface. On the left is a sidebar with a database tree. The main area displays the 'Structure' tab for the 'pointswitch\_rawdata' table. The table has 22 columns with various data types and attributes.

#	名稱	類型	編碼與排序	屬性	空值(Null)	預設值	備註	額外資訊	動作
<input type="checkbox"/>	1 Datetime	datetime(6)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	2 dt_ms_no	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	3 machine	varchar(5)	utf8mb4_general_ci		否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	4 direction	varchar(1)	utf8mb4_general_ci		否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	5 NLR	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	6 RLR	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	7 FOGR	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	8 NWR	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	9 RWR	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	10 NWKR	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	11 RWKR	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	12 SNWKR	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	13 SRWKR	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	14 NCurrent	double			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	15 RCurrent	double			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	16 WRVT	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	17 TURNCOUNT	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	18 ALARMCODE	int(11)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	19 IDX_Date	date			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	20 IDX_Week	int(1)			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	21 record_dt	datetime			否	無			<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>
<input type="checkbox"/>	22 update_dt	timestamp			否	current_timestamp()	ON UPDATE CURRENT_TIMESTAMP()		<a href="#">修改</a> <a href="#">刪除</a> <a href="#">更多</a>

檢附：pointswitch\_rawdata.sql

## 2.Node-RED

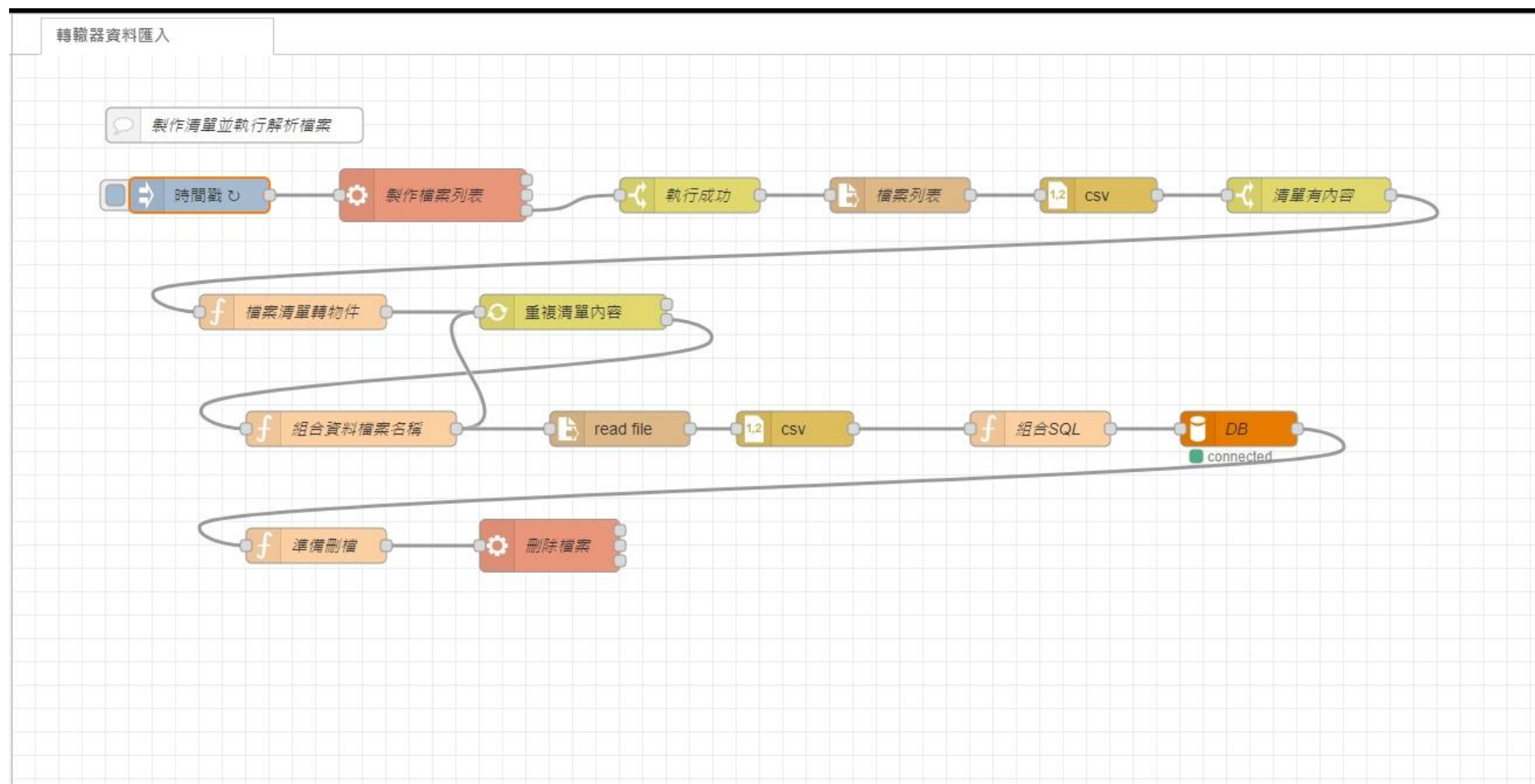
Node-RED 是 IBM Emerging Technology 開發，一套開放原始碼使用瀏覽器 Web 介面的視覺化物聯網開發工具，可以拖拉節點和連接節點來建立流程(Flows)，使用流程來建立物聯網應用程式。

參考資訊：Node-RED : Mariadb on Node-RED 使用

<https://www.youtube.com/watch?v=KPhUmvSShSo>


<https://www.youtube.com/watch?v=mvlub1N0U7I>

3.應用 Node-RED 清洗 RawData.csv 檔案(制作清單並執行解析檔案)及將轉檔後的資料匯入資料庫



檢附：轉轍器資料匯入(Node-RED).json

肆、Grafana，Data Source 設定，指定 Host、Database、User、Password，執行 Save&test，必須為 Database Connection OK

 **Data Sources / MySQL**  
Type: MySQL

[Settings](#)

Alerting supported

Name

MySQL

Default

☒

### MySQL Connection

Host	172.19.0.3:3306		
Database	pointmachinedb		
User	root	Password	<div>configuredReset</div>
Session timezone	(default)		
TLS Client Auth	<div><input checked="" type="checkbox"/></div>	With CA Cert	<div><input checked="" type="checkbox"/></div>
Skip TLS Verify	<div><input checked="" type="checkbox"/></div>		

### Connection limits

Max open	<div>unlimited</div>
Max idle	<div>2</div>
Max lifetime	<div>14400</div>

### MySQL details

Min time interval	<div>1m</div>
-------------------	---------------

User Permission

The database user should only be granted SELECT permissions on the specified database & tables you want to query. Grafana does not validate that queries are safe so queries can contain any SQL statement. For example, statements like `USE etherdb;` and `DROP TABLE user;` would be executed. To protect against this we highly recommend you create a specific MySQL user with restricted permissions. Checkout the [MySQL Data Source Docs](#) for more information.

Database Connection OK.

Back

Explore

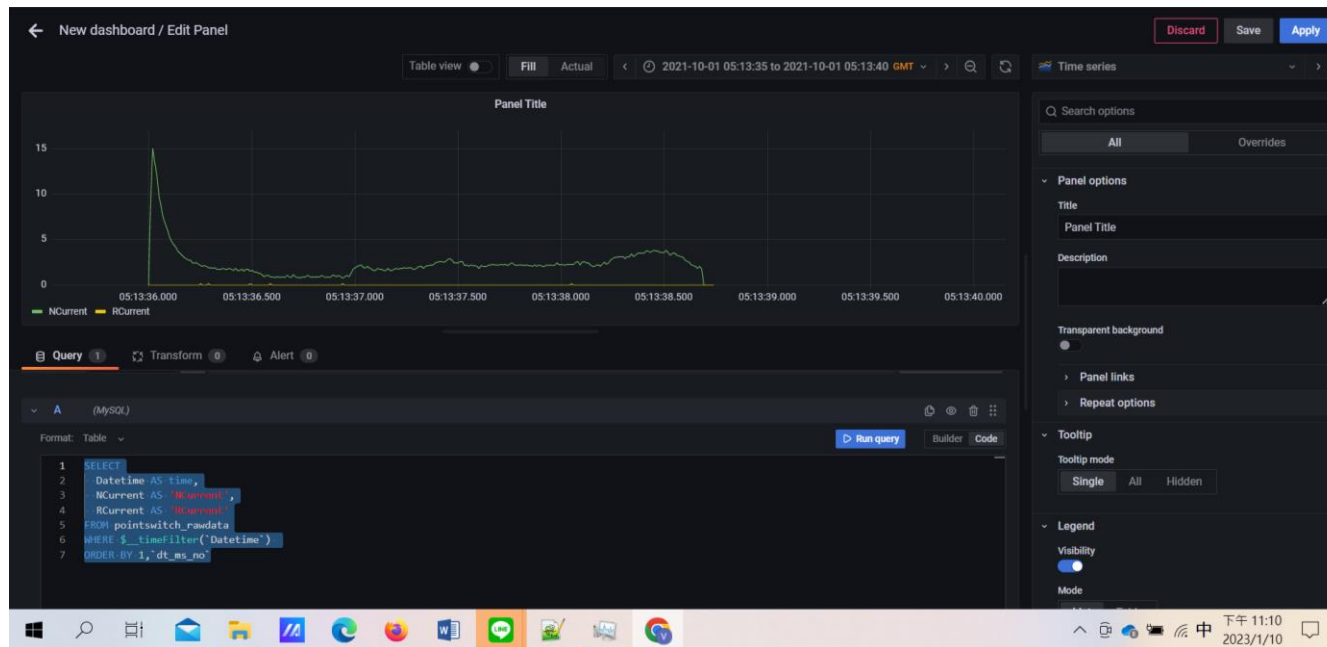
Delete

Save & test

1.New dashboard, 選擇輸入 Code 方式，輸入以下

```
SELECT
  Datetime AS time,
  NCurrent AS 'NCurrent',
  RCurrent AS 'RCurrent'
FROM pointswitch_rawdata
WHERE $__timeFilter(`Datetime`)
ORDER BY 1,`dt_ms_no`
```

注意「Time series」time 的設定，取用 NCurrent 及 RCurrent 值，按 Run query，Apply，並 Save dashboard



## 伍、4G-Router 轉址設定

採用手邊華為 4G-Router，將固定 IP 轉址內部連接樹莓派 IP Address:port，如，192.X.X.105:3000(Grafana)，並作測通測試。

 **HUAWEI**

繁體中文 (台灣) 說明 admin 登出

5G

主螢幕 | 統計資訊 | 簡訊 | 更新 | **設定** | 更多

快速設定

撥號

VoLTE

乙太網路

VPN

WLAN

DHCP

VoIP

安全性

PIN 碼管理

裝置管理

防火牆開關

MAC 篩選

LAN IP 篩選器

→ 虛擬伺服器

特殊應用程式

DMZ 設定

SIP ALG設定

UPnP 設定

NAT 設定

網域名稱篩選

DDNS

家長控制

TR-069 管理

系統

### 虛擬伺服器

設定虛擬伺服器，使外部的電腦能夠使用由LAN所提供的WWW、FTP或其他服務。

- **IP 位址：**指定位於LAN中的一部電腦來提供服務。
- **LAN/WAN 連接埠：**虛擬伺服器的電腦連接埠範圍是 1-65535。
- **協定：**服務所使用的協定。
- **注意：**在按下“應用”按鈕前，設定將不會生效。

#### 虛擬伺服器清單

名稱	WAN 連接埠	LAN IP 位址	LAN 連接埠	協定	狀態	選項
Grafana	3000-3000		3000-3000	TCP/UDP	開	<a href="#">編輯</a> <a href="#">刪除</a>

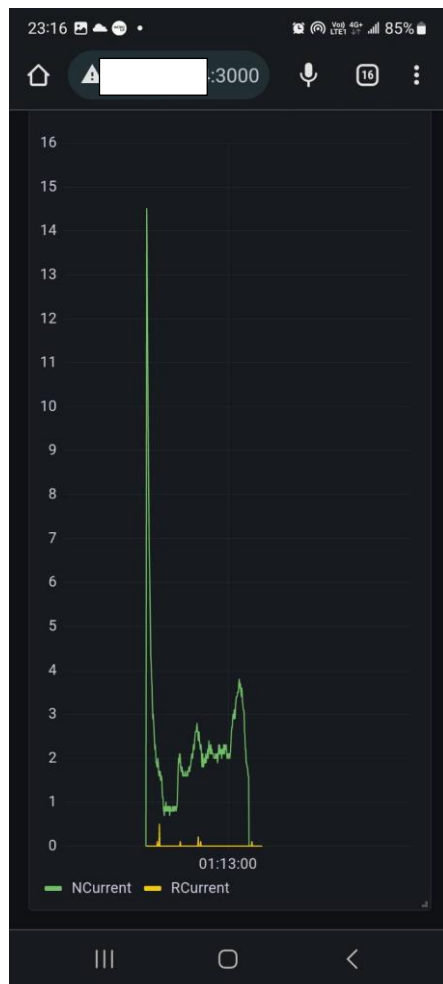
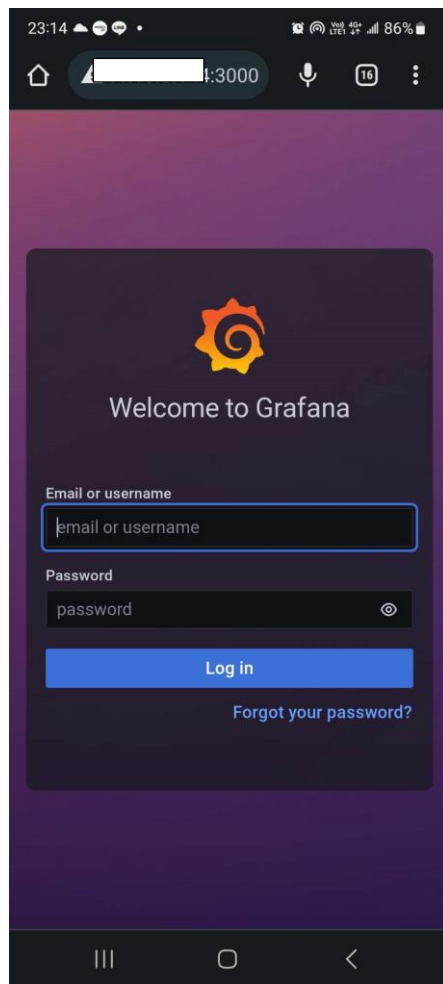
新增

應用

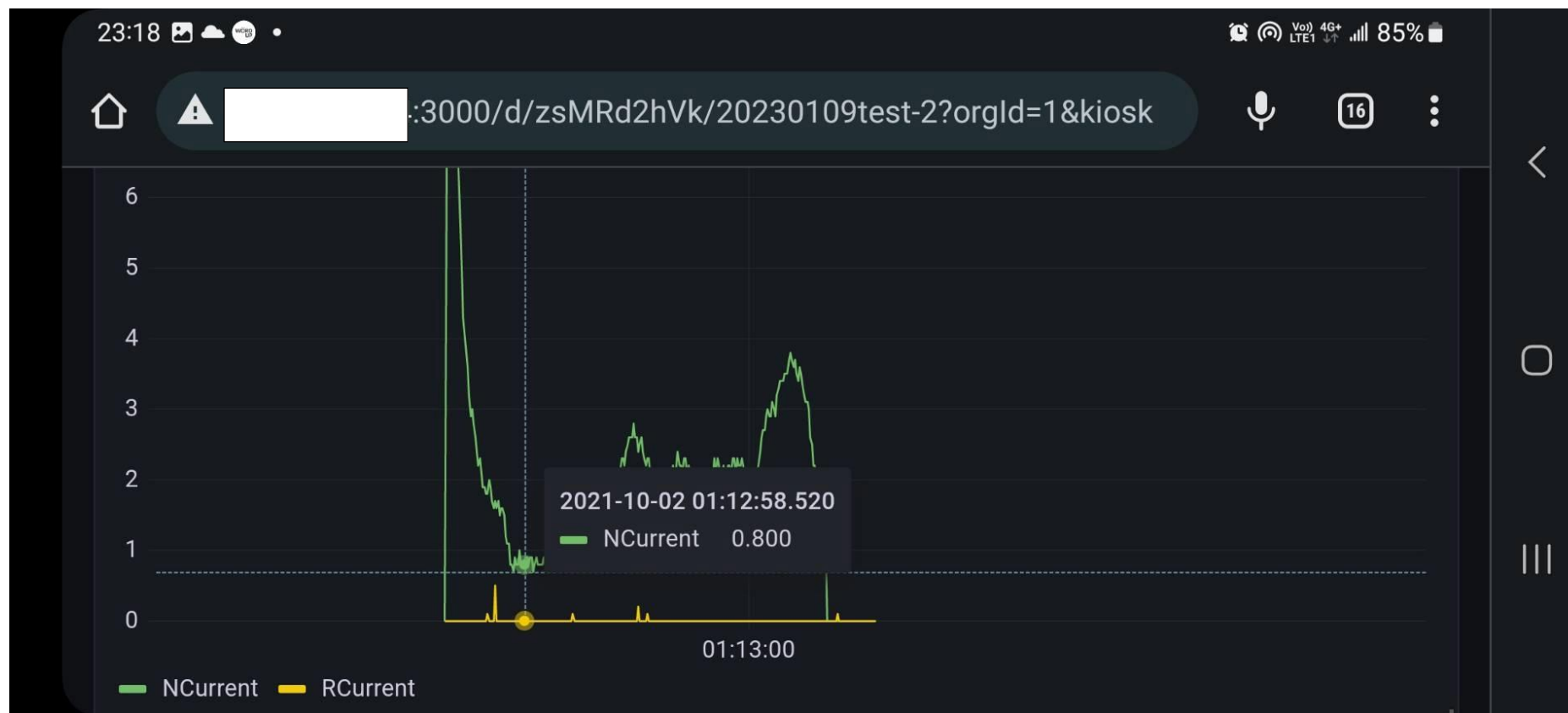


陸、手機登入

1.手機輸入固定 IP：X.X.XX:3000(Grafana port)，輸入帳號、密碼



直式檢視



横式檢視



①4G\_Router

②RaspbeeryPi\_IOT

③Turnout 監視工作站