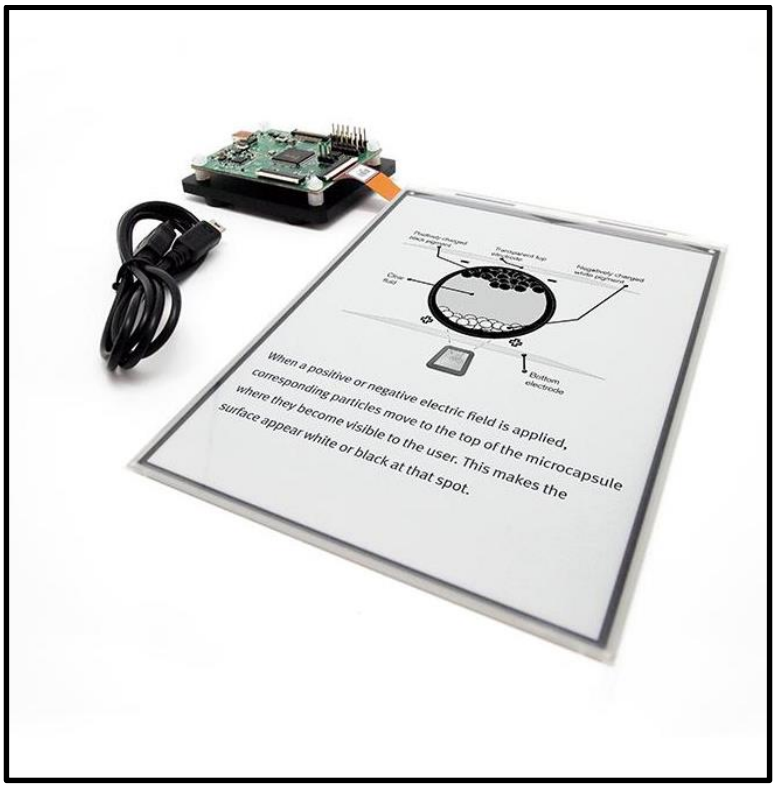
國立台灣海洋大學資訊工程學系專題報告

題目：使用 Line-bot 控制 E-Paper 電子門牌顯示



作者

學號:00757046　姓名:黃子軒　e-mail:00757046@mail.ntou.edu.tw

學號:00657051　姓名:林梓峰　e-mail:timfung121@gmail.com

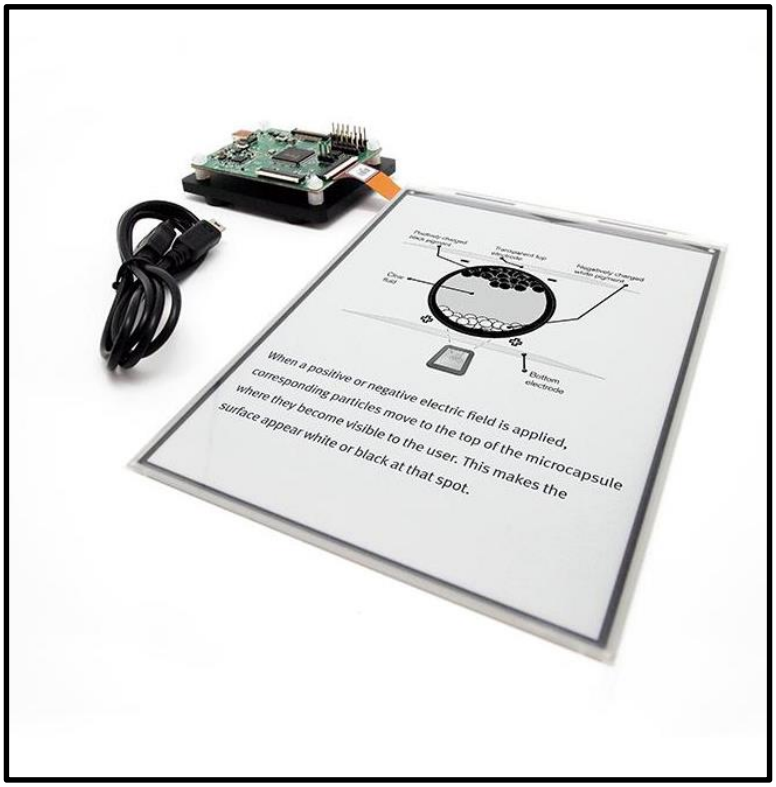
專題報告編號：競賽組第 25 組

指導教授：辛華昀 博士

中華民國 110 年 12 月 02 日

Project Report, NTOU CSE

Combining embedded system and Line-bot to control the display of E-Paper doorplate



Author:

Student Number:00757046,
Name:Huang, Tzu-Hsuan,
e-mail:00757046@mail.ntou.edu.tw

Student Number:00657051,
Name:Lam, Tsz Fung,
e-mail:timfung121@gmail.com

Project Report Number: Competition Group 25

Advisor：Dr. Haw Yun Shin

2021 / 12 / 02

目錄

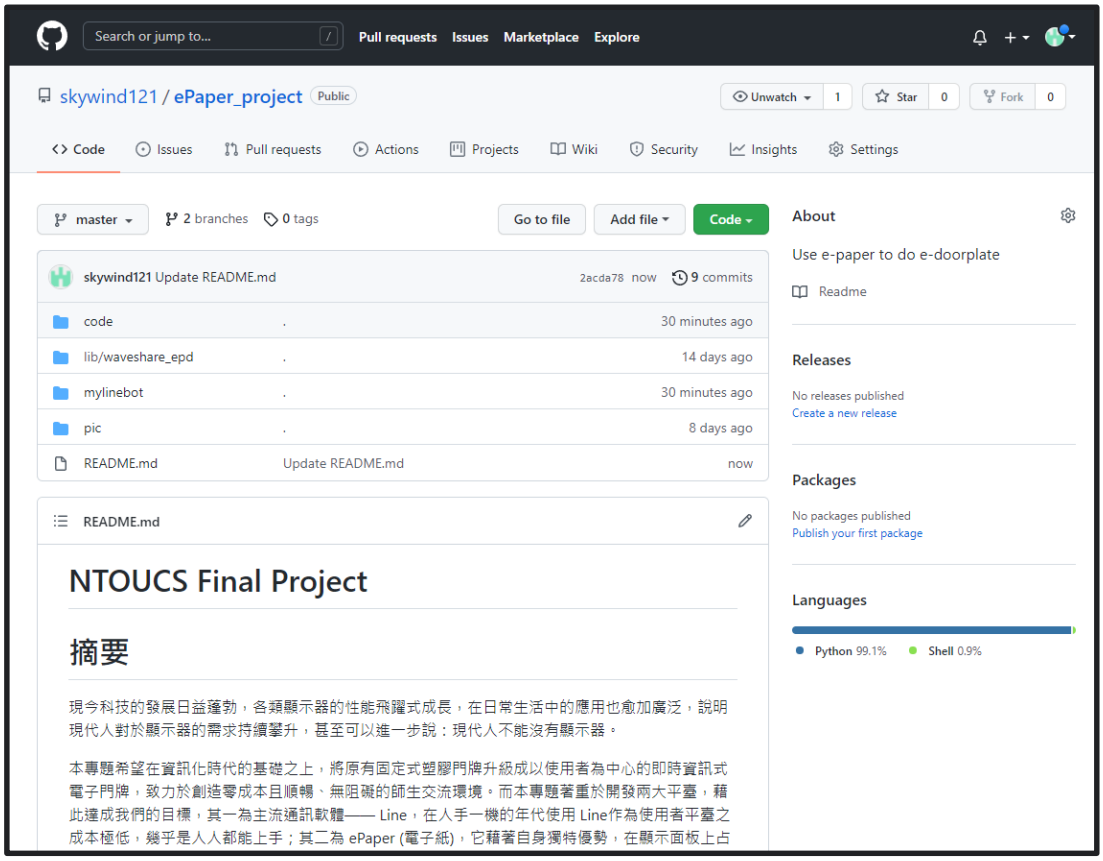
圖目錄

表目錄

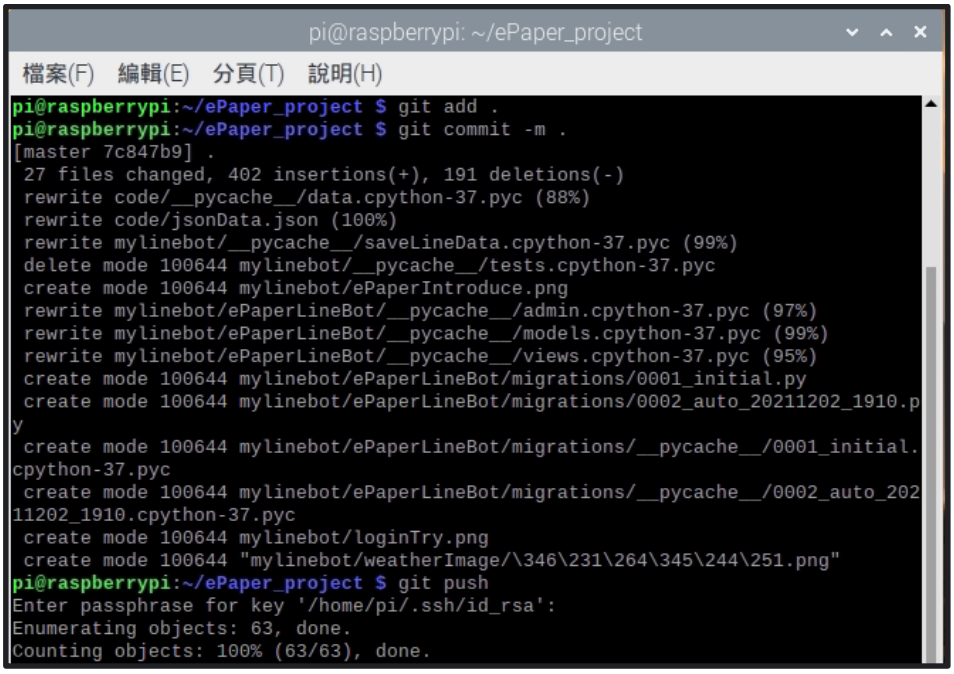
專題分工及貢獻度說明

編號	姓名	主要工作內容	專題貢獻度 (100%)
1	林梓峰	a、Line-bot UI 美化 b、E-Paper 電子門牌開發環境建置 c、E-Paper 電子門牌版面配置 d、E-Paper 電子門牌天氣預報功能 e、Demo 影片拍攝與剪輯 f、海報與書面報告撰寫	50%
2	黃子軒	a、Line-bot 基本環境建置 b、Line-bot 服務選單智慧模板化 c、PostgreSQL 資料庫建置 d、RPi.GPIO Sensor 功能發想與建置 e、Demo 影片拍攝 f、海報與書面報告撰寫	50%

專題管理系統截圖

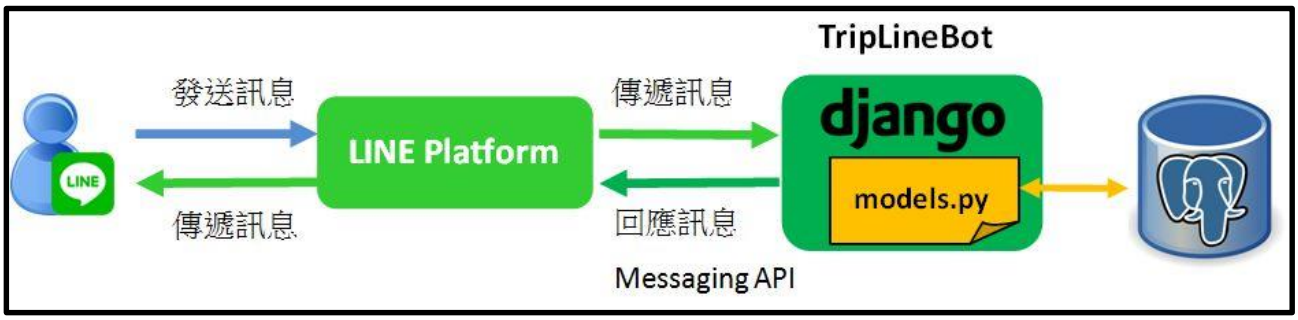


圖一、GitHub 界面



圖二、將每次修改後的新程式碼從 Raspberry Pi 更新至 GitHub

摘要（Abstract）



圖三、使用者使用 Line-bot API 從 PostgreSQL 資料庫讀取資料之執行架構

現今科技的發展日益蓬勃，各類顯示器的性能飛躍式成長，在日常生活中的應用也愈加廣泛，說明現代人對於顯示器的需求持續攀升，甚至可以進一步說：現代人不能沒有顯示器。

本專題希望在資訊化時代的基礎之上，將原有固定式塑膠門牌升級成以使用者為中心的即時資訊式電子門牌，致力於創造零成本且順暢、無阻礙的師生交流環境。而本專題著重於開發兩大平臺，藉此達成我們的目標，其一為主流通訊軟體—— Line，在人手一機的年代使用 Line 作為使用者平臺之成本極低，幾乎是人人都能上手；其二為 E-Paper（電子紙），它藉著自身獨特優勢，在顯示面板上占有一席之地，其超低功耗與雙穩態(Bi-Stable)的特性恰好符合我們的需求，結合 Raspberry Pi 進行有效地資料傳輸與控制 E-Paper 的顯示，不僅對於本專題提供優良的開發過程，甚或能看見不凡的互動成果。

一、 專題作品簡介（Introduction）

1. 研究動機

觀察系所教授的門牌上之狀態欄「請進、上課中、開會中」其實很少會被更改，而身處科技世代的我們，人人的手機裡都會安裝 Line ，因此本專題團隊決定透過 Line-bot 串接樹莓派並且讓 E-Paper 擔任電子門牌的角色，讓教授(電子門牌主人)可以隨時隨地使用 Line-bot 去更新電子門牌的所有資訊，其中包括姓名、職稱、狀態與電子郵件。

2. 研究目的

建立在研究動機所述之基礎，逐步擴大規模後，在教授端(電子門牌主人)雙方的電子門牌之間開始允許透過 Line 實現留言互動與狀態變更，而在教授端(電子門牌主人)與學生端(電子門牌訪客)亦能問答互動與狀態確認，讓電子門牌發揮最大功用與效益。

3. 研究器材

- 電子紙 (E-Paper)
- 電子紙驅動程序帽 (E-Paper Driver HAT)
- 樹莓派 (Raspberry Pi)
- 傳輸線 (HDMI)
- 傳輸線 (Micro USB)
- 記憶卡 (SD)
- 通用型之輸入輸出 (GPIO)
- Asus ZenFone 6 (Android OS)
- IPhone 8 Plus (IOS)

4. 背景知識簡介

- E-Paper

以電泳式電子墨水(Electrophoretic Ink)技術作為基礎，將其塗佈於一層塑膠薄膜上，再貼覆上薄膜電晶體(TFT)電路，並經由驅動 IC 控制帶靜電的有色微小顆粒，顆粒有兩色，各自帶正負不同的靜電，以同極排斥、異極相吸之原理，轉換微小顆粒的位置，形成可視的像素圖案。

圖四、樹莓派電子紙 E-Paper/eInk (7.5 吋,3 色,紅黑白)

- GPIO

GPIO 為通用型之輸入輸出的簡稱，功能類似 8051 的 P0—P3，其接腳可以提供程式設計師編寫程式去自由控制 I/O ，PIN 腳依現實考量可作為通用輸入（GPI）、通用輸出（GPO）或通用輸入與輸出（GPIO）。

- Raspberry Pi 3 Model B+

樹莓派是由英國樹莓派基金會開發的 Linux 單晶片電腦，有著低價成本、功能多元等優勢，目的是以自由軟體促進基礎的電腦科學教育。本專題使用 2018 年 Pi Day（三月十四日)發佈的第三代 B+ 改進版本，電源使用 Micro USB 介面，額外功率為 10.5 瓦，內建無線網路 802.11n 及藍芽 4.1。

圖五、Raspberry Pi 3 Model B+

- Django

Django 是一高等級的 Python 網路框架，使程式設計師能迅速開發安全與可維護的網站。它由經驗豐富的開發者們構建，同時是免費與開源的，有活躍繁榮的社群、豐富文檔以及許多解決方案。Django 負責處理網站開發中最麻煩的部分，因此我們就能專注於編寫應用程式，而無需從頭開發，它可以使應用程式具有完備性(開箱即用)、資訊安全(加密 Hash 函數)、可擴展性(無共享結構)、可維護性(DRY 原則與 MVC 模式)與可移植性(跨平臺運行)等優點。

- PostgreSQL

  專案 PostgreSQL 最初始於加州大學伯克利分校計算機系開發的 POSTGRES 版本 4.2，以其為基礎的自由對象關係型數據庫管理系統(ORDBMS)。PostgreSQL 是一套功能強大、開放原始碼物件關聯的資料庫系統，在系統可靠性、資料完整性和正確性皆獲得極佳評價，亦具備先進資料庫系統的功能，此外它也支持大部分 SQL 標準且額外提供了許多現代特性，因此有越來越多的企業用 PostgreSQL 來取代 MySQL，已成為開發首選的開源關係型數據庫。

- Ngrok

  Ngrok 屬於一種反向代理(Reverse Proxy：在電腦網路中是代理伺服器的一種)，其原理是可以把外界的請求轉發至本機端所指定的 Port，意即由 Ngrok 產生一串公開的網址，讓外部網路存取本機端上的 Port，通過公共端點和本地端運行的網路伺服器之間建立一安全通道，執行安全的資料傳輸。它的優點是測試前後端架構或 Debug 非常快速，而且還提供了 https 的服務讓使用上更加安全，甚至還能設置密碼保護傳輸過程不被攻擊，除此之外還可以捕獲與分析通道上的所有流量，便於後期分析和重放。

二、 軟硬體架構與整合 (Software and Hardware Architecture and Integration)

1. 軟體架構

- 使用語言：Python
- 使用架構：Django
- 使用 Python 套件：
    1. RPi.GPIO
    2. gspread
    3. json
    4. time
    5. logging
    6. traceback
    7. os
    8. sys
- 使用 Line-bot 套件：
    1. linebot
        a. LineBotApi
        b. WebhookParser
    2. linebot.exceptions
        a. InvalidSignatureError
        b. LineBotApiError
    3. linebot.models
        a. MessageEvent
        b. PostbackEvent

c. TextMessage

d. TextSendMessage,

e. TemplateSendMessage

f. ButtonsTemplate

g. ConfirmTemplate

h. MessageTemplateAction,

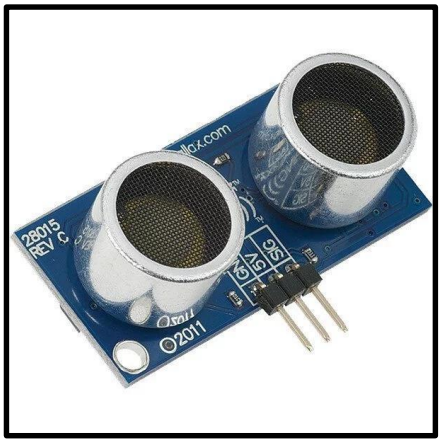
i. PostbackTemplateAction

j. URITemplateAction

- 使用應用程式
  1. LINE 即時通訊平臺
  2. Google Sheet 電子試算表

2. 硬體架構

- E-Paper
- Raspberry Pi 3 Model B+
- Distance Sensor



圖六、GPIO：距離感測器

- Buzzer

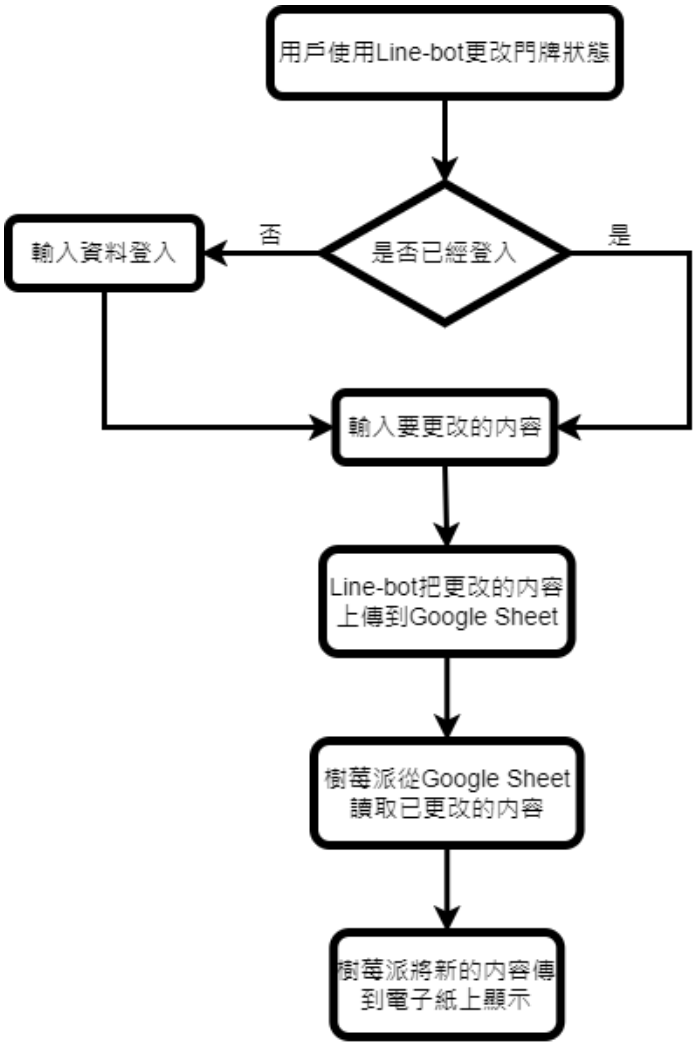
圖七、GPIO：蜂鳴器

- Button

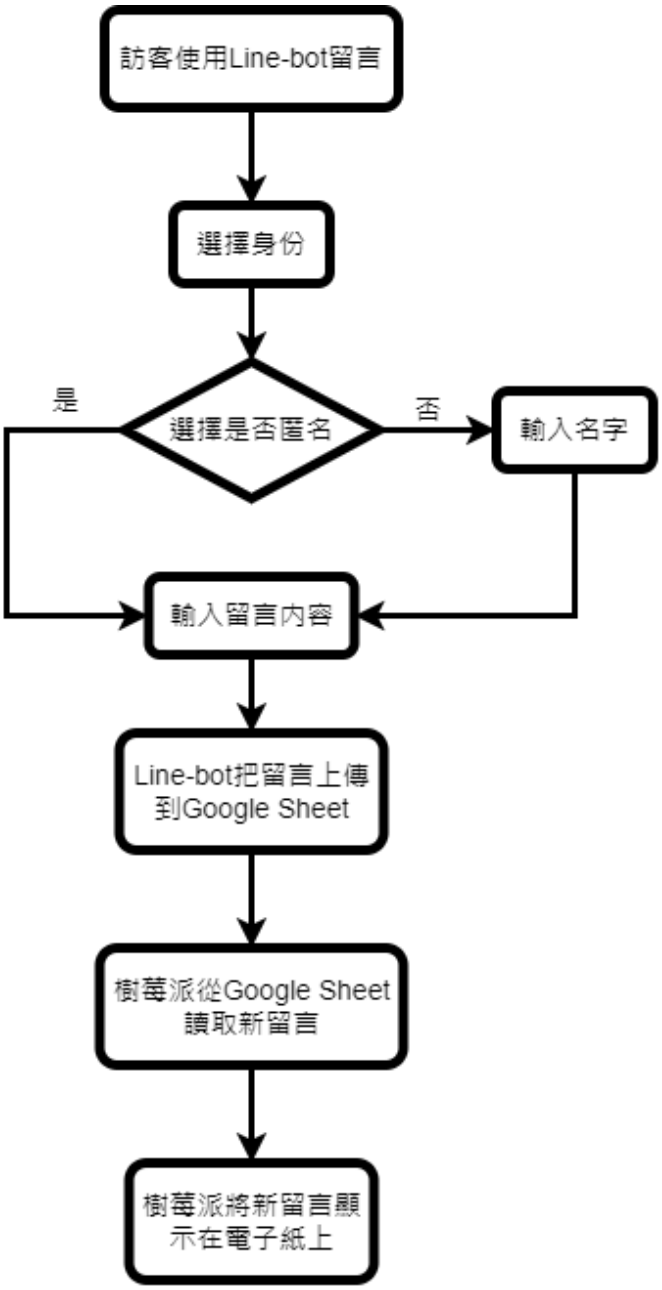
圖八、GPIO：瞬時按鈕

3. 軟硬體整合

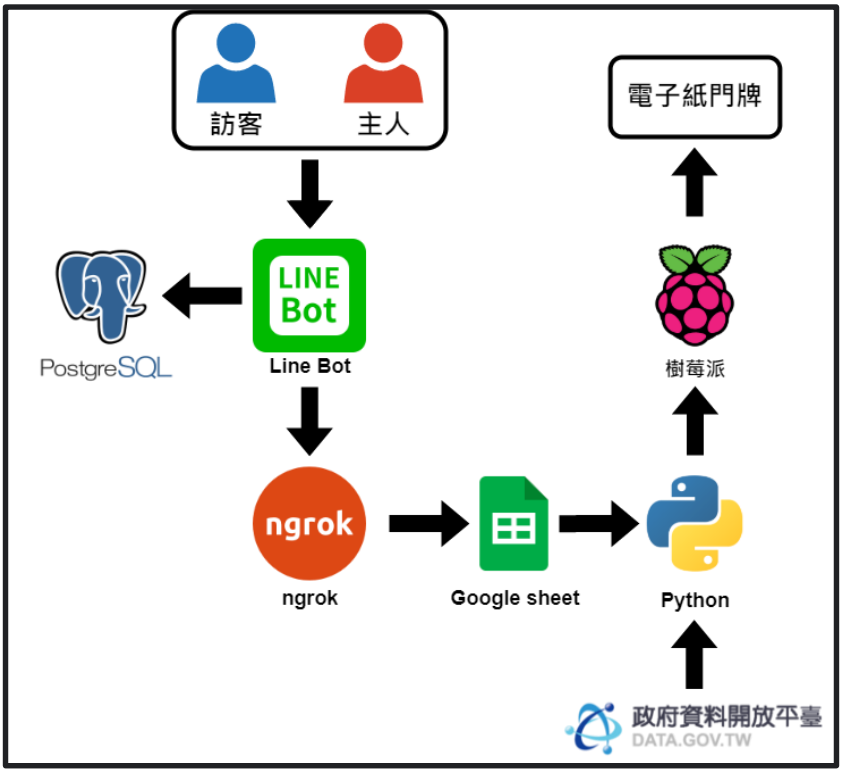
- 更改門牌狀態流程



- 新增留言流程


訪客使用Line-bot留言
選擇身份
選擇是否匿名
是
否
輸入名字
輸入留言內容
Line-bot把留言上傳到Google Sheet
樹莓派從Google Sheet讀取新留言
樹莓派將新留言顯示在電子紙上

三、 架構圖與實作方式（Architecture Diagram and Implementation Method）



圖九、電子門牌全系統架構圖

- 運用 Line-bot 實現互動介面,，提供電子門牌主人更改門牌資訊與電子門牌訪客留言給主人查看。
- 運用 Django 高級網路框架開發簡單易懂的互動介面，讓使用者無需說明文件也能輕易上手使用電子門牌。
- 使用 PostgreSQL 資料庫儲存使用者的個人基本資訊與輸入內容。
- 使用 ngrok 作為代理伺服器，連接 Localhost 和 Line-bot API 實現可修改性與提供互動性。
- 使用 Google sheet 作為資料庫，存儲留言內容與數量。
- 運用政府資料開放平臺提供的 API 來獲取未來的天氣資訊。
- 使用 Python 控制 GPIO 感測模組進行偵測與輸出，其中囊括按下 Button 替換頁面、透過 Distance Sensor 感測距離變化，當距離小於一定值時啟動 Buzzer 鳴叫編寫好之旋律。
- 運用 Raspberry Pi 3 作為主機運行 Python ，以及連接負責顯示端的 E-Paper 即時展示最新的資訊內容。

四、　模組設計描述（Module Description）

- **電子門牌頁面顯示**

　　透過 Python 影像處理套件 PIL 打開完成配置的圖片，將圖片顯示於電子紙上。

```
75  #show page 1
76  def page1():
77      Himage2 = Image.new('1', (epd.width, epd.height), 255)  # 255: clear the frame
78      bmp = Image.open(os.path.join(picdir, 'page1.bmp'))
79      bmp = bmp.resize((799,479),Image.BILINEAR)
80      Himage2.paste(bmp, (1,1))
81      mail = Image.open(os.path.join(picdir, 'mail.bmp'))
82      mail = mail.resize((80,60),Image.BILINEAR)
83      Himage2.paste(mail, (584,18))
```

圖十、ePaperCode.py

- **未來一周天氣預報資料擷取**

　　使用政府資訊開放平臺之天氣 API，將政府提供的天氣資訊傳送至 futureWeather.py，在顯示天氣頁面時 epaperCode.py 會去抓取儲存好的天氣資料並以設計好的位置與格式顯示。顯示技術則是使用電子紙套件中的 draw.text，將文字顯示於電子紙上。

```
1   import urllib.request as req
2   import json
3   url3="https://opendata.cwb.gov.tw/api/v1/rest/datastore/F-D0047-091?Authorization=rdec-key-123-45678-011121314"
4
5   with req.urlopen(url3) as res3:
6     futureWeather=json.load(res3)
7
8   keelung = futureWeather["records"]["locations"][0]["location"][12]
9   keelungName = keelung["locationName"]
10
11  tempCatch = keelung["weatherElement"][1]["time"]
12  humiCatch = keelung["weatherElement"][2]["time"]
13  weatherCatch= keelung["weatherElement"][6]["time"]
```

圖十一、futureWeather.py

```
115   #show page 2
116   def page2():
117       Himage2 = Image.new('1', (epd.width, epd.height), 255)  # 255: clear the frame
118
119       bmp = Image.open(os.path.join(picdir, 'page2.bmp'))
120       bmp = bmp.resize((799,479),Image.BILINEAR)
121       Himage2.paste(bmp, (1,1))
122
123       draw = ImageDraw.Draw(Himage2)
124       draw.text((20, 20), '未來一週天氣預報', font = font60, fill = 0)
125       draw.text((660, 35), keelungName, font = font45, fill = 0)
126       draw.text((20, 120), '今天', font = font30, fill = 0)
127       draw.text((148, 120), '明天', font = font30, fill = 0)
128       draw.text((258, 120), '後天', font = font30, fill = 0)
129       draw.text((20, 160), futureDate[0], font = font24, fill = 0)
130       draw.text((20, 200), '溫度', font = font30, fill = 0)
131       draw.text((20, 235), futureTemp[0] + '°C' , font = font24, fill = 0)
132       draw.text((20, 280), '濕度', font = font30, fill = 0)
133       draw.text((20, 315), futureHumi[0] + '%' , font = font24, fill = 0)
```

圖十二、ePaperCode.py

- **使用按鈕更換頁面**

使用 GPIO 套件讀取 Button 的高低電平信號，並使用 Button Debouncing 去判定 Button 是否被按下，判定成功後就可以更換電子門牌頁面。

```
#按鈕更換頁面
if input == GPIO.LOW and previousStatus == GPIO.HIGH and (currentTime - previousTime) > WAIT_TIME:
    previousTime = currentTime
    page = page + 1
    if page == 4:
        page = 1
    if page == 1:
        page1()
    elif page == 2:
        page2()
```

圖十三、distanceSensor.py

- **設定連線通道**

設定 Django 架構中的 ALLOWED_HOSTS，提供本地端與 Line-bot 的連線管道，其中包含 localhost、Django 預設連線通道、Ngrok 提供連往外部的通道。

```
30  # SECURITY WARNING: don't run with debug turned on in production!
31  DEBUG = True
32
33  ALLOWED_HOSTS = [
34      'localhost',
35      '127.0.0.1',
36      '3275-2402-7500-587-a6a5-ff66-3c6e-4897-4a21.ap.ngrok.io',  #允許的網域名稱
37  ]
```

圖十四、setting.py

- **Line-bot 事件處理設計**

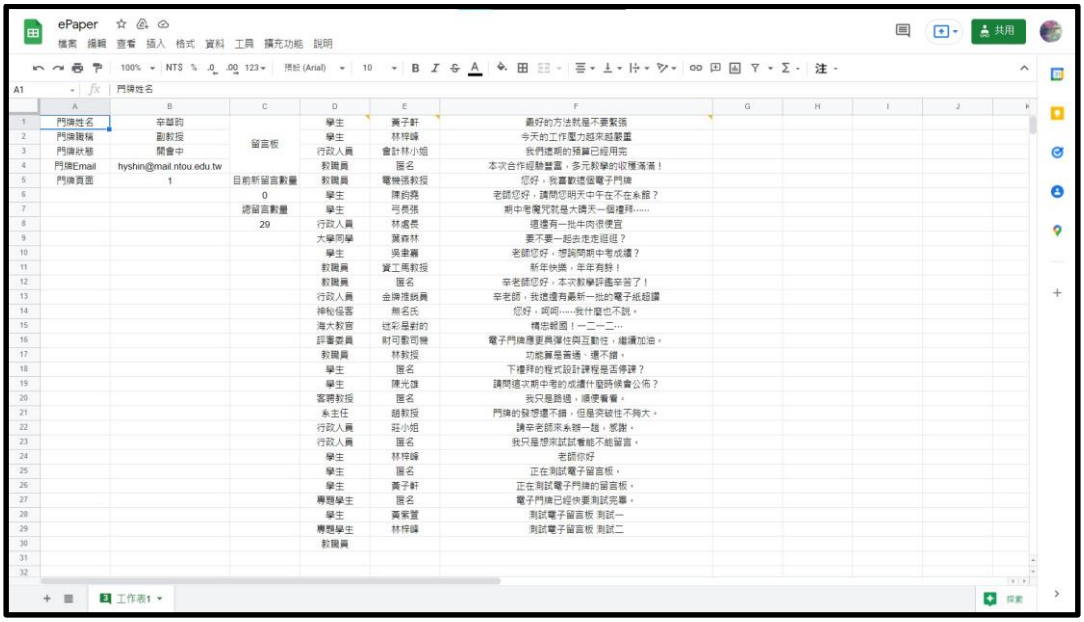
負責處理使用者從 Line-bot 端進行的所有操作，登入處理、輸入文字與按下按鈕等操作，都在 view.py 進行邏輯判斷與數值處理。

```
if isinstance(event, MessageEvent):  # 如果有訊息事件
    text = event.message.text
    uid = event.source.user_id
    profile = line_bot_api.get_profile(uid)
    lineName = profile.display_name

    if text == "!上一頁" or text == "！上一頁":

        displayPage = int(sheet.acell('B5').value)
        if 1 < displayPage and displayPage <= 3:
            sheet.update_acell('B5', str(displayPage - 1))
        else:                                          # displayPage == 1
            displayPage = 4
            sheet.update_acell('B5', str(displayPage - 1))

        saveData(saveName, saveJobTitle, saveStatus, saveEmail, displayPage-1, newMess)
        message = TextSendMessage(text=f'電子門牌第{displayPage - 1}頁面已顯示。')
        line_bot_api.reply_message(event.reply_token, message)
```

圖十五、views.py

- **Google Sheet 提供資料讀取與寫入**

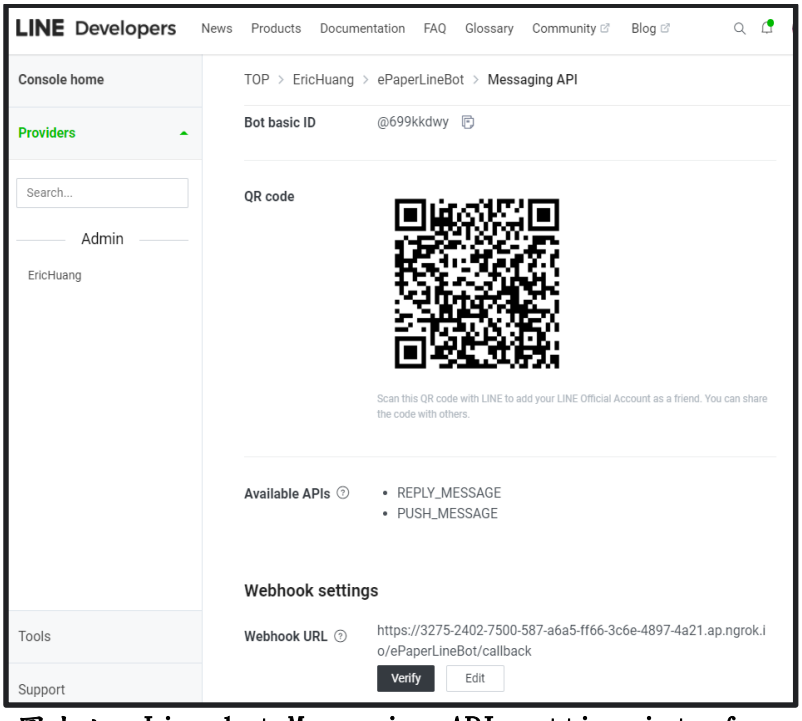
電子門牌的所有基礎資料都會在使用者輸入新的要求之後，由 Python 先寫入在 Google sheeet 裡面儲存，提供設計者做後續處理。



圖十六、Google Sheet

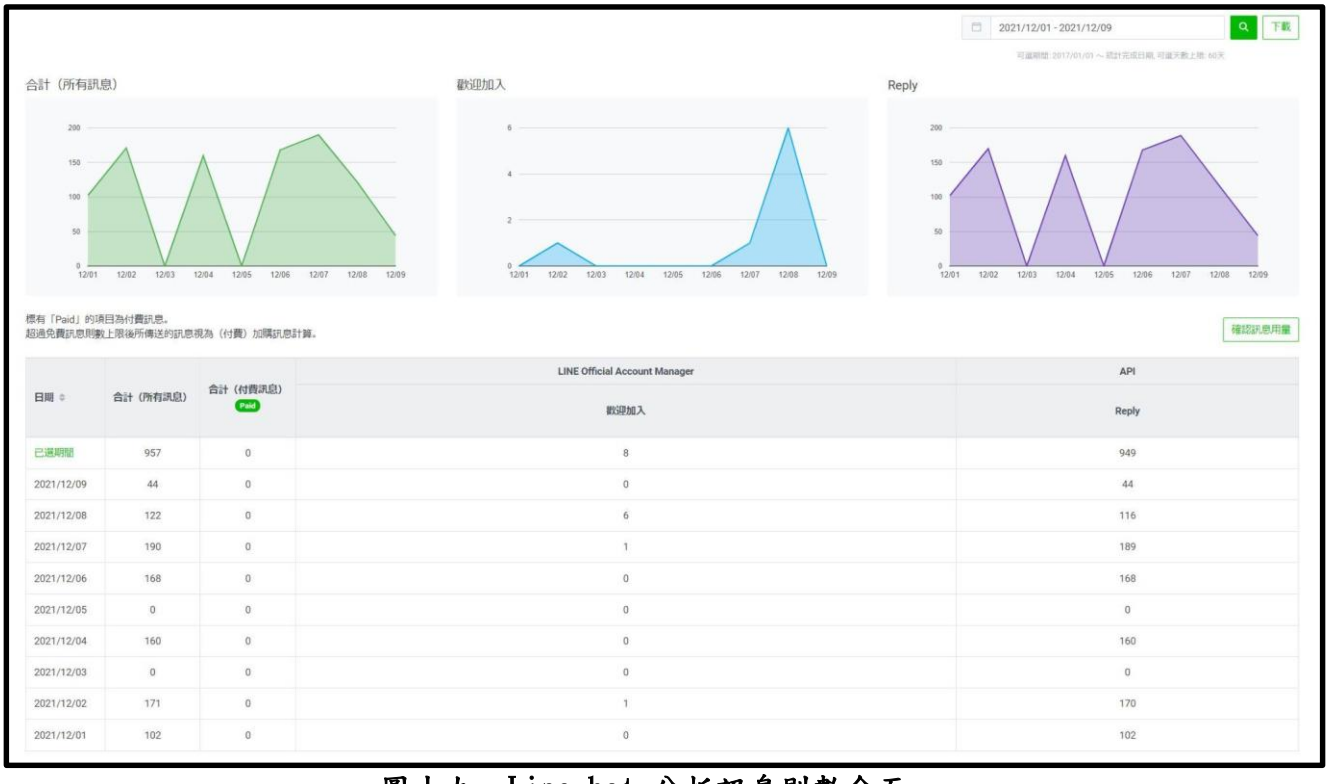
- **Line Developers 設定介面**

Line-bot 的設定界面，需要正確的設定才能使用相關功能。



圖十七、Line-bot Messaging API setting interface

圖十八、Line-bot 圖文選單設定介面



日期	合計（所有訊息）	合計（付費訊息） Paid	LINE Official Account Manager	API
			歡迎加入	Reply
已選期間	957	0	8	949
2021/12/09	44	0	0	44
2021/12/08	122	0	6	116
2021/12/07	190	0	1	189
2021/12/06	168	0	0	168
2021/12/05	0	0	0	0
2021/12/04	160	0	0	160
2021/12/03	0	0	0	0
2021/12/02	171	0	1	170
2021/12/01	102	0	0	102

圖十九、Line-bot 分析訊息則數介面

五、 實驗結果 (Experimental, Simulation Results)

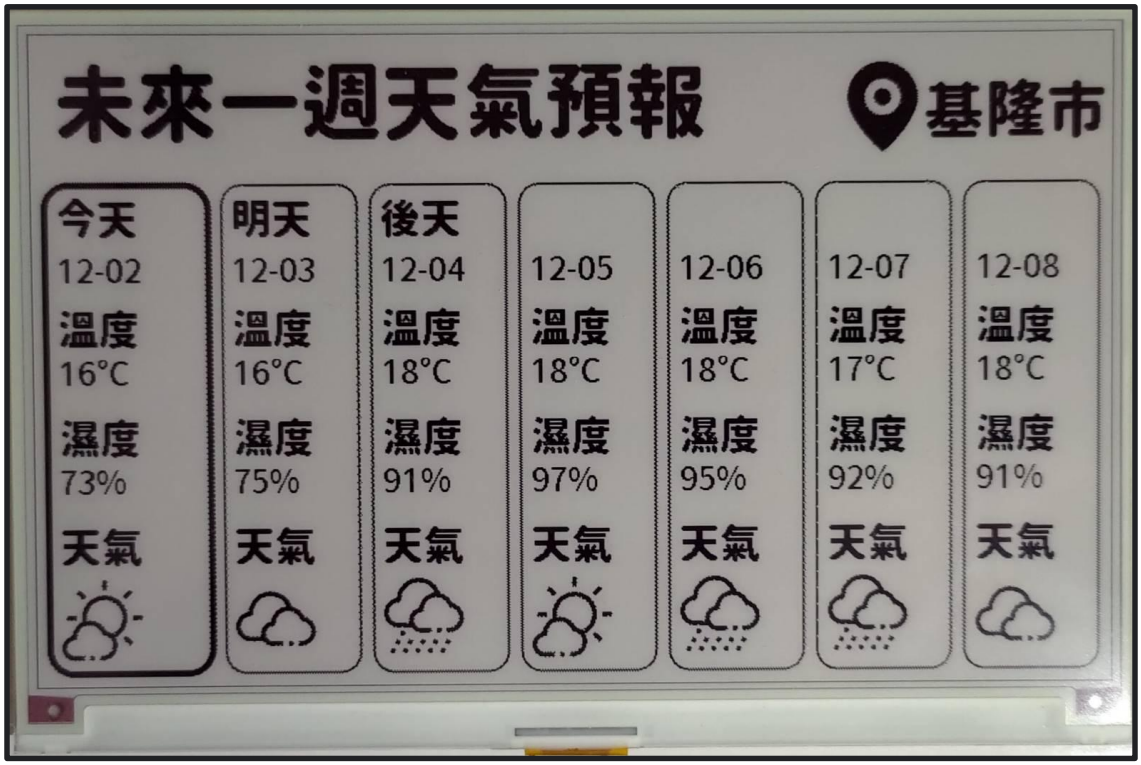
- **電子門牌第一頁面—門牌主頁面**

電子門牌第一頁面是門牌的主頁面，顯示電子門牌應當具備的基本資訊。最上方從左到右顯示了當日的日期、天氣、新留言數量與 Line-bot QR Code，中間顯示電子門牌主人的職稱、姓名與當前狀態，最下方則顯示 Email。

圖二十、電子門牌第一頁面—門牌主頁面

- **電子門牌第二頁面—天氣預報頁面**

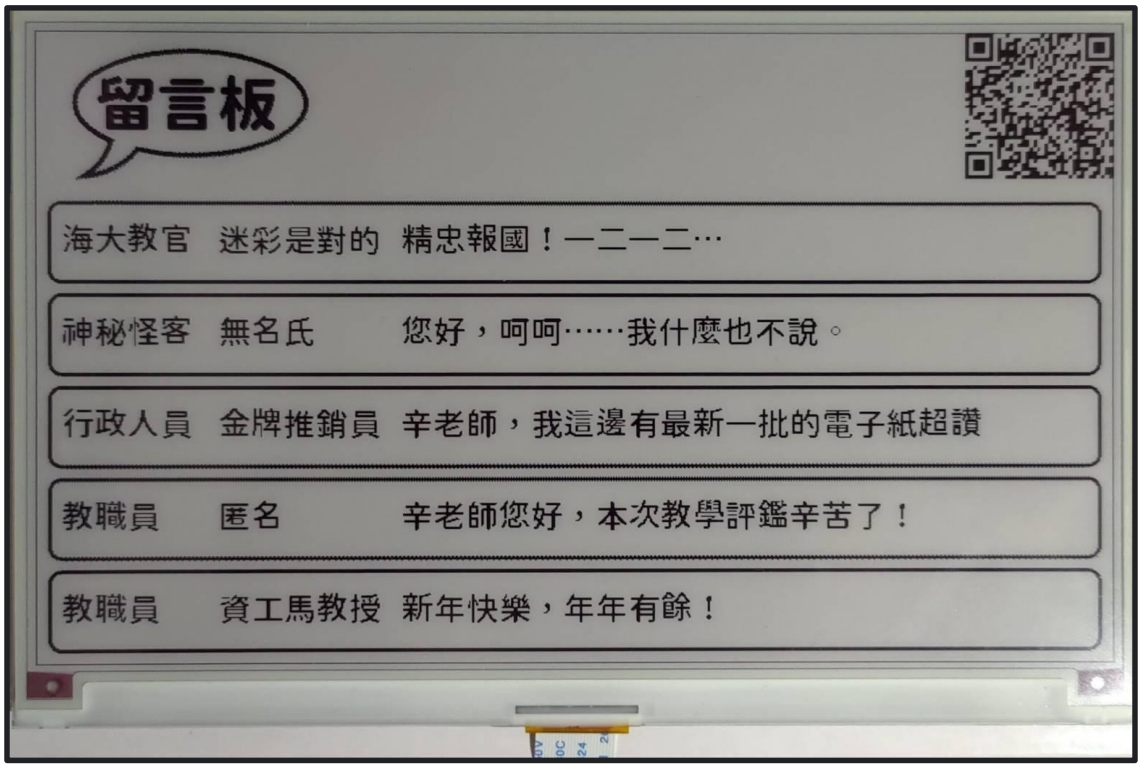
電子門牌第二頁面會顯示當前地點未來一週的天氣預報，預報中會日期、溫度、濕度以及代表天氣的圖標，被設定為在換日時會自動更新天氣預報。



圖二十一、電子門牌第二頁面—天氣預報頁面

- **電子門牌第三頁面—電子留言板頁面**

電子門牌第三頁面會顯示當前最新的五則留言，留言板會顯示留言者的身份、姓名與留言內容。



圖二十二、電子門牌第三頁面—電子留言板頁面

- **Line-bot 使用介面**

使用者透過掃描 QR Code，將 ePaperLineBot 加入好友後，即可以透過 Line-bot 與電子門牌互動。電子門牌主人需要使用【登入】功能執行初次登入、登入確認等動作之後，方可使用【更改門牌資料】功能。而電子門牌訪客則可以使用【留言】功能，選擇匿名留言或實名留言給電子門牌主人。【上一頁】/【下一頁】功能可以迅速切換電子門牌的顯示頁面。【電子紙介紹】則可以進入電子紙的維基百科、 Youtube 介紹影片與本專題的 Demo 影片，進一步了解電子紙的詳細資訊。

圖二十三、Line-bot 使用介面

六、 討論 (Discussion)

1. 問題與解決方法

- **電子紙更新速度太慢**

本來電子紙是使用三種顏色（紅、黑、白）的架構顯示畫面，但是三色架構的更新速度非常緩慢，需要使用高達三十多秒的時間才能更新完一個畫面，本專題需要經常更新畫面，不可能等待如此冗長的時序，因此團隊決定換成黑白兩色的架構，交換而來的是頁面僅需三秒就能更新完一個畫面。

- **Google Sheet 的響應時間太長**

每次從 Google Sheet 讀取或上傳資料，都需要二至三秒才能完成一筆傳輸操作，倘若資料量過多，響應時間就會被拉長，而且免費版的 Google sheet 每分鐘可存取次數也是有限的，當存取太多次，Google 短時間內就會禁止該用戶存取資料。因此團隊改良程式，設置了監聽功能，只有監聽到 Line-bot 有新需求傳入時，才會從 Google Sheet 取得最新傳輸的資料。

- **電子紙無法顯示過於精細的圖片**

電子紙使用的顯像技術，無法顯示太過精細的圖片，當團隊強行將圖片印刷上去顯示，會導致畫面形成一格格的劣質圖片，類似於像素圖，因此團隊將所有使用到的圖片都換成向量圖，雖然降低了圖片的細節程度，但是卻解決了圖片顯示在電子紙上變模糊的問題。

向量圖與點陣圖比較

點陣圖：雖然展示的細節較豐富，但放大後會看到明顯的鋸齒。

向量圖：沒辦法顯示過多細節，但即使把它放大依然清晰。

2. 未來展望

- 使用網頁爬蟲技術在教學務系統中，爬取電子門牌主人(教授)的課程資訊，並設定為在上課期間自動更改電子門牌的狀態。
- 透過串接 NLP 分析的 API 自動分析語意，從使用者輸入的字句解析其真正的需求。
- 電子留言板的留言經過 Line-Bot 推播給電子門牌主人(教授)的 Line。

七、 結論 (Conclusion)

- 一開始我們選用 Google Sheet 存放資料的確較為簡單與方便，卻在過程中發覺每次傳遞資料時，都需要耗費過多時間成本在 Raspberry Pi 與 Google Sheet 的網路連接上，因此我們決定在 PostgreSQL 也存放一些資料來改善此情況，而在未來我們也會嘗試在主機上建置獨立的資料庫，除了能將複雜的資料系統化地存放之外，亦能有條理地繼續擴充資料的豐富度，還能有效改善響應時間過長之問題。

- 而為了使 E-Paper 以最快速度顯示正確的更新資訊，我們也不斷嘗試替換傳輸的方法、順序及時機，甚至為了測試 E-Paper 印刷時間與主機相容的程度，我們先後也以 Raspberry Pi 2、LinkIt 7697 作為主機進行研究。在未來透過串接 NLP 分析的 API，可望讓 Line-bot 兼具智慧與更人性化的服務，提升使用者的對話體驗，並於對話中擷取出要點，在任何對話情境下有效地判斷使用者的真實需求。

附件（Appendices）

程式碼：電子門牌程式碼 - Github

Demo 影片：電子門牌 Demo - YouTube

參考資料與文獻（References）

1. 電子紙說明文件

   7.5inch e-Paper HAT (B) - Waveshare Wiki

2. gspread 使用說明

   gspread 5.0.0 documentation

3. 政府資料開放平臺

   政府資料開放平臺| 政府資料開放平臺

4. Python：透過 API 擷取網站資料

   輕鬆學習 Python：透過 API 擷取網站資料

5. Python：讀取、產生 JSON 格式資料教學與範例

   Python 讀取、產生 JSON 格式資料教學與範例

6. Python + Line Bot + PostgreSQL 教學：6 步驟快速上手 Line 機器人

   [Python+LINE Bot 教學]6 步驟快速上手 LINE Bot 機器人/python-line-bot.html

   [Python+LINE Bot+PostgreSQL 教學]一篇搞懂 LINE Bot 讀取資料庫的方法

   [Django 教學]Django 連接 PostgreSQL 資料庫教學

7. 用 Django 架構建置專屬的 LINEBOT

   https://ithelp.ithome.com.tw/articles/10239404

   https://ithelp.ithome.com.tw/articles/10240615

   https://ithelp.ithome.com.tw/articles/10241291

8. Python 與 LINE Bot 機器人全面實戰特訓班：Flask 最強應用