Auto-check 無人商店

Team 20 吳岱容 杜葳葳 陳騰鴻

Outline

- Introduction
- Demo
- 流程架構
- 實作介紹

Introduction

Motivation

- for 顧客：減去等候結帳的時間、提升購物體驗、個人化推薦
- for 商家：降低人力成本、提升庫存管理效率

Introduction

「**無人商店雲端系統**」，以「個人帳號」為中心的雲端服務，為顧客創造**客製化的購物體驗**。此外，系統會監控架上存貨，**即時通知供貨商補貨**，確保顧客都能買到想要的商品。最後，我們利用過去的購買紀錄，建置個人化的推薦系統。

Features

Auto-check 無人商店

顧客

- 拍照、QRcode註冊
- 個人購物紀錄頁面
- 掃臉直接購物
- 客製化商品推薦

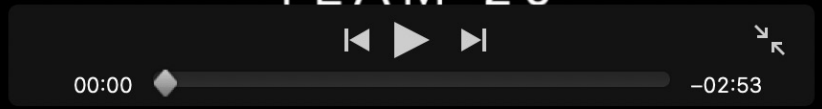
商家

- 庫存管理
- 會員資料管理
- E-mail自動通知廠商補貨
- 購物流程全自動化、店裡面不用雇店員

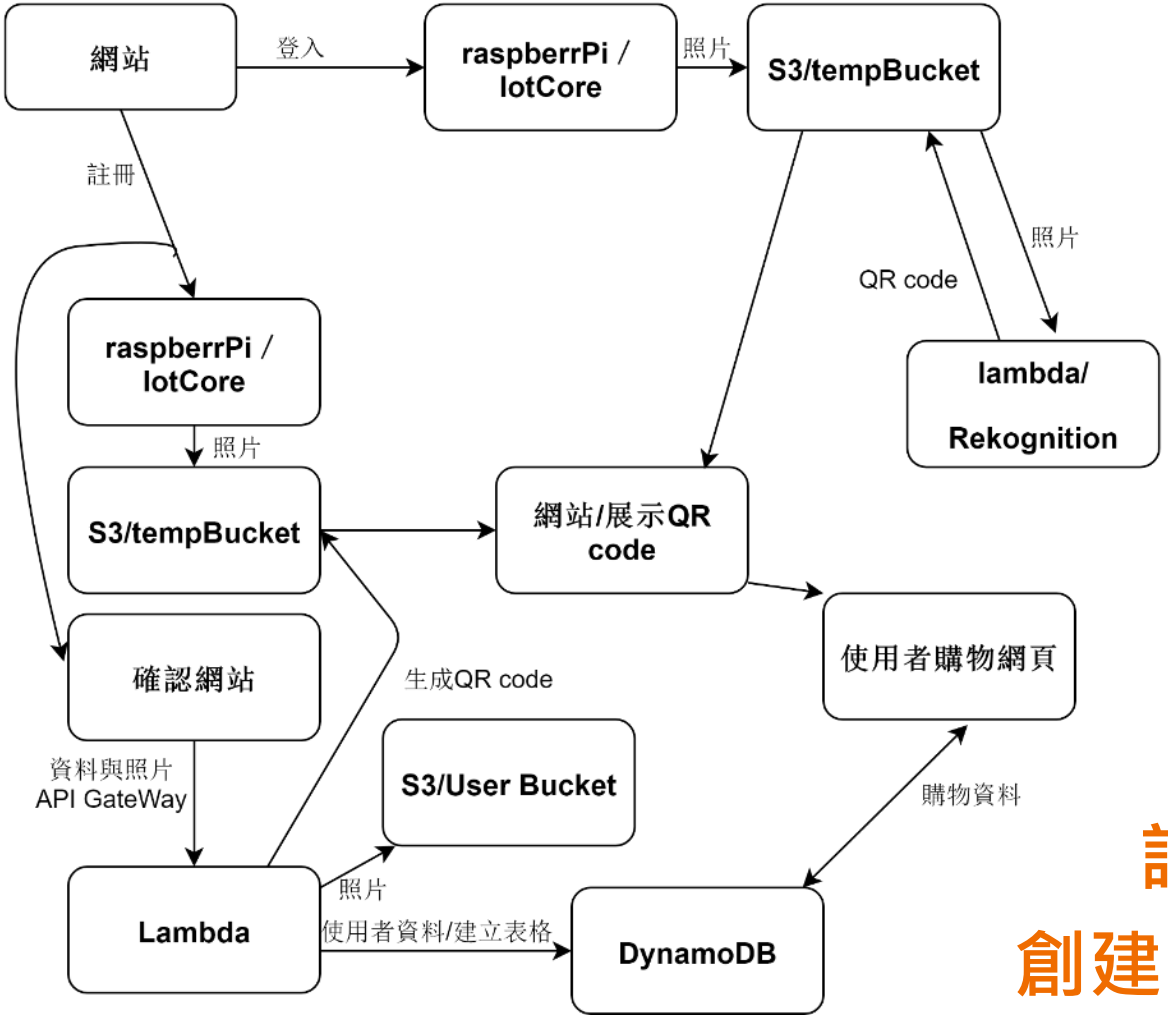
Demo

CLOUD PROGRAMMING
FINAL PROJECT
AUTO-CHECK

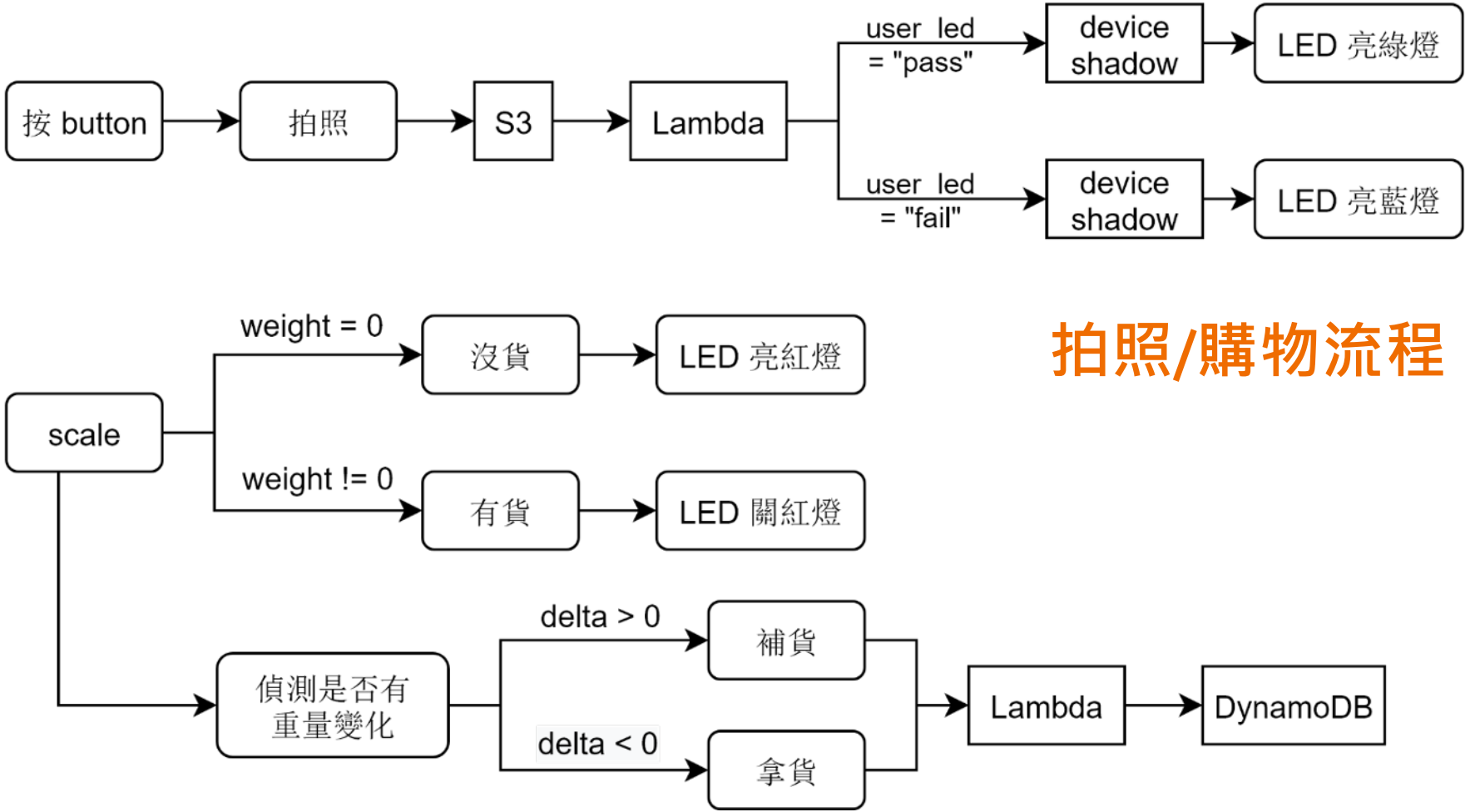
TEAM 20



流程架構


網站
登入
raspberrPi / IotCore
照片
S3/tempBucket
註冊
raspberrPi / IotCore
照片
S3/tempBucket
網站/展示QR code
QR code
照片
lambda/ Rekognition
使用者購物網頁
確認網站
生成QR code
資料與照片 API GateWay
S3/User Bucket
購物資料
Lambda
照片
使用者資料/建立表格
DynamoDB

註冊/登入
創建個人頁面

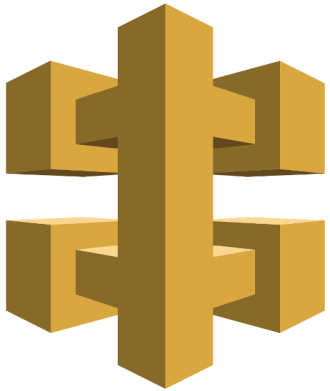

按 button
拍照
S3
Lambda
user_led
= "pass"
device
shadow
LED 亮綠燈
user_led
= "fail"
device
shadow
LED 亮藍燈
scale
weight = 0
沒貨
LED 亮紅燈
weight != 0
有貨
LED 關紅燈
偵測是否有
重量變化
delta > 0
補貨
delta < 0
拿貨
Lambda
DynamoDB

拍照/購物流程

實作介紹

API Gateway

- 主要工作：
  - **傳遞資訊（serverless架構），負責和DynamoDB、Lambda溝通**
  - 共用了三個資源分別接到兩個Lambda Function及DynamoDB
- 優點：
  - 沒有server的架構，維護上有便利之處
  - api gateway及lambda**減少server idle的cost**
- 缺點：
  - 功能限制較大

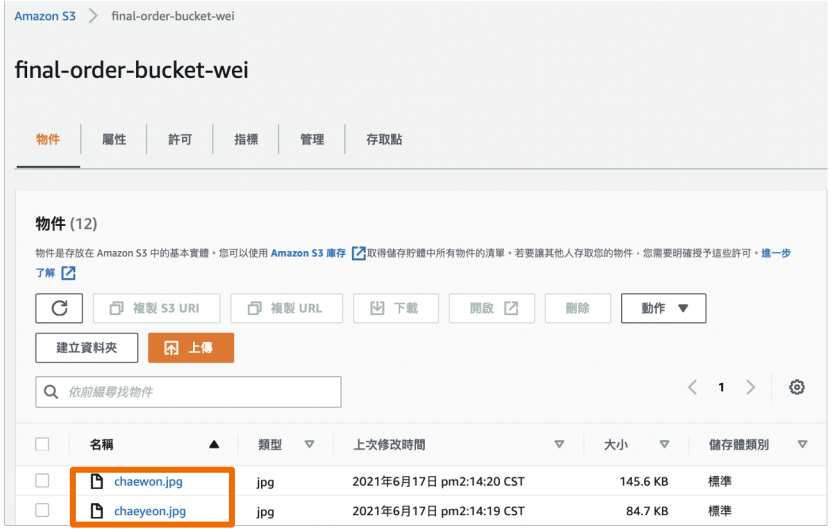


S3

- **會員** (user bucket)
  - 照片名稱是 user_id
- **暫存** (temp bucket)
  - 按button → Pi → S3
    - 會員註冊mode: → Lambda → S3/DynamoDB
    - 拿商品mode: → Lambda → Recognition 判斷是不是會員/是哪個會員

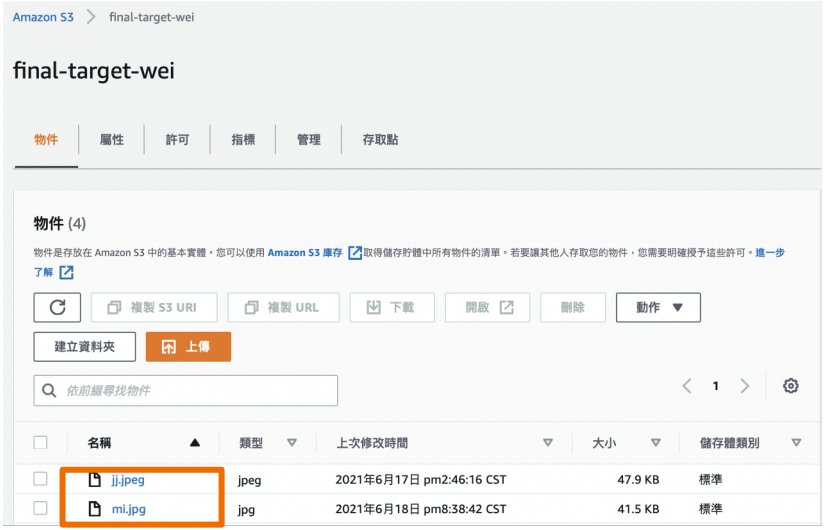
S3

會員 (user bucket)



照片名稱是user_id

暫存 (temp bucket)



照片名稱隨意

DynamoDB

名稱	狀態	分區鍵	排序鍵	索引
chaewon	作用中	timestamp (數字)	-	0
item_1	作用中	timestamp (數字)	-	0
item_2	作用中	timestamp (數字)	-	0
minju	作用中	timestamp (數字)	-	0
nako	作用中	timestamp (數字)	-	0
stock	作用中	item_id (字串)	-	0
user	作用中	user_id (字串)	-	0

- 會員資料 (user)
  - key: user_id, value: email
- 每個會員的購物紀錄 ([user_id])
  - key: timestamp, value: item_id, count, isPay
- 目前存貨 (stock)
  - key: item_id, value: price, stock
- 每個商品的記錄 ([item_id])
  - key: timestamp, value: count

DynamoDB 會員資料

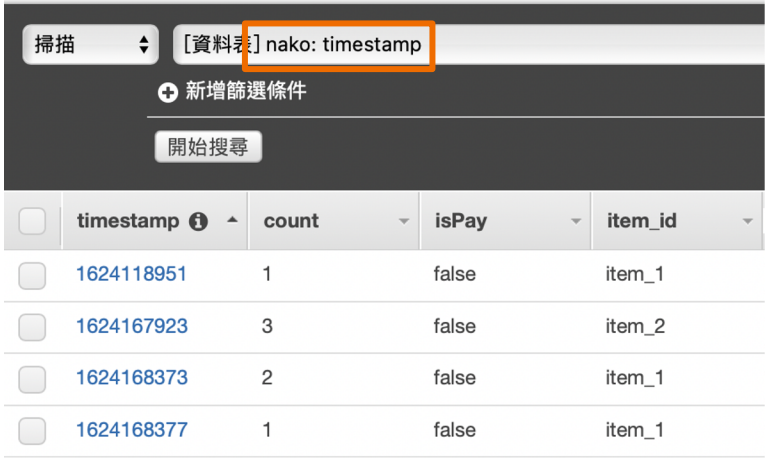
會員資料 (user)

key: user_id, value: email

user_id	email
chaewon	chaewon@gmail.com
minju	minju@gmail.com
nako	dora1227@pchome.com.tw

每個會員的購物紀錄 ([user_id])

key: timestamp, value: item_id, count, isPay



timestamp	count	isPay	item_id
1624118951	1	false	item_1
1624167923	3	false	item_2
1624168373	2	false	item_1
1624168377	1	false	item_1

DynamoDB 貨品資料

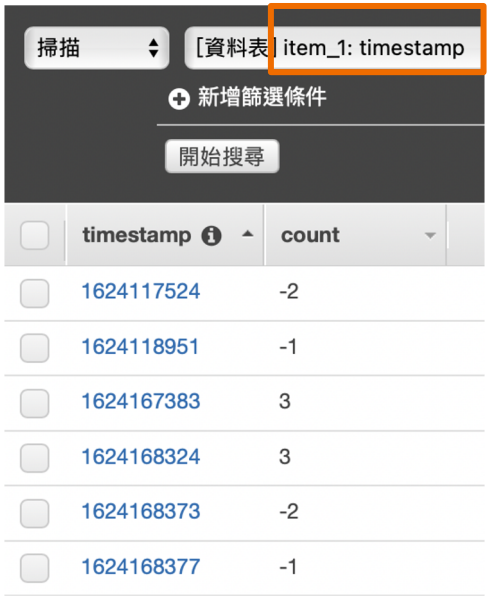
目前存貨 (stock)

key: item_id, value: price, stock

item_id	price	stock
item_1	20	0
item_2	10	3

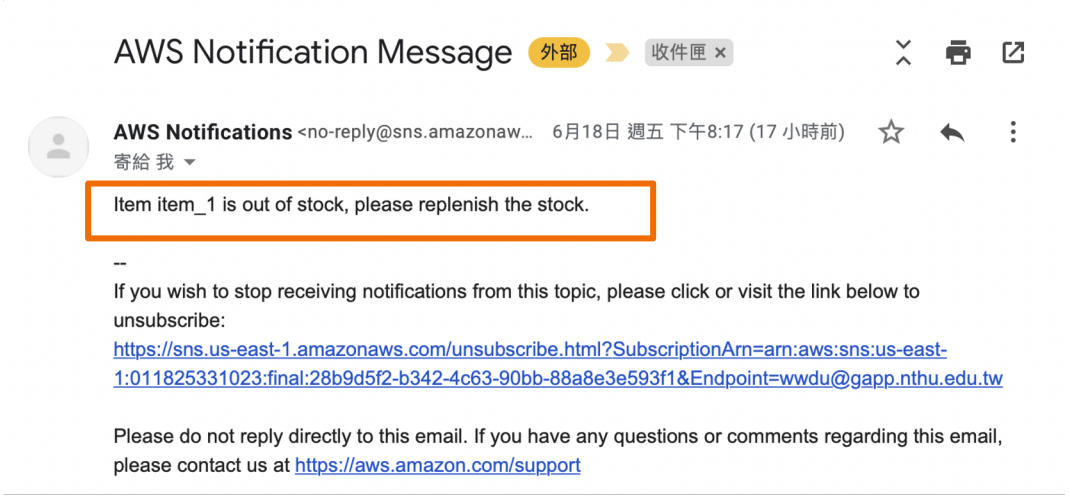
每個商品的記錄 ([item_id])

key: timestamp, value: count



SNS

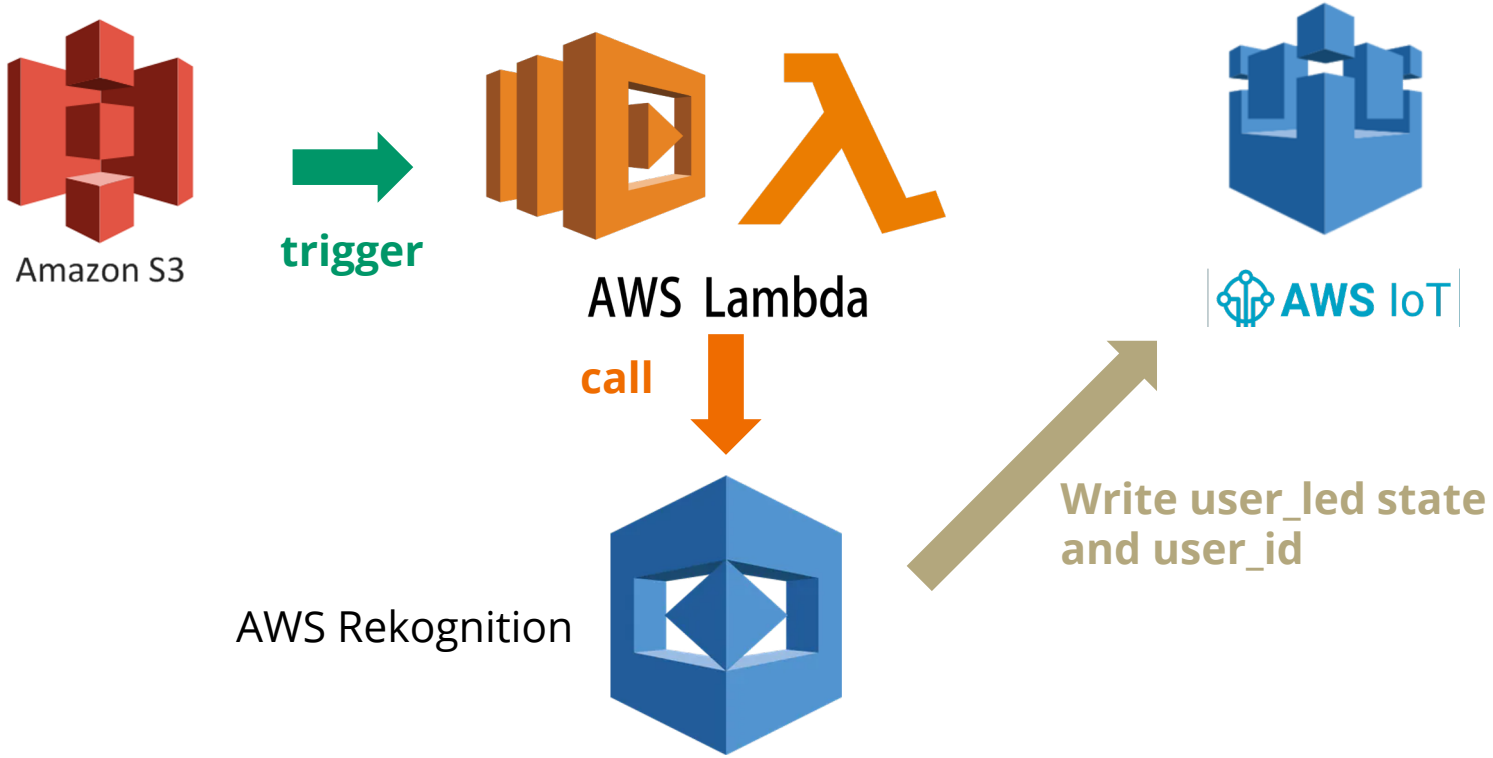
- 當架上某個商品**沒貨**的時候寄信通知廠商
  - IoT core → Lambda → DynamoDB → **if (stock=0)** → SNS



AWS Rekognition

- 主要工作：
  - **登入**時的人臉識別
  - **購物**時的人臉識別
- 演算法：
  - 從 trigger event 取得目標辨識照片 **T**
  - 依序將每一張會員 bucket 的照片 **S** 與 **T** 作比較：**在 T 中尋找 S 裡最大的臉**
  - 若成功配對一組臉，從 **S** 的檔案名稱提取 user_id

Lambda: 使用者人臉識別



Lambda: 庫存追蹤和購買紀錄

- **IoT Rule** 在 Device Shadow 更新時 trigger Lambda
  - `SELECT * FROM '$aws/things/final/shadow/update/accepted'`
- Lambda 從 Device Shadow 讀取資訊並寫入 databases
- 若缺貨， Lambda 使用 SNS 服務寄通知信

AWS IoT core

- Device Shadow
  - **AWS** (S3 → Recognition → Lambda) → **Pi**
    - **user_id**: 會員的id
    - **user_led**: 控制燈號(green/blue)
  - **Pi** → **AWS** (Lambda → DynamoDB)
    - **need_update**: 紀錄需不需要寫入 database
    - **user_id**: 會員的id(或店員的id)
    - **item_id**: 買的商品(或補貨的商品)
    - **count**: 買的數量(或補貨的數量)
      - 註: 補貨數值是+, 買貨數值是-

AWS IoT core

影子狀態：

```
{
  "desired": {
    "welcome": "aws-iot",
    "item_id": "item_1",
    "count": 2,
    "user_led": "pass",
    "need_update": "False",
    "user_id": "nako"
  },
  "reported": {
    "welcome": "aws-iot",
    "item_id": "item_1",
```

拿(補)商品名稱

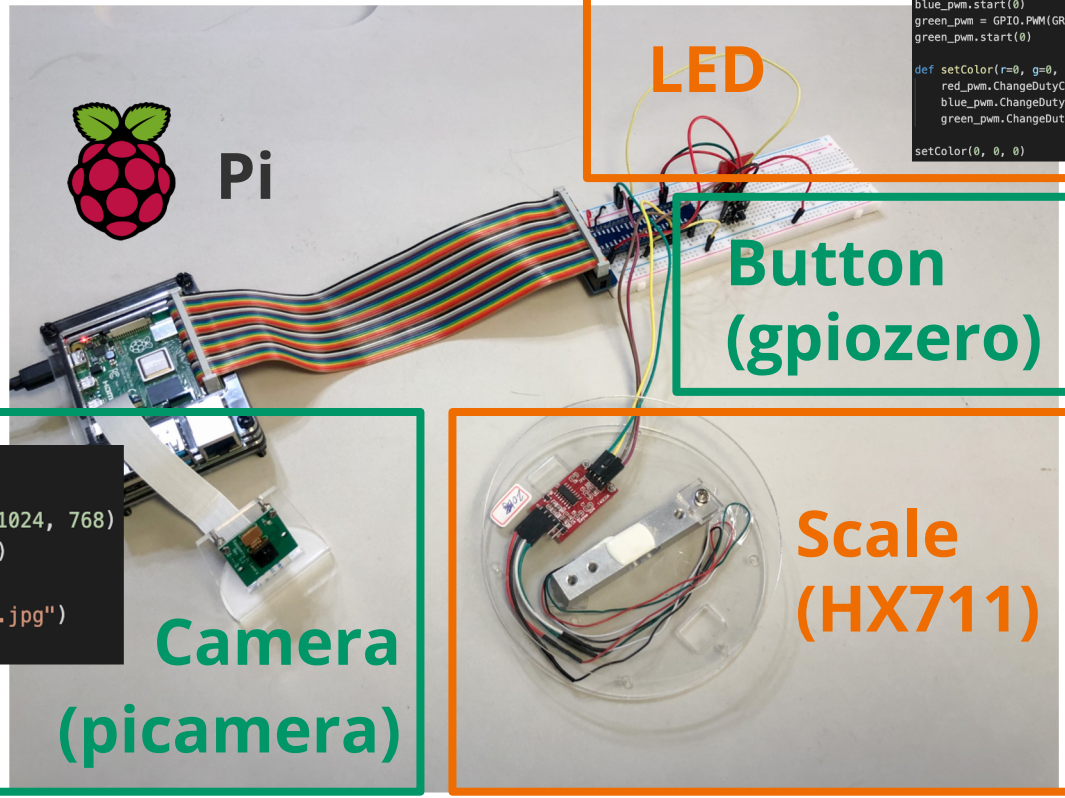
拿(補)多少商品

控制是否為會員的燈號

是否需要更新DynamoDB

會員的名字

Raspberry Pi



LED

```
RED_LED_PIN = 16
BLUE_LED_PIN = 20
GREEN_LED_PIN = 25
PWM_FREQ = 200

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(RED_LED_PIN, GPIO.OUT)
GPIO.setup(BLUE_LED_PIN, GPIO.OUT)
GPIO.setup(GREEN_LED_PIN, GPIO.OUT)

red_pwm = GPIO.PWM(RED_LED_PIN, PWM_FREQ)
red_pwm.start(0)
blue_pwm = GPIO.PWM(BLUE_LED_PIN, PWM_FREQ)
blue_pwm.start(0)
green_pwm = GPIO.PWM(GREEN_LED_PIN, PWM_FREQ)
green_pwm.start(0)

def setColor(r=0, g=0, b=0):
    red_pwm.ChangeDutyCycle(100-int(r/255*100))
    blue_pwm.ChangeDutyCycle(100-int(b/255*100))
    green_pwm.ChangeDutyCycle(100-int(g/255*100))

setColor(0, 0, 0)
```

Button (gpiozero)

```
button = Button(17) # GPIO17
button_is_pressed = 0 # init t
```

```
if button.is_pressed:
    button_is_pressed = 1
```

Camera (picamera)

```
if button_is_pressed:
    camera = PiCamera()
    camera.resolution = (1024, 768)
    camera.start_preview()
    sleep(2)
    camera.capture("image.jpg")
    camera.close()
```

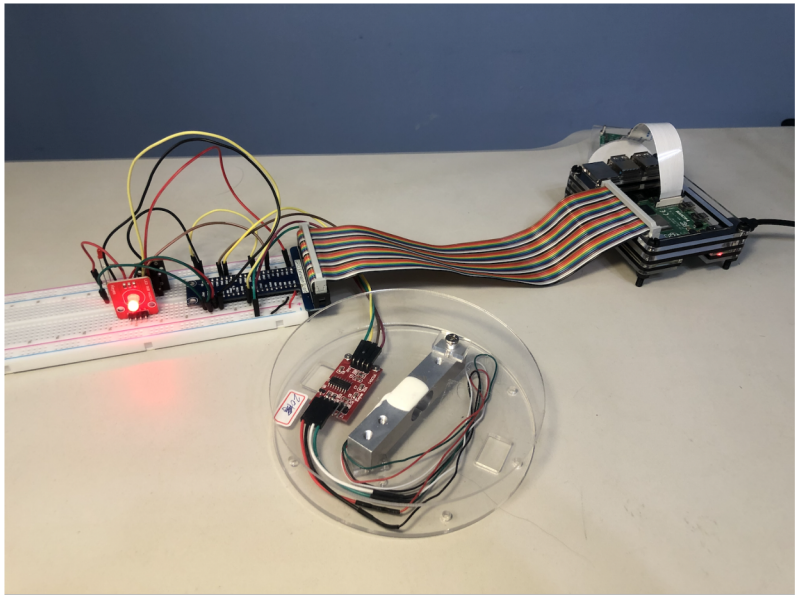
Scale (HX711)

```
EMULATE_HX711=False
referenceUnit = 1
hx = HX711(5, 6)
hx.set_reading_format("MSB", "MSB")
hx.set_reference_unit(referenceUnit)
hx.reset()
hx.tare()
print("Tare done! Add weight now...")
```

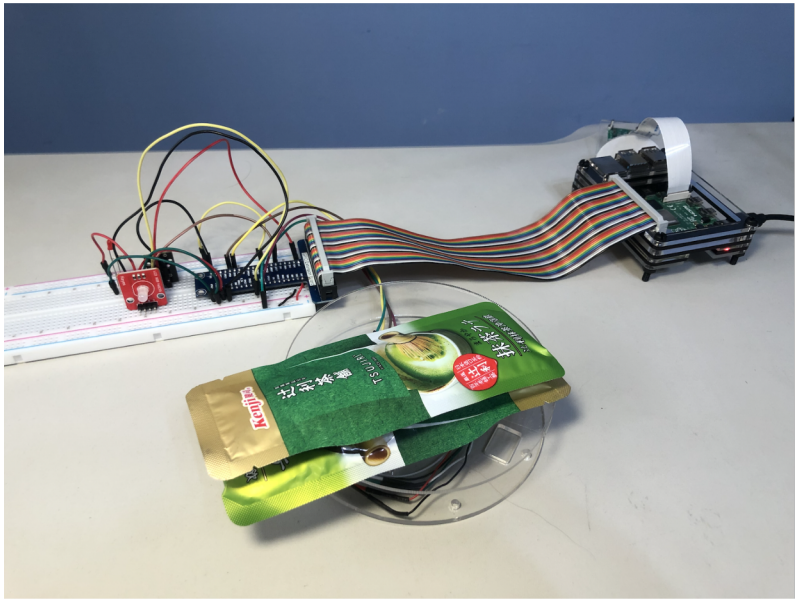
```
val = hx.get_weight(5)
hx.power_down()
hx.power_up()
time.sleep(0.1)
```

Raspberry Pi

缺貨（亮紅燈）



有貨（紅燈熄滅）



AWS Personalize

recommendation_system

user_id	item_id	count	timestamp
1	1	5	874965758
1	2	3	876893171
1	3	4	878542960
1	4	3	876893119
1	5	3	889751712
1	7	4	875071561
1	8	1	875072484
1	9	5	878543541
2	1	4	888550871
6	1	4	883599478
6	7	2	883599102
6	8	4	883600657

Recommendations

Recommendation ID

RID-abf34aac-48fb-4f0d-89cc-f9a8afdf64e3

Item ID	Score
235	0.0090934
70	0.0077030
402	0.0054251
234	0.0046498
640	0.0039225
949	0.0038401

大

小

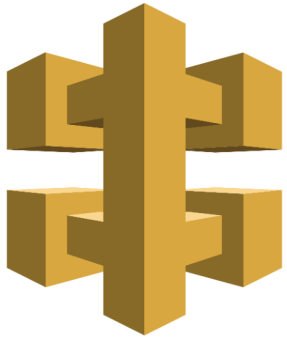
Test campaign results

User ID Info

This is the user ID of the user you want to see campaign results for. This user ID needs to be obtained from your user-interactions or user dataset.

1

SNS



Amazon
API Gateway

aws

Summary

Amazon S3

Amazon DynamoDB

Rekognition

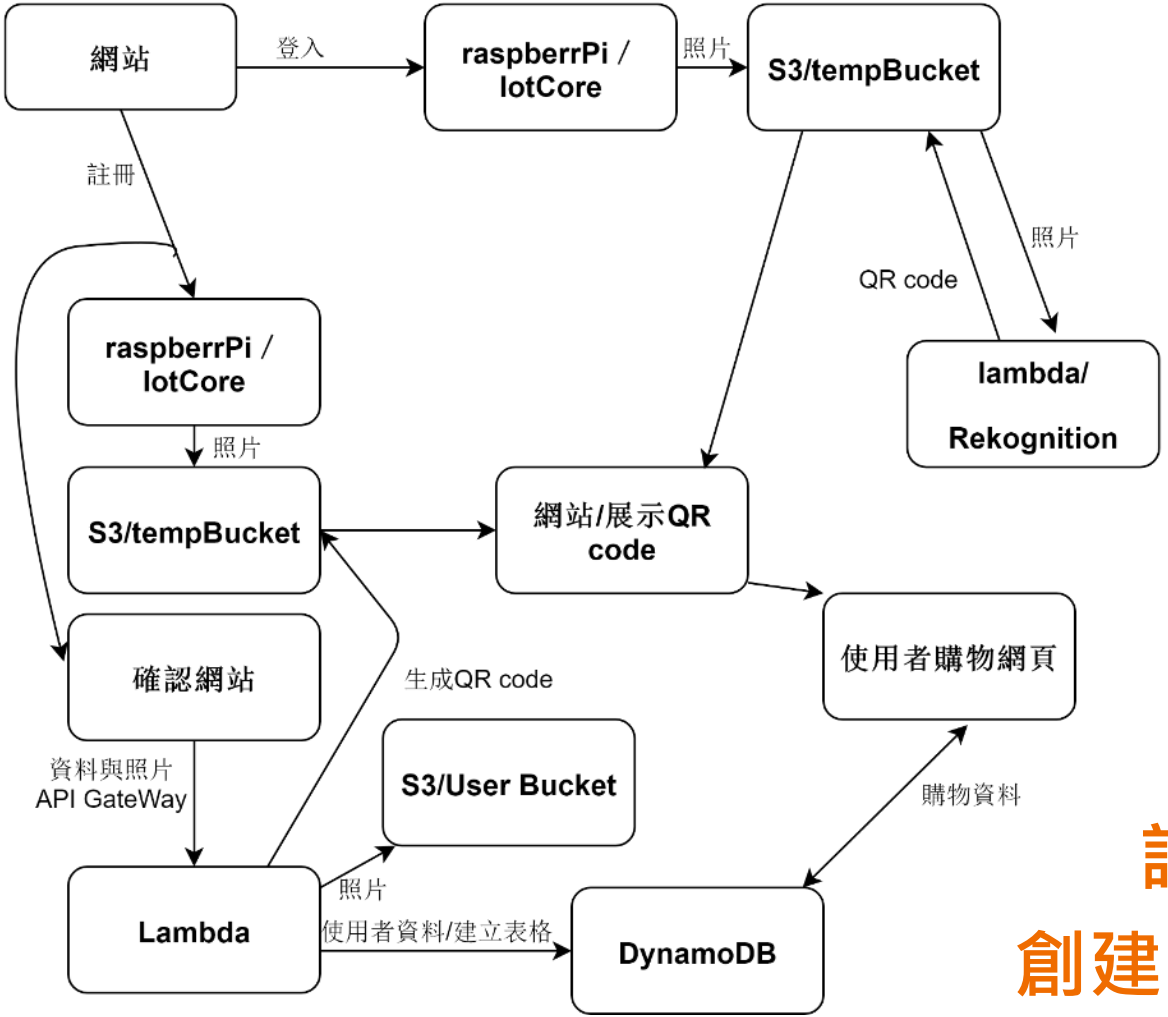
AWS Lambda

AWS IoT core

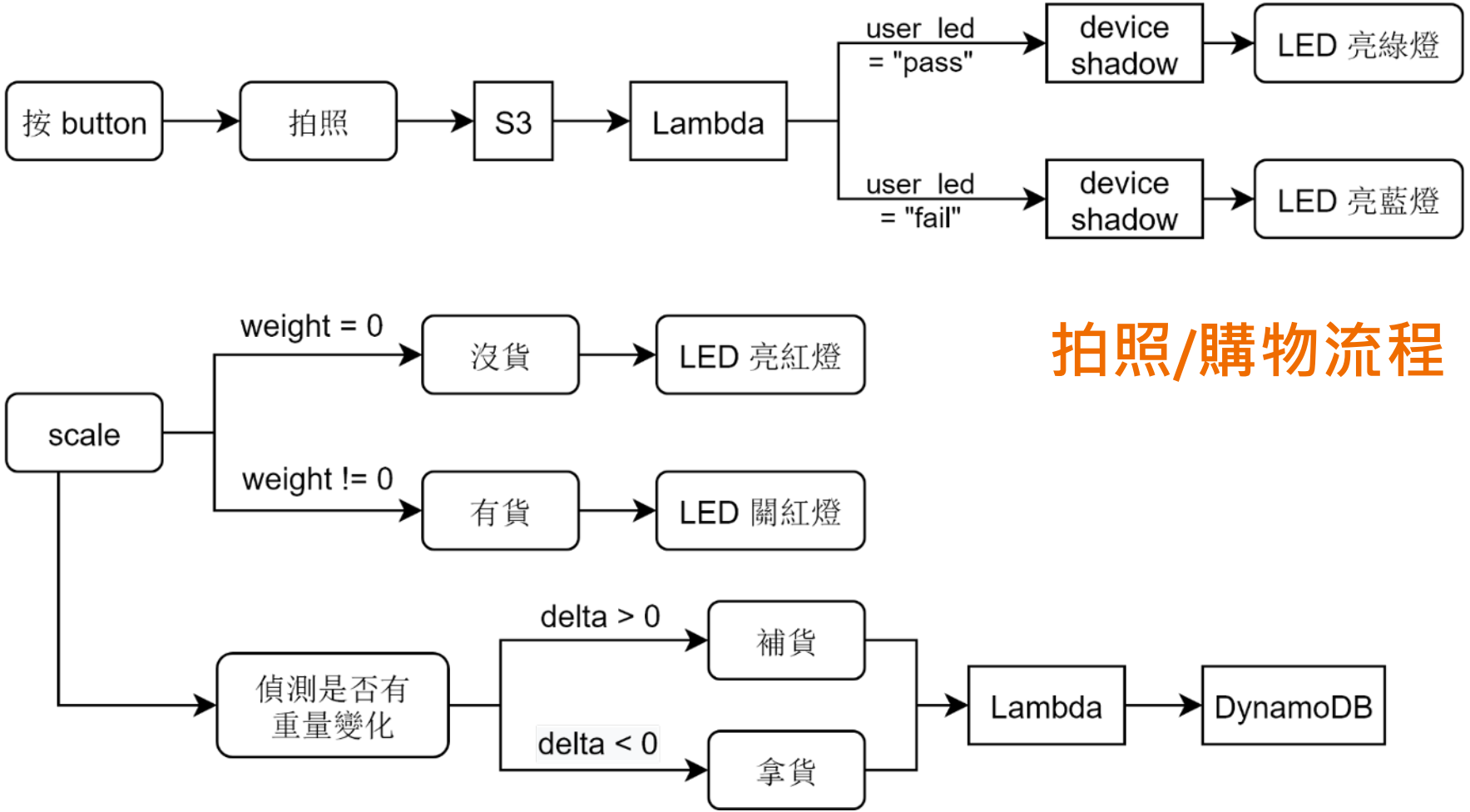
AWS Personalize

Thank you

Team 20


網站
登入
raspberrPi / IotCore
照片
S3/tempBucket
註冊
raspberrPi / IotCore
照片
S3/tempBucket
網站/展示QR code
QR code
照片
lambda/ Rekognition
使用者購物網頁
確認網站
生成QR code
資料與照片 API GateWay
S3/User Bucket
購物資料
Lambda
照片
使用者資料/建立表格
DynamoDB

註冊/登入
創建個人頁面


按 button
拍照
S3
Lambda
user_led
= "pass"
device
shadow
LED 亮綠燈
user_led
= "fail"
device
shadow
LED 亮藍燈
scale
weight = 0
沒貨
LED 亮紅燈
weight != 0
有貨
LED 關紅燈
偵測是否有
重量變化
delta > 0
補貨
delta < 0
拿貨
Lambda
DynamoDB

拍照/購物流程