

期末專題報告



Pricer電子標籤API功能整合應用

PYT357102 洪嘉興

2022.06.12



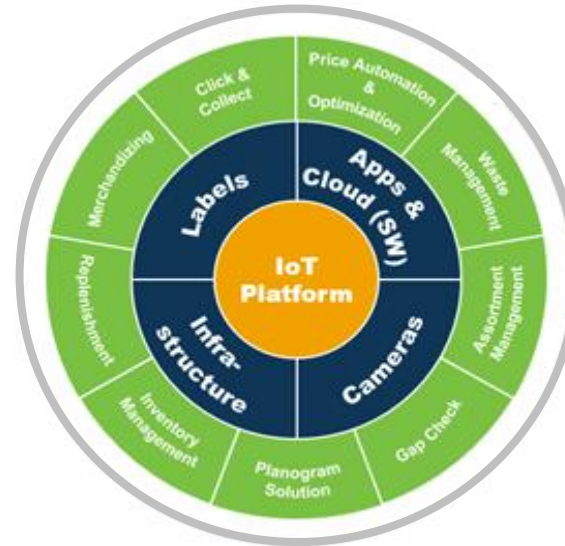
NTU CSIE

專題報告概要

- **Pricer系統架構**
- 整合功能說明
- 程式架構說明
- 實際結果展示

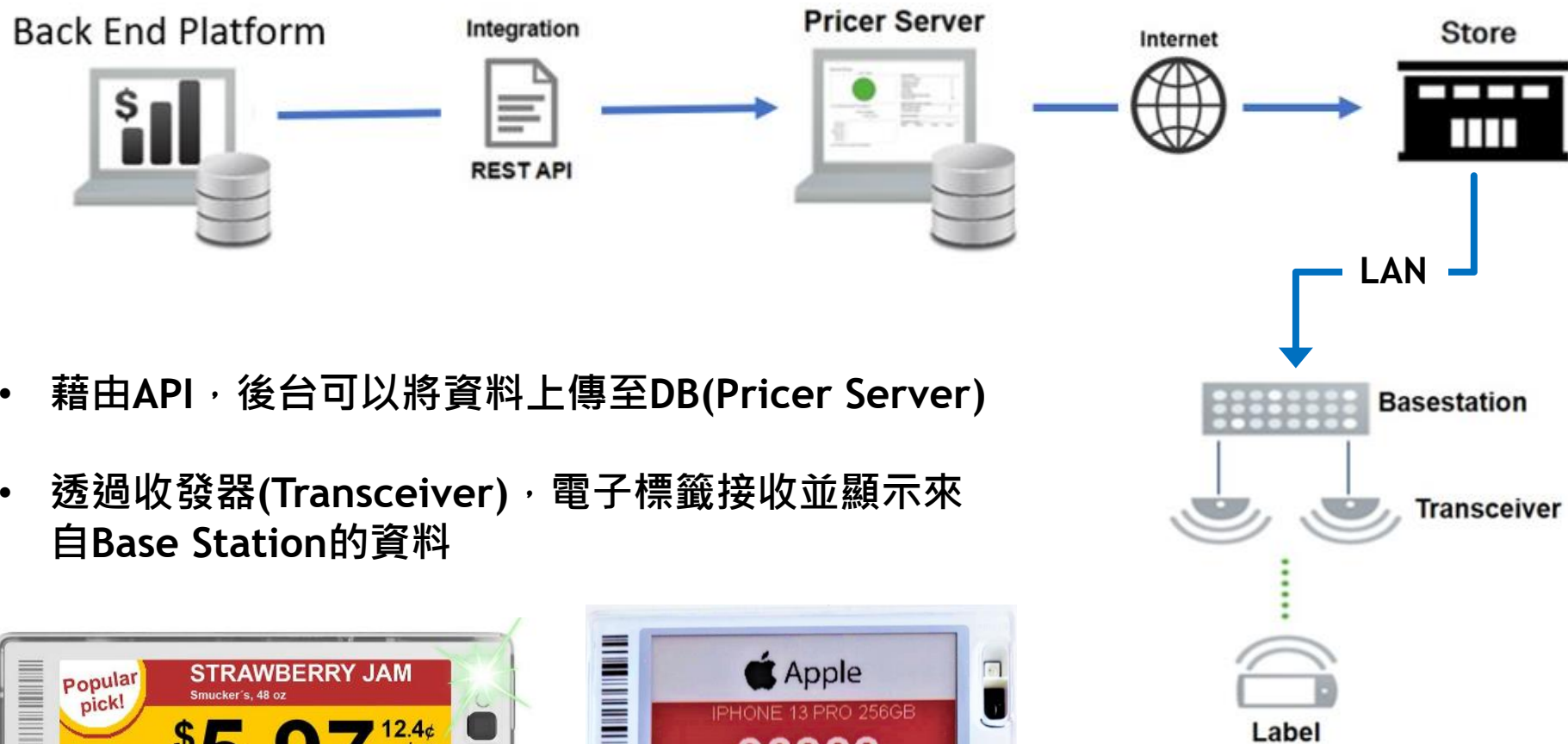
Pricer in Brief

Pricer was founded in Sweden in 1991 and is nowadays a global leader in providing in-store digital shelf-edge solutions that enhance both store performance and the shopping experience.



資料來源: <https://www.pricer.com/>

Pricer 系統架構



- 藉由API，後台可以將資料上傳至DB(Pricer Server)
- 透過收發器(Transceiver)，電子標籤接收並顯示來自Base Station的資料



整合功能說明

- **應用場景：**當單筆或多筆資料(ex: 商品價格)上傳時，可以比對DB既有數據並同步啟動閃燈模式，有助於確認資料更新狀況(ex: 即時商品促銷)
- **功能說明：**
 - 讀取上傳之資料(*.csv)
 - 與既有之後台數據比對確認是否有更新
 - 使用者確認後將資料更新至後台並顯示於電子標籤
 - 啟動閃燈模式提示有更新之商品
- **Pricer既有API功能：**
 - 將資料上傳至Server並顯示於電子標籤
 - 由後台取出特定SKU(item code)之數據
 - 針對特定標籤(Barcode)或SKU(item code)啟動閃燈
 - API以JSON格式編輯

PATCH

/api/public/core/v1/items Update or add multiple items

GET

/api/public/core/v1/items Get multiple items

POST

/api/public/core/v1/flash/items Flash items

```
{
  "configuration": {
    "color": "#ff00ff",
    "duration": 4,
    "flashType": 10,
    "realTime": true
  },
  "itemIds": [
    "string"
  ]
}
```

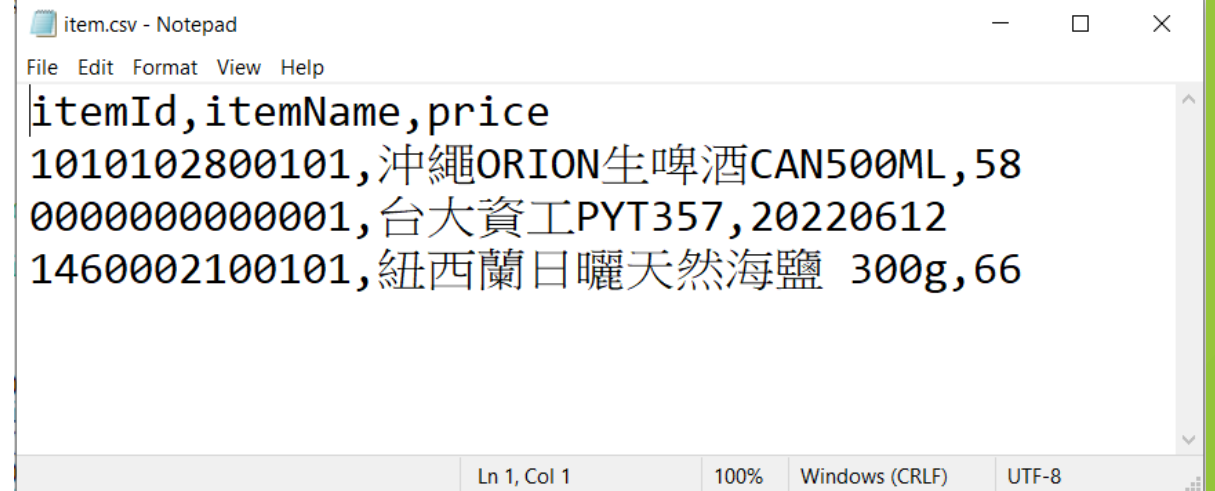
程式架構說明

File Edit Format Run Options Window Help

```
import os, sys, json, requests

#將更新價格由csv讀出
itemid = []
itemname = []
newprice = []

exist = os.path.exists("item.csv")
if not exist:
    print("Please put csv file in the folder")
    sys.exit(0)
else:
    file = open("item.csv", encoding="utf-8")
    filel = file.readlines()
    for i in range(1, len(filel)):
        fl = filel[i].split(",")
        itemid.append(fl[0])
        itemname.append(fl[1])
        newprice.append(fl[2].replace("\n", ""))
    else:
        new = dict(zip(itemid, newprice))
        name = dict(zip(itemid, itemname))
        file.close()
```



item.csv - Notepad

File Edit Format View Help

itemId,itemName,price
1010102800101,沖繩ORION生啤酒CAN500ML,58
00000000000001,台大資工PYT357,20220612
1460002100101,紐西蘭日曬天然海鹽 300g,66

Ln 1, Col 1 100% Windows (CRLF) UTF-8

- Item.csv為使用者準備上傳至後台的檔案
- 測試檔案中有三個品項，各包含三項資訊
- 這段程式會將讀取的資料寫為兩組以item id為key的dictionary:
 - new: {"itemid": "new price"}
 - name: {"itemid": "item name"}

程式架構說明

```
#由系統中抓出原價並比對是否有更新
headers = {'content-type': 'application/json'}
getURL = "http://localhost:3333/api/public/core/v1/items/id?projection=S"

updated_id = []
for i in itemid:
    URL = getURL.replace("id", i)
    response_g = requests.get(URL, headers=headers, auth=('config','config'))
    r = response_g.json()
    itemprice = [r["itemId"], r["price"]]
    if eval(itemprice[1])!= eval(new[itemprice[0]]):
        updated_id.append(itemprice[0])

print("Item(s) with updated price:\n")
for m in updated_id:
    print("%s, new price: %s 元" %(name[m], new[m]))
print()
```

- 將指定item id的既有資料由後台抓出並比對新值
- 後台取出之資料為JSON格式
- 將有更新數據之item id寫入list: updated_id
- 印出updated_id內容由使用者再次確認

GET

/api/public/core/v1/items Get multiple items

```
{
  "itemId": "00000000000001",
  "itemName": "台大資工PYT357",
  "price": "20220612"
}
```

```
{
  "itemId": "1010102800101",
  "itemName": "沖繩ORION生啤酒CAN500ML",
  "price": "58"
}
```

```
{
  "itemId": "1460002100101",
  "itemName": "紐西蘭日曬天然海鹽 300g",
  "price": "66"
}
```

資料來源: <https://www.pricer.com/>

程式架構說明

#讓有更新的品項閃燈

```
flashURL = "http://localhost:3333/api/public/core/v1/flash/items"
f1 = open("item_flash.txt",encoding="utf-8")
item_f = f1.read()
f1.close()
```

```
check = str(input("Press 'Y' to confirm the new price(s); linked label(s) will flash: "))
```

```
if check != "Y":
```

```
    print("Cancel. The price is not updated")
```

```
    sys.exit(0)
```

```
else:
```

```
    for j in updated_id:
```

```
        flash = item_f.replace("string", j).replace("\n", "")
```

```
        response_f = requests.post(flashURL, data=flash.encode("utf-8"), headers=headers, auth=('config','config'))
```

```
        print("item %s is flashing"%j if response_f == 200 or 202 else "%s is not processed"%j)
```

#讓有更新的品項變價

```
patchURL = "http://localhost:3333/api/public/core/v1/items"
```

```
f = open("item_update.txt",encoding="utf-8")
```

```
item_u = f.read()
```

```
f.close()
```

```
for k in updated_id:
```

```
    price = item_u.replace("new_id", k).replace("new_price", new[k]).replace("\n", "")
```

```
    response_p = requests.patch(patchURL, data=price.encode("utf-8"), headers=headers, auth=('config','config'))
```

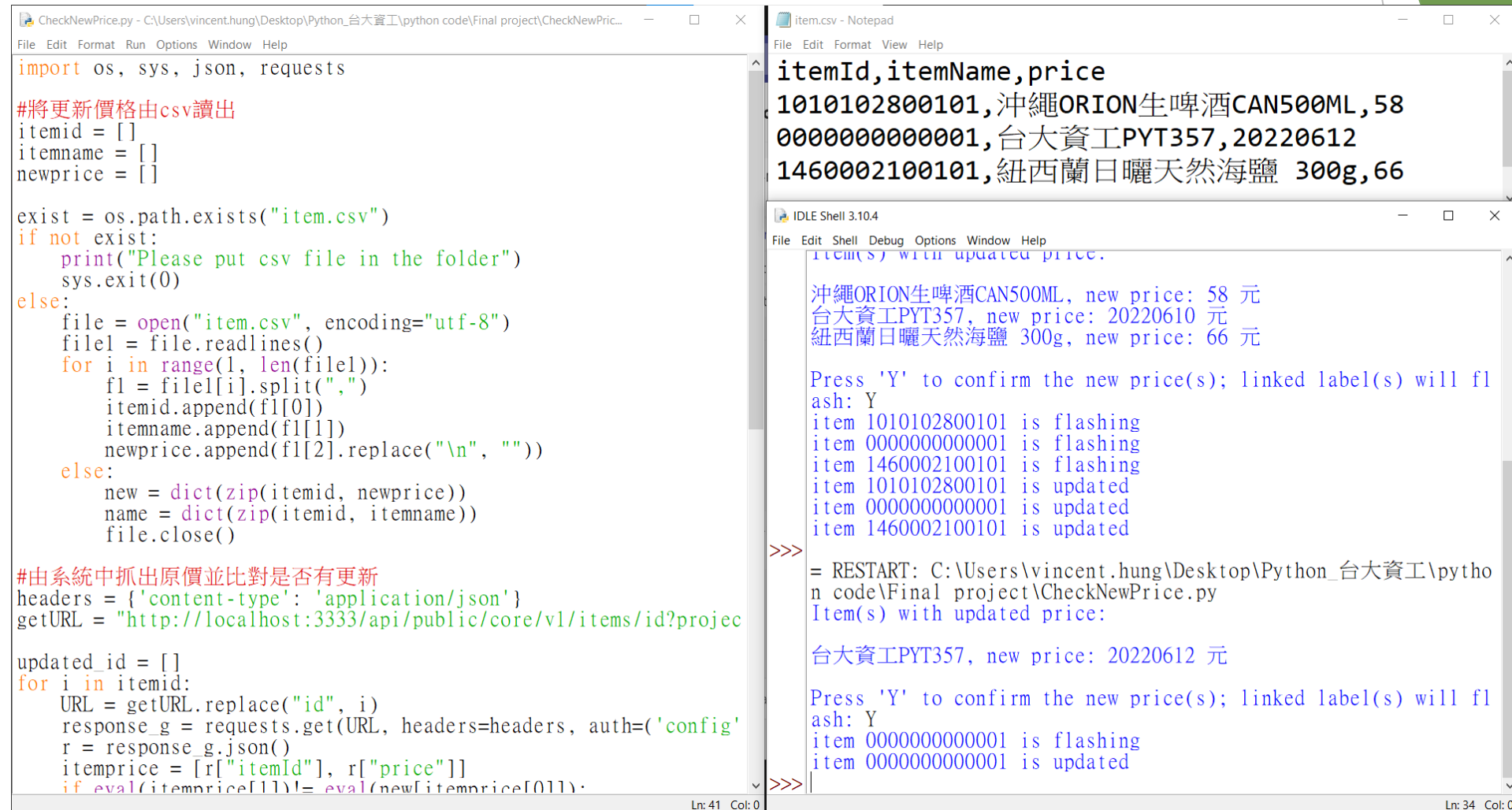
```
    print("item %s is updated"%k if response_p == 200 or 202 else "%s is not processed"%k)
```

```
item_flash.txt - Notepad
File Edit Format View Help
{
  "configuration": {
    "color": "#ff00ff",
    "duration": 10,
    "flashType": 20,
    "realTime": true
  },
  "itemIds": [
    "string"
  ]
}
```

```
item_update.txt - Notepad
File Edit Format View Help
{
  "itemId": "new_id",
  "price": "new_price"
}
```

- 使用者確認更新後將有updated_id之內容寫入對應之API格式(item_flash.txt and item_update.txt)
- 啟動閃燈模式並變更電子標籤之顯示
- Response ID 200及202表示成功呼叫API
- 若成功印出對應動作，反之則提示執行失敗

Live Demo



The screenshot displays a live demo of a Python application. On the left, a code editor window titled 'CheckNewPrice.py' shows the source code. The code imports 'os', 'sys', 'json', and 'requests'. It reads a CSV file 'item.csv' and updates prices from a web API. Comments in Chinese explain the steps: reading from CSV, checking for updates, and fetching from the API. The code uses 'zip' to pair item IDs with new prices and 'eval' for comparison. The status bar at the bottom indicates 'Ln: 41 Col: 0'.

```
import os, sys, json, requests

#將更新價格由csv讀出
itemid = []
itemname = []
newprice = []

exist = os.path.exists("item.csv")
if not exist:
    print("Please put csv file in the folder")
    sys.exit(0)
else:
    file = open("item.csv", encoding="utf-8")
    filel = file.readlines()
    for i in range(1, len(filel)):
        fl = filel[i].split(",")
        itemid.append(fl[0])
        itemname.append(fl[1])
        newprice.append(fl[2].replace("\n", ""))
    else:
        new = dict(zip(itemid, newprice))
        name = dict(zip(itemid, itemname))
        file.close()

#由系統中抓出原價並比對是否有更新
headers = {'content-type': 'application/json'}
getURL = "http://localhost:3333/api/public/core/v1/items/id?projec

updated_id = []
for i in itemid:
    URL = getURL.replace("id", i)
    response_g = requests.get(URL, headers=headers, auth=('config'
    r = response_g.json()
    itemprice = [r["itemId"], r["price"]]
    if eval(itemprice[1])!= eval(new[itemprice[0]]):
```

On the right, a Notepad window titled 'item.csv' shows the CSV data:

itemId	itemName	price
1010102800101	沖繩ORION生啤酒CAN500ML	58
0000000000001	台大資工PYT357	20220612
1460002100101	紐西蘭日曬天然海鹽 300g	66

Below the CSV, an IDLE Shell window titled 'IDLE Shell 3.10.4' shows the program's output. It displays the items with updated prices, prompts for confirmation, and shows the results of the API calls. The status bar at the bottom indicates 'Ln: 34 Col: 0'.

```
Item(s) with updated price:
沖繩ORION生啤酒CAN500ML, new price: 58 元
台大資工PYT357, new price: 20220610 元
紐西蘭日曬天然海鹽 300g, new price: 66 元

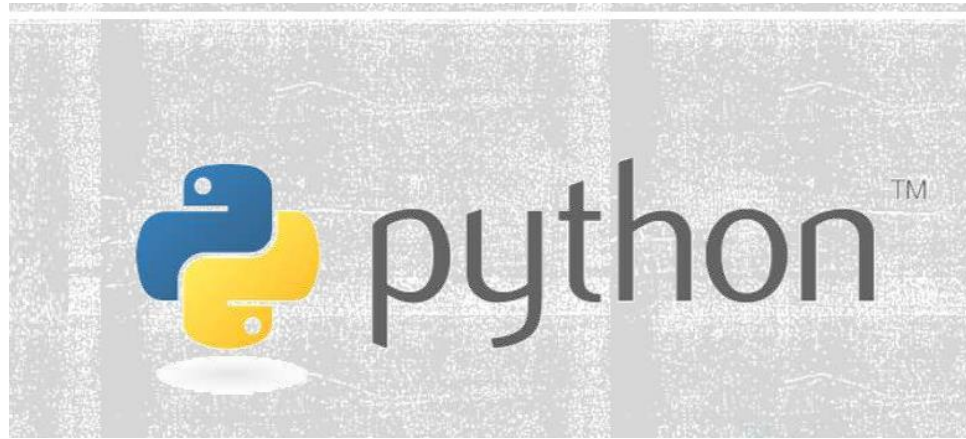
Press 'Y' to confirm the new price(s); linked label(s) will fl
ash: Y
item 1010102800101 is flashing
item 0000000000001 is flashing
item 1460002100101 is flashing
item 1010102800101 is updated
item 0000000000001 is updated
item 1460002100101 is updated

>>>
= RESTART: C:\Users\vincent.hung\Desktop\Python_台大資工\pytho
n code\Final project\CheckNewPrice.py
Item(s) with updated price:

台大資工PYT357, new price: 20220612 元

Press 'Y' to confirm the new price(s); linked label(s) will fl
ash: Y
item 0000000000001 is flashing
item 0000000000001 is updated

>>>
```

Thank you for your attention