加入 寫入資料庫功能

目前寫入資料庫的功能只建立在"POST”、”DELETE”功能底下

使用"POST"成功新增進新資料、”DELETE”刪除資料時，才會執行寫入資料庫

後續看需求更改內容

```
# 載入必須套件
from flask import Flask, request
from flask_restful import Resource, Api
import json
import pymysql


# 創建Flask app物件
app = Flask(__name__)
api = Api(app)

# 只放要寫入資料庫的資料陣列，寫入之前都會重新抓"當下" "items"內元素，等確實寫入之後，會"清空"這陣列
(del list_data[:])，後續要寫入資料庫時，再重新抓當下"items"內元素
# 可以確保都是操作最新的"items"狀態，便不會出現沒寫入某筆資料的狀況
list_data = []


# 連線參數設定
dbhost = "localhost"
dbuser = "root"
dbpassword = "allen33323"
dbname = "new_1"

# 資料庫連線設定
db = pymysql.connect(host=dbhost, user=dbuser, password=dbpassword, database=dbname, c
harset="utf8")

# 建立操作游標
cursor = db.cursor()

# 建立寫入資料庫、刪除資料語法
sql_insert = "INSERT INTO postman_test( name , gender , age ) VALUES (%s,%s,%s)"
sql_delete = "DELETE FROM postman_test WHERE name= (%s)"



# 創建陣列，用於存放資料
items = [
    {
```

```
        "name": "laby",
        "gender": "female",
        "age": 22
    },

    {
        "name": "evan",
        "gender": "male",
        "age": 26
    }
]



# 建立一個class "ItemsList" ， 底下功能都歸屬在 "ItemsList"之下
class ItemsList(Resource):


# 查找資料，使用 "GET"功能
    # 邏輯：
    # 查詢全部資料：
    # 先取得網址欄位內的網址內容，並且replace掉前半段"http://127.0.0.1:5000/items"
    # 取代之後判斷是否為 "空字串"，如果是，代表為"查詢全部資料"，return 全部資料(items)
    # ================================================================================
    # 查詢單一資料
    # 取代之後，如果不是 "空字串"，表示後面有其他內容，代表為"查詢某一筆資料"(預設是查詢name)
    # replace掉查詢名字部分的字串("?name=")，這樣就只會剩下查詢的 "名字"字串(例如：evan)
    # 遍歷查找(next)，查找條件items['name']== url(url 是"名字"字串)成立的資料，令其為 "item"
    # 如果存在，回傳 item
    # 如果不存在，則回傳不存在的訊息

    def get(self):

        # 取得網址輸入欄位內的url
        url = request.url

        # 用空字串取代掉網址的前半段(白話文：刪掉網址前半段)
        url = url.replace("http://127.0.0.1:5000/items", "")

        # 查詢全部資料，條件為空字串
        if url == "":
            return {'items': items}  # 回傳全部資料(items)


        # 查詢單一資料，條件為 "非"空字串
        else:
            url = url.replace("?name=", "")      # replace掉查詢名字部分的字串("?name=")
            item = next(filter(lambda x: x['name'] == url, items), None)    # 查找符合條
件的資料
            if item:
                return {'item': item}, 200 if item else 404      # 回傳符合條件的資料
            else:
                return {'message': f'An item with name \'{item}\' is not exist'}, 201

# ================================================================================
=========

# 新增資料或是查詢某筆資料，使用 "POST"功能
```

```
    # 邏輯：
    # 先取得 "Body"→"raw"中的內容(request.get_data())，再將其轉換成JSON資料格式(json.loads)
    # 轉換完成後，令其為 "data"，此時data的資料型態為"dictionary"
    # 建立一個空陣列list1，用來裝全部資料(items)中，"鍵是name的資料"
    # 判斷 data['name']是否存在於list1，data['name']是user輸入的"name"資料
    # ================================================================================
==
    # post查找資料：
    # 如果存在，遍歷查找(next)，查找條件items['name']== data['name']成立的資料
    # 取出該筆資料，並令其為"item"，該資料包含"name"、"gender"、"age"
    # return item
    # ================================================================================
==
    # post新增資料
    # 如果不存在，判斷輸入的資料是否完整(包含"name"、"gender"、"age")，一開始已經預設會輸入"nam
e"了，所以需要判斷"gender"、"age"
    # 如果 "gender"、"age"不存在data.keys中，表示資料輸入不完整，會回傳一道訊息(errormessage)，
表明該筆輸入("name")不存在，且如果想要新增資料，則需要輸入完整的資料
    # 如果 "gender"、"age"存在data.keys中(else)，表示資料輸入完整，request.get_data()取得輸入
的內容並轉換成JSON資料型態，令其為"data"
    # 把 "data"內資料分出鍵、值，並令其為"item"(注意，不是items)
    # 把"item"加入進全部資料"items"中
    # 回傳 item 成功建立的訊息
    # ================================================================================
==
    # 提出"當下"items內元素，把那些元素的"值"放進"list_data"
    # "list_data"內每一個元素的第"0"個元素(i[0])是"name"的資料，跟"com"組合，做出"select ~ whe
re"指令，用於 防止後續 "重複寫入"
    # 執行寫入 "cursor.execute(sql_insert, i)"
    # 重點：最後 "清空" "list_data" (del list_data[:])，理由是目前只有"POST"、"DELETE"會接續寫
入資料庫功能，在寫入前會先將"當前" "items"內元素放進"list_data"
    # 如此可確保當前操作的"list_data"資料都存在於當下的"items"內
    # ================================================================================
==
    # "POST"新資料步驟：
    # 取出當前items裡面的"name"值並放入"list1" → 判斷輸入的資料不存在於"list1"內 → 輸入完整資料
 →  新增進"items" → 把當下"items"內元素依序放入"list_data"(這時list_data是空陣列)
    # → 組合出寫入語法  → 寫入 → 清空"list_data"

    def post(self):

        # 用來裝items中，鍵是name的資料
        list1 = []

        # 取得user輸入的內容，並轉換成JSON格式
        data = request.get_data()
        data = json.loads(data)


        for partdata in items:
            list1.append(partdata['name'])      # 把items中，鍵是name的資料裝進list1
        print("list1 = " , list1)

        # 如果輸入的name存在於list1
        if data['name'] in list1:
            data_name = data['name']
            item = next(filter(lambda x: x['name'] == data_name, items), None)       #
查找包含name的該筆資料
```

```
            print("exist")
            return item     # 回傳該筆資料


        # 如果輸入的name "不" 存在於list1
        else:
            # 如果"gender"、"age"不存在於data的"鍵"之中(data.keys())，表示name不存在之外，資料
輸入也不完整
            if "gender" not in data.keys() or "age" not in data.keys():
                print("not exist")

                # 告訴user該name不存在之外，如果想新增出該name的資料，請輸入完整資料(包含"nam
e","genfer","age")
                errormessage = "message : An item is not exists,if you want to add dat
a,please enter the full data message(name,genfer,age) "
                return errormessage     # 回傳 errormessage 訊息


            # 如果"gender"、"age"存在於data的"鍵"之中(data.keys())
            else:
                # 取得user輸入的內容，並轉換成JSON格式
                data = request.get_data()
                data = json.loads(data)

                # 把 "data"內資料分出鍵、值，並令其為"item"
                item = {'name': data['name'], 'gender': data['gender'], "age": data["a
ge"]}

                # item新增進items
                items.append(item)
                print("sucess")
                print("items = " , items)

                # 提出當下全部資料(items)內元素
                for dd in items:
                    list_data.append([dd['name'], dd["gender"], dd["age"]])     # 把所
有元素都放進list_data
                print("list_data = ",list_data)    # 確認用

                # 提出list_data內元素
                for i in list_data:
                    i0 = i[0]       # i[0]是"name"資料
                    com = 'SELECT * FROM postman_test WHERE name = %s '
                    strt = (i0)

                    cursor.execute(com, strt)

                    # 重點注意：cursor.fetchall()指令，顯示出"資料庫底下"Result Grid"全部資料
                    # 如果沒有資料，則該指令"0"，若是有資料，該指令長度則為"1"
                    search_all = cursor.fetchall()

                    # 防止重複寫入條件
                    if len(search_all) == 0:
                        print("add")
                        print("name = ", i0)

                        try:
                            cursor.execute(sql_insert, i)
```

```
                            # 提交修改
                            db.commit()
                            print('success')
                            print("")

                        except pymysql.Error as e:
                            # 發生錯誤時停止執行SQL
                            db.rollback()
                            print('error = ', str(e))


                del list_data[:]    # 清空"list_data"
                print("list_data",list_data)
                return {'message': f'An item with name \'{item}\' has been added succe
ssfully'}, 201        # 回傳 item已成功新增 訊息


# ================================================================================
=========
# 刪除某筆品項

    # 邏輯：
    # 先取得 "Body"→"raw"中的內容(request.get_data())，再將其轉換成JSON資料格式(json.loads)
    # 轉換完成後，令其為 "data"，此時data的資料型態為"dictionary"
    # 建立一個空陣列list2，用來裝全部資料(items)中，鍵是name的資料
    # 判斷 data['name']是否存在於list2，data['name']是user輸入的"姓名"資料
    # 如果data['name']存在於list2，表示list2裡面存在那筆資料，才 "能" 被刪除
    # 遍歷查找(next)，查找條件items['name']== data['name']成立的資料
    # 將查找到的資料，從"items"中移除
    # 回傳 移除成功的訊息
    # ============================================================================
    # "DELETE"步驟：
    # 取出當前items裡面的"name"值並放入"list1" → 判斷輸入的資料存在於"list1"內 → 查找符合輸入nam
e的該筆資料 → 從"items"中移除該筆資料
    # → 把當下"items"內元素依序放入"list_data"(這時list_data是空陣列) → 組合出刪除語法  → 執行
後，資料庫刪除該資料 → 清空"list_data"

    def delete(self):

        # 用來裝items中，鍵是name的資料
        list1 = []

        # 取得user輸入的內容，並轉換成JSON格式
        data = request.get_data()
        data = json.loads(data)


        for partdata1 in items:
            list1.append(partdata1['name'])  # 把items中，鍵是name的資料裝進list1


        # 如果輸入的name存在於list1中
        if data['name'] in list1:
            data_name = data['name']
            item = next(filter(lambda x: x['name'] == data_name, items), None)      #
遍歷查找(next)，查找條件items['name']== data['name']成立的資料
            items.remove(item)      # 移除該筆資料
            print("sucess")
```

```
            # 抓取當下"items"內元素
            for dd in items:
                list_data.append([dd['name'], dd["gender"], dd["age"]])
            print("list_data = ", list_data)  # 確認用


            strt = data['name']

            try:
                cursor.execute(sql_delete, strt)
                # 提交修改
                db.commit()
                print('success')
                print("")

            except pymysql.Error as e:
                # 發生錯誤時停止執行SQL
                db.rollback()
                print('error = ', str(e))

            del list_data[:]    # 清空"list_data"

            print("list_data = ", list_data)
            return {'message': f'An item with name \'{item}\' has been deleted success
fully'}, 201      # 回傳 移除成功 訊息

        else:
            return {'message': 'An item is not exist'}, 201     # 回傳 該筆資料不存在 訊息



# 建立使用方法
api.add_resource(ItemsList, '/items')

if __name__ == "__main__":
    app.run(port=5000, debug=True)


# 關閉連線
cursor.close()
db.close()
```