

圖書系統

各網站平等帳密

▶ 平等幼稚園信箱：

帳號：equality.nknu@gmail.com

密碼：nknu@709

▶ Heroku：

帳號：equality.nknu@gmail.com

密碼：equality@709

▶ gmail/facebook：

帳號：chatbot520@gmail.com

密碼：chatbot@520

▶ line：

帳號：chatbot520@gmail.com

密碼：chat520bot

使用的工具

- ▶ 語言：python
 - Flask
 - Postgresql
- ▶ server : Heroku
 - <https://www.heroku.com/>
- ▶ 寫code軟體 : sublime
 - <https://www.sublimetext.com>

第一步

建立資料庫

► all_connect_db.py

```
all_connect_db.py
1  from flask import Flask
2  from flask_sqlalchemy import SQLAlchemy
3  from flask_script import Manager
4  from flask_migrate import Migrate, MigrateCommand
5  import csv
6  import psycopg2
7  from werkzeug.security import generate_password_hash, check_password_hash
8
9  app = Flask(__name__)
10
11 #下面的連結要填 database 的 url
12 app.config['SQLALCHEMY_DATABASE_URI'] = 'postgres://aenplcguyghqsr:d4150af71b02722634f0729b96db7cd591fb92381527917b5c8757abe990cbba@ec2-107-21-98-165.com'
13
14 db = SQLAlchemy(app)
15 migrate = Migrate(app, db)
16
17 manager = Manager(app)
18 manager.add_command('db', MigrateCommand)
19
20
21 with open('booklist.csv', 'newline='',encoding='utf8') as f:
22     csv_reader = csv.DictReader(f)
23     csv_reader = csv.reader(f)
24     book_id = []
25     book_name = []
26     book_publisher = []
27     for row in csv_reader:
28         book_id.append(row[0])
29         book_name.append(row[1])
30         book_publisher.append(row[2])
31
32 conn = psycopg2.connect(database="d5pdiejktvknug", user="aenplcguyghqsr", password="d4150af71b02722634f0729b96db7cd591fb92381527917b5c8757abe990cbba", host="ec2-107-21-98-165.com")
33 cur = conn.cursor()
34 cur.execute("DROP TABLE bookList")
35 conn.commit()
36
37 cur.execute('''CREATE TABLE bookList(id SERIAL PRIMARY KEY,book_id TEXT,book_name TEXT,if_borrow TEXT,people_id TEXT,borrow_status TEXT,borrow_class TEXT''')
38
39 for i in range(0, len(book_id)):
40     book_id_i = book_id[i]
41     book_name_i = book_name[i]
42     book_publisher_i = book_publisher[i]
43     cur.execute("INSERT INTO booklist(book_id,book_name,publisher) VALUES ('{bookid}', '{bookname}', '{publisher}')".format(bookid=book_id_i,bookname=book_name_i,publisher=book_publisher_i))
44     conn.commit()
45
46 cur.execute("UPDATE booklist SET if_borrow='未借出' WHERE if_borrow IS NULL")
47 conn.commit()
48
49
50
```

Line 1, Column 1 master [645] Tab Size: 4 Python

第一步

建立資料庫

- ▶ all_connect_db.py

```
all_connect_db.py
1 from flask import Flask
2 from flask_sqlalchemy import SQLAlchemy
3 from flask_script import Manager
4 from flask_migrate import Migrate, MigrateCommand
5 import csv
6 import psycopg2
7 from werkzeug.security import generate_password_hash, check_password_hash
8
9 app = Flask(__name__)
10
11 #下面的連結要填 database 的 url
12 app.config['SQLALCHEMY_DATABASE_URI'] = 'postgres://aenplcguyghqsrd4150af71b02722634f0729b96db7cd591fb92381527917b5c8757abe990cbba@ec2-107-21-98-16'
13
14
15 migrate = Migrate(app, db)
16
17 manager = Manager(app)
18 manager.add_command('db', MigrateCommand)
19
20
21 with open('booklist.csv', 'newline='',encoding='utf8') as f:
22     csv_reader = csv.DictReader(f)
23     book_id = []
24     book_name = []
25     book_publisher = []
26     for row in csv_reader:
27         book_id.append(row['id'])
28
29 conn = psycopg2.connect(database="d5pdiejktvknu", user="aenplcguyghqsrd4150af71b02722634f0729b96db7cd591fb92381527917b5c8757abe990cbba", password="d4150af71b02722634f0729b96db7cd591fb92381527917b5c8757abe990cbba", host="ec2-107-21-98-16")
30 cur = conn.cursor()
31 #cur.execute("DROP TABLE booklist")
32 #conn.commit()
33
34 cur.execute("CREATE TABLE booklist(id SERIAL PRIMARY KEY, book_id TEXT, book_name TEXT, if_borrow TEXT, people_id TEXT, borrow_status TEXT, borrow_class TEXT")
35
36 for i in range(0, len(book_id)):
37     book_id_i = book_id[i]
38     book_name_i = book_name[i]
39     book_publisher_i = book_publisher[i]
40     cur.execute("INSERT INTO booklist(book_id, book_name, publisher) VALUES ('{}', '{}', '{}')".format(bookid=book_id_i, bookname=book_name_i, publisher=book_publisher_i))
41     conn.commit()
42
43 cur.execute("UPDATE booklist SET if_borrow='未借出' WHERE if_borrow IS NULL")
44 conn.commit()
45
46
47
48
49
50
```

Line 1, Column 1 master [645] Tab Size: 4 Python

第一步

建立資料庫

- ▶ all_connect_db.py

- ▶ Sql 的 url 和 database 的資訊在 heroku裡面

The screenshot shows the Heroku dashboard for an application named "你的APP". The top navigation bar includes links for Overview, Resources, Deploy, Metrics, Activity, Access, and Settings. The "Overview" tab is selected. In the main content area, under the heading "Installed add-ons", there is a card for "Heroku Postgres Hobby Dev". A blue button labeled "按這裡" (Click here) with an orange arrow pointing to it is overlaid on the card. The text "\$0.00/month" is displayed next to the add-on name. To the right of the card is a link "Configure Add-ons". Below this section, another card shows "Dyno formation \$0.00/month" with the status "ON" and the text "This app is using free dynos". At the bottom of the card, it lists the processes: "web unicorn all_psy:app".

第一步

建立資料庫

- ▶ all_connect_db.php

The screenshot shows the Heroku Datastore settings page for a PostgreSQL database named 'heroku-postgresql'. The page includes navigation tabs for Overview, Durability, Settings (which is selected), and Dataclips. The 'ADMINISTRATION' section contains two main buttons: 'View Credentials...' (with an orange arrow pointing to it) and 'Reset Database...' (with a red border). Below these are sections for 'Reset Database' and 'Destroy Database'.

DATA

Datastores > 你的POSTGRESQL

SERVICE heroku-postgresql PLAN hobby-dev BILLING APP stella-library-test

Overview Durability Settings Dataclips

ADMINISTRATION

Database Credentials

Get credentials for manual connections to this database.

View Credentials... 按這裡

Reset Database

Reset the database to its originally-provisioned state, deleting all data inside it.

Reset Database... Reset Database...

Destroy Database

Destroys the database and all of the data inside it.

Destroy Database... Destroy Database...

第一步

建立資料庫

- ▶ all_connect_db.

The screenshot shows the Heroku Datastore settings page for a PostgreSQL database named 'postgresql-metric-85850'. The page includes navigation links for Overview, Durability, Settings (which is selected), and Dataclips. It displays administrative credentials for manual connections, noting that they are not permanent and are rotated periodically. A red box highlights the credential information, which is also summarized in a red callout box labeled '資訊在這裡' (Information is here). The detailed credential information is as follows:

Host	ec2-107-21-98-165.compute-1.amazonaws.com
Database	d5pdiejktvknug
User	aenplcguyghqsr
Port	5432
Password	d4150af71b02722634f0729b96db7cd591fb92381527917b5c8757abe990cbba
URI	postgres://aenplcguyghqsr:d4150af71b02722634f0729b96db7cd591fb92381527917b5c8757abe990cbba@ec2-107-21-98-165.compute-1.amazonaws.com:5432/postgresql-metric-85850
Heroku CLI	heroku pg:psql postgresql-metric-85850 --app stella-library-test

Reset Database

Reset the database to its originally-provisioned state, deleting all data inside it.

第一步——建立資料庫

各個TABLE

- ▶ booklist
 - id, book_id, book_name, if_borrow(借出、未借出), people_id, borrow_status(老師、學生等) , borrow_class, people_name, borrow_time, publisher
- ▶ loginlist (登入資料)
 - id, account, password, email, status
- ▶ history (書的借出還書時間)
 - id, borrow_time, book_id, book_name, publisher, if_borrow, people_id, people_status, people_class, people_name
- ▶ peoplelist
 - id, people_id, people_status, people_class, people_name
- ▶ teachaid_list
 - id, teachaid_id, teachaid_name, teachaid_number, if_borrow, people_id, borrow_status, borrow_class, people_name, borrow_time, teachaid_image
- ▶ teachaid_his (教具歷史)
 - id, borrow_time, teachaid_id, teachaid_name, teachaid_number, if_borrow, people_id, people_status, people_class, people_name

第一步

建立資料庫

- ▶ all_connect_db.py
- ▶ 準備好之後在終端機
(cmd) 裡run，會等
很久很久

```
$ python all_connect_db.py
```

第二步

各個頁面及功能

對應的頁面檔案名稱

▶ all_psy.py

```
102     return render_template('user_home.html', account=account, password=password)
```

▶ 頁面設計：html

- 全部頁面皆在templates裡

```
166     <button onclick="location.href='{{ url_for('borrow_teachaid_peo') }}'>
```

html 裡的url_for 後面接的是all_psy.py檔案裡function的名稱

```
3217  
3218     @app.route('/teachaid/borrow/peo/input/<account>/<password>', methods=['GET', 'POST'])  
3219     def borrow_teachaid_peo(account, password):
```

各個頁面及功能

圖書館系統

all_psy.py

書目搜尋 →

```
281  
282 @app.route('/search', methods=['GET', 'POST'])  
283 def search():
```

登入 →

```
68 @app.route('/login', methods=['GET', 'POST'])  
69 def login():
```

第二步-各個頁面及功能

書目搜尋

all_psy.py

The diagram illustrates the flow of data from the search interface to the search results table. A large blue arrow points downwards from the search interface at the top to the search results table at the bottom.

書目搜尋

```
281
282     @app.route('/search', methods=['GET', 'POST'])
283     def search():
```

關鍵字 搜尋

首頁

搜尋結果

```
284
285     @app.route('/search_solution', methods=['POST'])
286     def search_solution():
```

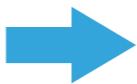
首頁	書目編	書名	出版社	是否 已借
	B00126	找一找我的烏龜在哪裡?	風車圖書	未借 出
	B00169	我們是好朋友	漢聲	未借 出
	R00184	媽媽我愛你	美奇旺旺	未借

第二步-各個頁面及功能

登入

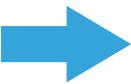
all_psy.py

```
68     @app.route('/login', methods=['GET', 'POST'])  
69     def login():
```



```
113    @app.route('/login/solution', methods=['POST'])  
114    def login_solution():  
115
```

```
73     @app.route('/login/success/<account>/<password>')  
74     def login_sucessed(account, password):
```



管理者

書籍

[借書清單](#)

[借書](#)

[還書](#)

[歷史紀錄](#)

管理

[成員搜尋](#)

[成員管理](#)

[書目管理](#)

[帳號管理](#)

[教具管理](#)

教具

[教具搜尋](#)

[教具借閱](#)

[教具歸還](#)

[教具借出清單](#)

[歷史紀錄](#)

[登出](#)

第二步-各個頁面及功能

登入

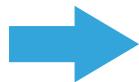
all_psy.py

```
68 @app.route('/login', methods=['GET', 'POST'])  
69 def login():
```



```
113 @app.route('/login/solution', methods=['POST'])  
114 def login_solution():  
115
```

```
73 @app.route('/login/success/<account>/<password>')  
74 def login_sucessed(account, password):
```



login_solution 裡有一個generate_password_hash
是為了將密碼在網頁之間傳輸時是以亂數的樣子呈現

```
129 manager_password = generate_password_hash(manager_detail[1])  
130
```

每一頁都需確認帳號密碼是否是對的
密碼的亂數確認是用check_password_hash
若帳號密碼不對就會跳回首頁

管理者

書籍 借書清單 借書 還書 歷史紀錄	管理 成員搜尋 成員管理 書目管理 帳號管理 教具管理	登出
---	---	--------------------

```
327 cur.execute("SELECT account, password FROM loginlist WHERE id='1'")  
328 manager_detail = cur.fetchone()  
329  
330 cur.execute("SELECT account, password FROM loginlist WHERE id='2'")  
331 user_detail = cur.fetchone()  
332  
333 manager_account = manager_detail[0]  
334 manager_password = manager_detail[1]  
335  
336 user_account = user_detail[0]  
337 user_password = user_detail[1]  
338  
339 if (account==manager_account):  
340     check_manager = check_password_hash(password, manager_password)  
341     if (check_manager==1):  
342         form = Borrow_Peo_Form()  
343         back_home = '管理者'  
344         return render_template('borrow_peo_id_input.html', form=form, ac  
345     else:  
346         back_to_library_home = url_for('library')  
347         return redirect(back_to_library_home)
```

第二步-各個頁面及功能

all_psy.py



第二步-各個頁面及功能

all_psy.py

借出搜尋

借書清單

```
1058 @app.route('/unreturn/<account>/<password>', methods=['GET', 'POST'])
1059 def unreturn(account,password):
```

搜尋輸入

• 可輸入書目編碼、書名、成員編碼、成員姓名、成員班級、成員身份等關鍵字。

搜尋方式

關鍵字

Go

全部借書清單

回到管理者首頁

登出

借出搜尋結果

```
1091 @app.route('/unreturn_solution/<account>/<password>', methods=['GET', 'POST'])
1092 def unreturn_solution(account,password):
```

回到管理者首頁								
書目編碼	書名	出版社	是否已借出	成員編碼	成員身份	成員班級	成員姓名	借出時間
B00001	遲來的禮物	啟思圖書	已借出	00025	學生	海馬村	劉星佑	2019/03/03, 13時9分15秒
B00002	水的遊戲	天下雜誌	已借出	00025	學生	海馬村	劉星佑	2019/03/09, 11時45分38秒

回到管理者首頁

首頁

or

全部借出的書

借出搜尋結果

```
1199 @app.route('/unreturn_all/<account>/<password>', methods=['GET', 'POST'])
1200 def unreturn_all(account,password):
```

回到管理者首頁								
書目編碼	書名	出版社	是否已借出	成員編碼	成員身份	成員班級	成員姓名	借出時間
B00001	遲來的禮物	啟思圖書	已借出	00025	學生	海馬村	劉星佑	2019/03/03, 13時9分15秒
B00002	水的遊戲	天下雜誌	已借出	00025	學生	海馬村	劉星佑	2019/03/09, 11時45分38秒

回到管理者首頁

首頁

第二步-各個頁面及功能

all_psy.py



借書

```
322     @app.route('/borrow/people_id/input/<account>/<password>')
323     def borrow_peo(account,password):
324         return render_template('borrow_peo.html')
```

借書

成員編碼 Go

[回到管理者首頁](#)

[登出](#)

→ 365 @app.route('/borrow/book_id/input/<account>/<password>/<peo_>')
366 def borrow_book(account,password,peo_word,the_book):

借書

成員編碼 : 00025 身份 : 學生 班級 : 海馬村 姓名 :

書目編碼 Go

借出的書 :

書目編碼	書名	借書時間
B00001	遲來的禮物	2019/03/03, 13時9分15秒
B00002	水的遊戲	2019/03/09, 11時45分38秒

[下一位](#) [回到管理者首頁](#)

[登出](#)

→ 529 @app.route('/borrow_solution/<account>/<password>', methods=
530 def borrow_solution(account,password):

B00044--親親自然52 : 借書成功 !

借書

成員編碼 : 00097 身份 : 家長 班級 : 家長03 姓名 : 家長0

書目編碼 Go

借出的書 :

書目編碼	書名	借書時間
B00044	親親自然52	2019/06/15, 13時31分18秒

[下一位](#) [回到管理者首頁](#)

[登出](#)

借書

```
529 @app.route('/borrow_solution/<account>/<password>', methods=
530 def borrow_solution(account,password):
591     elif bor_peo_sta=='學生' or bor_peo_sta=='家長':
592         if (bor_book_num>=2):
593             error='{}已達借書上限'.format(peo_name=bor_peo_name)
```

學生及家長的借書上限
都只有兩本



第二步-各個頁面及功能

all_psy.py



還書

```
810 @app.route('/return/book_id/input/<account>/<password>',  
811 ▼ def return_book(account,password):
```

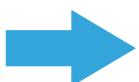
還書

書目編碼

Go

[回到管理者首頁](#)

[登出](#)



```
857 @app.route('/return_solution/<account>/<password>/<submit>',  
858 def return_solution(account,password,submit_empty,the_bo
```

B00044--親親自然52：還書成功！

還書

成員編碼 : 00097 身份 : 家長 班級 : 家長03 姓名 : 家長0

書目編碼

Go

[回到管理者首頁](#)

[登出](#)

第二步-各個頁面及功能

all_psy.py



[歷史紀錄](#)

```
2396     @app.route('/history_input/<account>/<password>', methods=['GET'])
2397     def history_input(account, password):
2398         return render_template('history_input.html', account=account, password=password)
```

歷史紀錄

成員編碼 [查詢](#)

[回到管理者首頁](#)[登出](#)

```
2438     @app.route('/history_soultion/<account>/<password>', methods=['GET'])
2439     def history_solution(account, password):
2440         return render_template('history_soultion.html', account=account, password=password)
```

歷史紀錄

成員編碼 : 00097 身份 : 家長 班級 : 家長03 姓名 : 家長0

借出時間	書目編碼	書名	出版社
2019/06/15, 13時31分18秒	B00044	親親自然52	親親自然

[回到管理者首頁](#)[登出](#)

第二步-各個頁面及功能

all_psy.py



成員搜尋

```
2224 @app.route('/people_search/<account>/<password>', methods=['GET'])
2225 def people_search(account, password):
```

搜尋成員

成員關鍵字

• 可輸入想查詢的成員姓名、編碼、班級、身份等關鍵字

Go

全部成員

回到管理者首頁

退出

全部成員

成員

回到成員管理 回到管理者首頁

成員編碼	成員班級	成員身份	成員姓名
00025	學生	海馬村	李
00026	學生	海馬村	王
00029	學生	海馬村	張

or

2266 @app.route('/people_search/all/<account>/<password>', methods=['GET'])
2267 def peo_search_all(account, password):

成員

回到成員管理 回到管理者首頁

成員編碼	成員班級	成員身份	成員姓名
00025	學生	海馬村	李
00026	學生	海馬村	王
00029	學生	海馬村	張

→

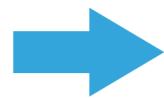
```
2317 @app.route('/people_search/solution/<account>/<password>', methods=['GET'])
2318 def peo_search_solution(account, password):
```

第二步-各個頁面及功能

all_psy.py



成員管理



管理者--成員管理

成員刪除 成員新增 更改成員

全部成員

回到管理者首頁

登出

```
1756 @app.route('/manager_people/choose/<account>/<password>')
1757 def people_choose(account,password):
```

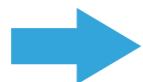
第二步-各個頁面及功能

all_psy.py

成員管理



成員刪除



要刪除的成員關鍵字

關鍵字

搜尋

```

1799 @app.route('/delete_peo/search/<account>/<password>', methods=['GET'])
1800 def delete_peo_search(account,password):

```

回到管理者首頁

登出

輸入要刪除的成員編碼

成員編碼：

Go

成員編碼	成員身份	成員班級	成員姓名
------	------	------	------

99999	老師	ss	ss
-------	----	----	----

回到管理者首頁

```

1845 @app.route('/delete_peo/input/<account>/<password>', methods=['GET'])
1846 def delete_peo_input(account,password):

```

確認要刪除？

成員編碼	成員身份	成員班級	成員姓名
99999	老師	ss	ss

確認

回到管理者首頁

```

1924 @app.route('/delete_peo/sure/<account>/<password>', methods=['GET'])
1925 def delete_peo_sure(account,password,key_word):

```

刪除成功

```

2007 @app.route('/delete_peo/solution/<account>/<password>/<solution>', methods=['GET'])
2008 def delete_peo_solution(account,password,select_peo_id):

```

成員編碼	成員身份	成員班級	成員姓名
99999	老師	ss	ss

繼續刪除

回到管理者首頁

登出



第二步-各個頁面及功能

all_psy.py

成員管理

管理者--成員管理

```
1756 @app.route('/manager_people/choose/<account>/<password>')
1757 def people_choose(account,password):
```

成員刪除	成員新增	更改成員
全部成員		
回到管理者首頁		
登出		

```
2072 @app.route('/insert_peo/input/<account>/<password>', methods=['GET'])
2073 def insert_peo_input(account,password):
```

新增成員

編碼	99999
• 編碼必須是五碼數字	
身分	<input checked="" type="radio"/> 老師 <input type="radio"/>
班級	ss
姓名	ss
<input type="button" value="新增"/>	
回到管理者首頁	
登出	

成員新增



新增成功

成員編碼	成員身份	成員班級	成員姓名
99999	老師	ss	ss
繼續新增	全部成員	回到管理者首頁	
登出			

```
2117 @app.route('/insert_peo/solution/<account>/<password>', methods=['POST'])
2118 def insert_peo_solution(account,password):
```

成員

[回到成員管理](#) [回到管理者首頁](#)

成員編碼	成員班級	成員身份	成員姓名
00025	學生	海馬村	李
00026	學生	海馬村	王
00029	學生	海馬村	張

全部成員



第二步-各個頁面及功能

all_psy.py



更改成員 →

```

2837 @app.route('/change_people/search/<account>/<password>', 
2838 def change_peo_search(account,password):
    
```

搜尋成員

成員關鍵字 Go

[回到管理者首頁](#)

[登出](#)

```

2881 @app.route('/change_people/input/<account>/<password>', 
2882 def change_peo_input(account,password):
    
```

輸入要更改的成員編碼

成員編碼 : Go

成員編碼	成員身份	成員班級	成員姓名
99999	老師	ss	kk

[回到管理者首頁](#)

[登出](#)

新增成員

編碼 : 99999
身分 老師

班級 ss

姓名 kk

[修改](#)

[回到管理者首頁](#)

```

2968 @app.route('/change_people/detail/<account>/<password>/<
2969 def change_peo_detail(account,password,peo_search):
    
```

更改成功

成員編碼	成員身份	成員班級	成員姓名
99999	老師	ee	kk

[繼續更改](#) [全部成員](#) [回到管理者首頁](#)

[登出](#)

```

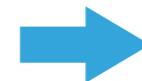
3060 @app.route('/change_people/solution/<account>/<password>')
3061 def change_peo_solution(account,password,peo_input):
    
```

第二步-各個頁面及功能

all_psy.py



[書目管理](#)



管理者--書目管理

[書目刪除](#) [書目新增](#) [全部書目](#) [回到管理者首頁](#)

[登出](#)

```
1256 @app.route('/manager_book/choose/<account>/<password>')
1257 def book_choose(account,password):
```

[全部書目](#)



全部書目

[回到書目管理](#) [回到管理者首頁](#)

書目編碼	書名	出版社
B00001	遲來的禮物	啟思圖書
B00002	水的遊戲	天下雜誌
B00003	發現好玩的數字	艾閣萌
B00004	顏色.形狀.大小相對詞	企鵝
B00005	好餓的毛毛蟲	上誼

```
1566 @app.route('/all_book/<account>/<password>', methods=['GET'])
1567 def all_book(account,password):
```

第二步-各個頁面及功能

all_psy.py

書目管理

管理者--書目管理

書目刪除 | 書目新增 | 全部書目 | 回到管理者首頁

登出

```
1256 @app.route('/manager_book/choose/<account>/<password>')
1257 def book_choose(account,password):
```

新增書目

書目新增



書名編碼

1610

1611

• 編碼開頭必須為 B，後面加上五碼數字

書目

出版社

新增

回到管理者首頁

登出

```
1610 @app.route('/insert_book/input/<account>/<password>', methods=['POST'])
1611 def insert_book_input(account,password):
```

新增成功

1654

1655

```
1654 @app.route('/insert_book/solution/<account>/<password>', methods=['POST'])
1655 def insert_book_solution(account,password):
```

書目編碼	書名	出版社	借書狀況
B99997	mad	kkk	未借出

繼續新增

回到管理者首頁

登出



第二步-各個頁面及功能

all_psy.py

書目管理

管理者--書目管理

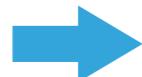
書目刪除 書目新增 全部書目 回到管理者首頁

登出

```
1256 @app.route('/manager_book/choose/<account>/<password>')
1257 def book_choose(account,password):
```

輸入要刪除書目的關鍵字

書目刪除



關鍵字 搜尋

[回到管理者首頁](#)

登出

```
1299 @app.route('/delete_book/search/<account>/<password>', methods=['GET'])
1300 def delete_book_search(account,password):
```

輸入要刪除的書目編碼

書目編碼 : Go

書目編碼	書名	出版社	是否已借出	成員編碼	成員身份	成員班級	成員姓名	借出時間
B00999	和機器王的飲食之旅		未借出	None	None	None	None	None
B99997	mad	kkk	未借出	None	None	None	None	None

[回到管理者首頁](#)

```
1343 @app.route('/delete_book/input/<account>/<password>', methods=['GET'])
1344 def delete_book_input(account,password):
```

確認要刪除？

書目編碼	書名	出版社	是否已借出	成員編碼	成員身份	成員班級	成員姓名	借出時間
B99997	mad	kkk	未借出	None	None	None	None	None

確認

[回到管理者首頁](#)

```
1425 @app.route('/delete_book/sure/<account>/<password>/<key_word>', methods=['GET'])
1426 def delete_book_sure(account,password,key_word):
```

刪除成功

書目編碼	書名	出版社	借書狀況
B99997	mad	kkk	未借出

[繼續刪除](#)

[回到管理者首頁](#)

1511

```
@app.route('/delete_book/solution/<account>/<password>/<select_book_id>', methods=['GET'])
def delete_book_solution(account,password,select_book_id):
```

登出

第二步-各個頁面及功能

all_psy.py



帳號管理



輸入要更改的帳號

輸入帳號

Go

回到管理者首頁

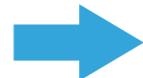
2551

```
@app.route('/manager_change/input/<account>/<password>')
```

2552

```
def manager_change_input(account,password):
```

登出



管理者--成員管理

更改密碼

更改信箱

回到管理者首頁

登出

2583

```
@app.route('/manager_change/choose/<account>/<password>')
```

2584

```
def manager_change_choose(account,password):
```

管理者--成員管理

第二步-各個頁面及功能

all_psy.py

帳號管理

更改密碼

更改信箱

回到管理者首頁

登出

```
2633 @app.route('/manager_change/password/<account>/<password>', methods=['GET', 'POST'])
2634 def manager_change_password(account, password, change_account):
```

更改密碼

原密碼

新密碼

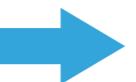
確認新密碼

Go

[回到管理者首頁](#)

[登出](#)

更改密碼



更改密碼

equality3806384密碼更改成功！

[回到管理者首頁](#)

[登出](#)

```
2730 @app.route('/change/email/input/<account>/<password>/<change_account>', methods=['GET', 'POST'])
2731 def change_email_input(account, password, change_account):
```

更改信箱

輸入密碼

輸入新的信箱

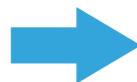
再次輸入新的信箱

Go

[回到管理者首頁](#)

[登出](#)

更改信箱



信息發送成功！

請去

[新信箱](#)

收信確認信箱喔！若您的新信箱沒有收到更改成功的信，請按下面重新設定信箱的按鈕再重新設定一次喔！

[重新設定信箱](#)

[回到首頁](#)

[登出](#)

```
2762 @app.route('/change/email/solution/<account>/<password>', methods=['GET', 'POST'])
2763 def change_email_solution(account, password, change_account):
```

管理者--成員管理

第二步-各個頁面及功能

all_psy.py

帳號管理

更改密碼 更改信箱 回到管理者首頁

登出

```
2633 @app.route('/manager_change/password/<account>/<password>')
2634 def manager_change_password(account,password,change_ac
```

更改密碼

原密碼

新密碼

確認新密碼 Go

更改密碼



[回到管理者首頁](#)

[登出](#)

更改密碼

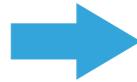
equality3806384密碼更改成功！

[回到管理者首頁](#)

[登出](#)

```
2669 @app.route('/manager_change/solution/<account>/<password>')
2670 def manager_change_solution(change_account,account,pa
```

更改信箱



更改信箱

輸入密碼

輸入新的信箱

再次輸入新的信箱

Go

[回到管理者首頁](#)

[登出](#)

更改信箱成功！

若您有收到這封信，表示您的信箱更改成功！

```
2730 @app.route('/change_email')
2731 def change_email():
    account = request.args.get('account')
```

equality.nknu@gmail.com

已傳送 -...@gmail.com 上午4:40

更改信箱成功！

收件人：

新信箱沒有收到更改成功的信，請按下面重新設定信
頁

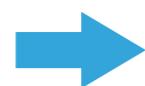
[ion/<account>/<password>,
nt, password, change_account\)](#)

第二步-各個頁面及功能

all_psy.py



教具管理



管理者--教具管理

```
3782 @app.route('/manager_teachaid/choose/<account>/<password>')  
3783 def teachaid_choose(account,password):
```

教具刪除

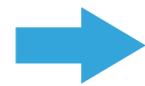
教具新增

全部教具

回到管理者首頁

登出

全部教具



教具列表

```
3822 @app.route('/all_teachaid/<account>/<password>', methods=['GET'])  
3823 def all_tea(account,password):
```

回到管理者首頁

教具編碼	教具名	財產編號	是否已借出
T00001	凸形積木1-1	007-4-07-F-01	已借出
T00002	凸形積木1-2	007-4-07-F-01	已借出
T00003	立體拼圖(小熊)	007-4-07-F-18	已借出
T00004	立體拼圖(獅子)	007-4-07-F-19	已借出
T00005	拼圖(繪本)	007-4-07-F-20	已借出
T00006	蘋果樹	007-4-07-F-21	已借出
T00007	幾何圖形組	007-4-07-F-22	已借出

第二步-各個頁面及功能

all_psy.py

教具管理



教具刪除 教具新增 全部教具 回到管理者首頁

登出

```
3863 @app.route('/insert_teachaid/input/<account>/<password>', methods=['POST'])
3864 def insert_tea_input(account, password):
```

新增教具

教具編碼

• 編碼開頭必須為 T，後面加上五碼數字

教具名

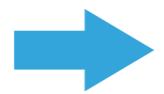
財產編號

新增

回到管理者首頁

登出

教具新增



新增成功

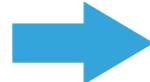
教具編碼	教具名	財產編號	借出狀況
------	-----	------	------

T99994	kk	mm	未借出
--------	----	----	-----

繼續新增

回到管理者首頁

```
3907 @app.route('/insert_teachaid/solution/<account>/<password>', methods=['POST'])
3908 def insert_tea_solution(account, password):
```



第二步-各個頁面及功能

all_psy.py

教具管理



教具刪除 教具新增 全部教具 回到管理者首頁
登出

輸入要刪除教具的關鍵字

教具刪除



關鍵字 | 搜尋

```
4014 @app.route('/delete_teachaid/search/<account>/<password>')
4015 def delete_tea_search(account,password):
```

回到管理者首頁

登出

輸入要刪除的教具編碼

```
4058 @app.route('/delete_teachaid/input/<account>/<password>')
4059 def delete_tea_input(account,password):
```

教具編碼 : Go									
教具編碼	教具名	財產編號	是否已借出	成員編碼	成員身份	成員班級	成員姓名	借出時間	
T99994	kk	mm	未借出	None	None	None	None	None	
T99998	test	test	已借出	00093	老師	白鯨村教室	教室(3)	2019/03/09, 11時26秒	

[回到管理者首頁](#)

[登出](#)

確認要刪除？

教具編碼	教具名	財產編碼	是否已借出	成員編碼	成員身份	成員班級	成員姓名	借出時間
T99994	kk	mm	未借出	None	None	None	None	None

[確認](#)

[回到管理者首頁](#)

[首頁](#)

```
4140 @app.route('/delete_teachaid/sure/<account>/<password>')
4141 def delete_tea_sure(account,password,key_word):
```

```
4226 @app.route('/delete_teachaid/solution/<account>/<password>')
4227 def delete_tea_solution(account,password,select_tea):
```

刪除成功

教具編碼	教具名	財產號碼	借出狀況
T99994	kk	mm	未借出

[繼續刪除](#)

[回到管理者首頁](#)

[登出](#)

第二步-各個頁面及功能

all_psy.py



教具搜尋

關鍵字 搜尋

全部教具 回到首頁

登出

3141 @app.route('/teachaid/search/input/<account>/<password>')
3142 def search_teachaid_input(account,password):

教具列表

回到管理者首頁

教具編碼	教具名	財產編號	是否已借出
T99998	test	test	已借出

回到管理者首頁

登出

3170 @app.route('/teachaid/search/solution/<account>/<password>')
3171 def search_teachaid_solution(account,password):

第二步-各個頁面及功能

all_psy.py



只有老師及班級能夠借教具

教具借閱

3216
3217

```
@app.route('/teachaid/borrow/peo/input/<account>/<password>')
def borrow_teachaid_peo(account, password):
```

[教具借閱](#)


成員編碼

Go

[回到管理者首頁](#)[登出](#)

教具借閱

成員編碼：00097 身份：家長 班級：家長03 姓名：家長0

教具編碼

Go

[下一位](#)[回到管理者首頁](#)[登出](#)3249
3250

```
@app.route('/borrow/teachaid_id/input/<account>/<password>')
def borrow_teachaid(account, password, peo_word, the_id):
```

3343
3344

```
@app.route('/borrow/teachaid/solution/<account>/<password>')
def borrow_teachaid_solution(account, password):
```

T00088--兩片點數拼圖：借書成功！

教具借閱

成員編碼：00093 身份：老師 班級：白鯨村教室 姓名：教室(3)

教具編碼

Go

借出的書：

教具編碼	教具名	借出時間
T99998	test	2019/03/09, 11時26分33秒
T00088	兩片點數拼圖	2019/06/15, 22時16分54秒
T00001	□形積木1-1	2019/03/03, 13時41分25秒
T00007	幾何圖形組	2019/03/03, 13時41分33秒

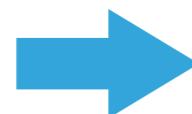
[下一位](#)[回到管理者首頁](#)

第二步-各個頁面及功能

all_psy.py



教具歸還



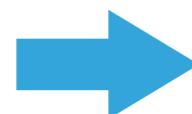
教具歸還

```
3471 @app.route('/return/teachaid_id/input/<account>/<pa
3472 def return_teachaid(account,password):
```

教具編碼 |

[回到管理者首頁](#)

[登出](#)



T99998--test：歸還成功！

教具歸還

成員編碼：00093 身份：老師 班級：白鯨村教室 姓名：教室(3)

書目編碼 |

未還的教具：

教具編碼	教具名	借出時間
T00088	兩片點數拼圖	2019/06/15, 22時16分54秒
T00001	□形積木1-1	2019/03/03, 13時41分25秒
T00007	幾何圖形組	2019/03/03, 13時41分33秒

[回到管理者首頁](#)

```
3506 @app.route('/return/teachaid/solution/<account>/<pa
3507 def return_teachaid_solution(account,password,submi
```

[登出](#)

第二步-各個頁面及功能

all_psy.py



借出搜尋

3644 @app.route('/unreturn/teachaid/<account>/<password>')
 3645 def unreturn_teachaid(account,password):

搜尋輸入

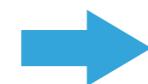
• 可輸入教具編碼、教具名、成員編碼、成員姓名、成員班級、成員身份等關鍵字。

搜尋方式 Go

[全部教具借出清單](#) [回到管理者首頁](#)

[登出](#)

教具借出清單

**借出搜尋結果**

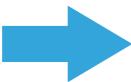
3677 @app.route('/unreturn/teachaid/solution/<account>/<password>')
 3678 def unreturn_teachaid_solution(account,password):

回到管理者首頁									
教具編碼	書名	財產名稱	是否已借出	成員編碼	成員身份	成員班級	成員姓名	借出時間	
T00004	立體拼圖(獅子)	2019/03/03, 13時33分13秒	007-4-07-F-19	已借出	00091	老師	海馬村教室	(1)	
T00002	□形積木1-2	2019/03/03, 13時33分13秒	007-4-07-F-01	已借出	00091	老師	海馬村教室	(1)	

or

借出搜尋結果**全部借出的教具**

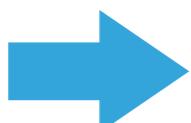
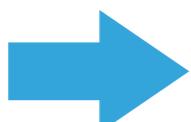
回到管理者首頁									
教具編碼	書名	財產名稱	是否已借出	成員編碼	成員身份	成員班級	成員姓名	借出時間	
T00004	立體拼圖(獅子)	2019/03/03, 13時33分13秒	007-4-07-F-19	已借出	00091	老師	海馬村教室	(1)	
T00002	□形積木1-2	2019/03/03, 13時33分13秒	007-4-07-F-01	已借出	3743	3744	3743	3744	@app.route('/unreturn_all/teachaid/<account>/<password>') def unreturn_all_teachaid(account,password):



第二步-各個頁面及功能

all_psy.py





忘記密碼

登入

帳號

密碼 Go

[忘記密碼](#)

[首頁](#)



忘記密碼

帳號

[首頁](#)

```
177 @app.route('/forget_password/input', methods=['GET'])
178 def forget_input():
```

信息發送成功！

信箱

收信喔並更改密碼喔！

[首頁](#)

```
182 @app.route('/forget_password/solution', methods=['GET'])
183 def forget_solution():
```



更改密碼

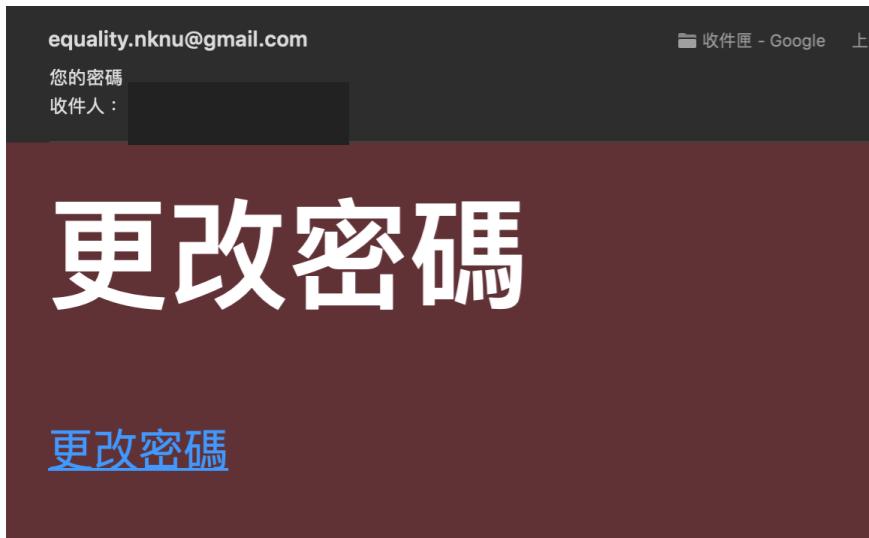
[更改密碼](#)

收件匣 - Google 上午

equality.nknu@gmail.com

您的密碼
收件人：

忘記密碼



更改密碼

[更改密碼](#)

在reset 密碼時，會傳一個變數token，是為了限制更改密碼的時間

```
216 def validate_confirm_token(token):
217     """驗證回傳令牌是否正確，若正確則回傳True:param token:驗證令牌:return:回傳驗證是否正確，正確為True"""
218     s = TimedJSONWebSignatureSerializer(current_app.config['SECRET_KEY'])
219     try:
220         data = s.loads(token) # 驗證
221     except SignatureExpired:
222         # 當時間超過的時候就會引發SignatureExpired錯誤
223         return False
224     except BadSignature:
225         # 當驗證錯誤的時候就會引發BadSignature錯誤
226         return False
227     return data
```

上面這個函數就是讓超過時間的連結失效

```
236 @app.route('/forget_reset/<account>/<token>', methods=['POST'])
237 def forget_reset(account, token):
```

變更密碼

新密碼

確認新密碼

[確認](#)

[首頁](#)

更新成功！

```
113 @app.route('/login/solution', methods=['POST'])
114 def login_solution():
```

登入

帳號

密碼 [Go](#)

[忘記密碼](#)

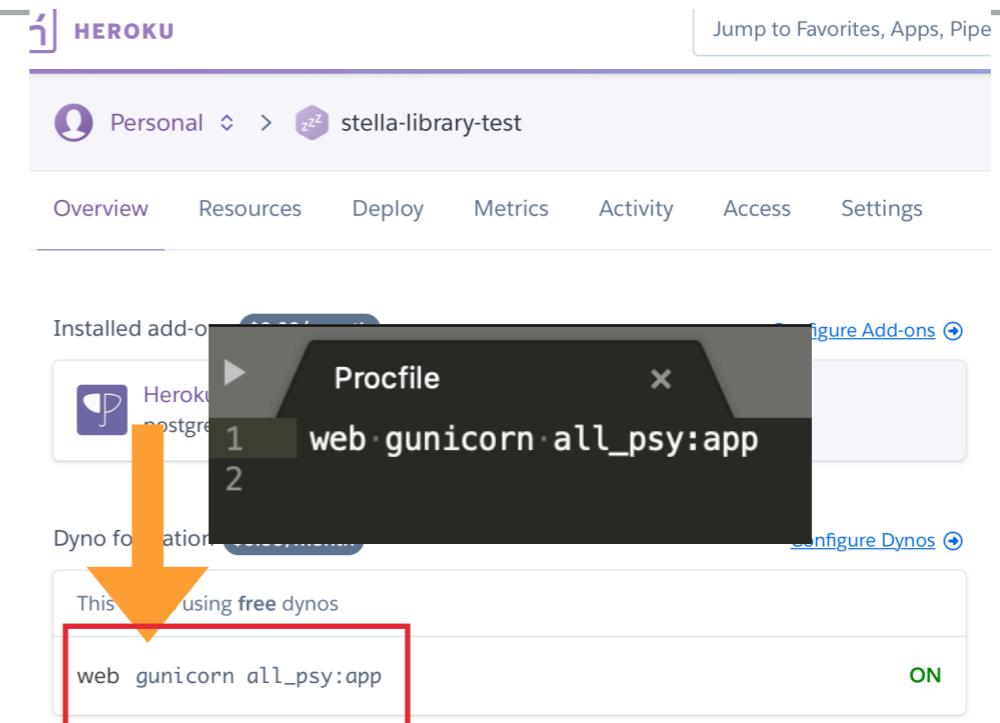
[首頁](#)

第三步

準備剩下所需的檔案

▶ Profile

- 是讓heroku知道要跑哪一個程式，所以如果主程式要改名，這裡的 all_psyc也要改



▶ requirements.txt

- 告訴 heroku有哪些 package要先 install

```
1 gunicorn
2 WTForms
3 requests
4 Werkzeug
5 Flask
6 Flask-Admin
7 Flask-Cors
8 Flask-Login
9 Flask-Mail
10 Flask-Migrate
11 Flask-Script
12 Flask-WTF
13 anaconda-project
14 app
15 pandas
16 psycopg2
```

第四步 上傳HEROKU

打開終端機 (cmd)

進入資料夾

```
$ cd 資料夾路徑
```

登入heroku

```
$ heroku login
```

初始化git

初始化 git

```
1 $ git config --global user.name "你的名字"  
2 $ git config --global user.email 你的信箱
```

注意：你的名字 和 你的信箱 要換成各自的名字 和 信箱

初始化 git

```
1 git init
```

注意：僅第一次使用時要輸入

用git將資料夾及heroku連接

```
heroku git:remote -a {HEROKU_APP_NAME}
```

heroku_app_name 是填你自己的app名稱

上傳

```
git add .  
git commit -m "Add code"  
git push -f heroku master
```

Git add 後面的點是指將資料夾中裡全部的檔案上傳，**不建議這麼做**，因為會容易髒掉，會不知道自己有沒有多上傳東西

建議一個一個檔案加進去，因為我們檔案沒有很多，直接在add 後面打檔案 (ex. all_psy.py) ，檔案和檔案中間空格

完成！

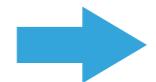
信箱設定參考

- ▶ <https://github.com/twtrubiks/Flask-Mail-example>

FLASK 參考

- ▶ <https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-i-hello-world>
- ▶ <https://blog.liang2.tw/posts/2015/09/flask-draw-member/>
- ▶ <http://www.bjhee.com/flask-ext7.html>

all_psy.py



or

全部借出的書