1. 系統介紹
   1. 系統架構功能介紹

此餐點POS系統分成兩部分：

* + 1. 會員制點餐系統
    2. 餐點及訂單管理系統
  1. 資料庫建置

使用了Python的sqlite3模組來建立和操作SQLite資料庫，資料庫建置及資料表屬性如下：

* + - 1. 會員資料表(users)：
         * id (主鍵)：會員ID
         * username：會員名稱
         * password：密碼
         * phone：手機號碼
      2. 訂單資料表(orders)
* Id：訂單編號
* user\_id(外鍵)：會員ID，可直接連結會員資料
* table\_id(外鍵) ：餐點清單ID，可直接連結清單的餐點資料
* timestamp：訂單成立時間
  + - 1. 購買清單資料表(carts)
* id(主鍵)：購買清單ID
* table\_id(外鍵)：餐點清單ID
* item\_id(外鍵)：餐點ID，查詢資料
* quanitity：餐點數量
  + - 1. 餐點資料表(menu\_items)
* id(主鍵)：餐點ID
* categories (外鍵)：種類ID，對應到種類ID
* name：餐點名稱
* image：圖片
* price：金額
* description：餐點介紹
  + - 1. 餐點種類資料表(categories)
         * Id：種類ID
         * name：名稱

1. 點餐系統
   1. 點餐流程圖
   2. 點餐頁面

一張含有 文字, 食物, 餐點, 點心 的圖片

自動產生的描述在點餐頁面中，可以看到所有的餐點，並且可透過左側的種類清單，顯示特定的餐點種類。

* + 1. 加入清單

一張含有 文字, 食物, 菜餚, 盤子 的圖片

自動產生的描述一張含有 文字, 食物, 菜餚, 盤子 的圖片

自動產生的描述在餐點中，可以選擇數量，並加入購買清單，先建立一個add\_to\_cart()的函式，以尋找orders中table\_id 的最大值，並加該值進行累加，作為新的清單。

* 網頁程式碼

|  |
| --- |
| <div class="container-lg text-center">  <div class="row align-items-center">  <div class="col-md-3 align-items-top">  <!--list-->  <div  class="list-group list-group-flush list-group-item-action shadow-sm mb-5 bg-body rounded"  >  <a  href="{{url\_for('home')}}"  class="list-group-item list-group-item-action active"  aria-current="true"  >  全部餐點  </a>  {%for category in categories%}  <a  href="/{{category.0}}"  class="list-group-item list-group-item-action {%if selected\_category== category.0%}active{%endif%}"  aria-current="true"  >  {{category.0}}  </a>  {%endfor%} {%if cart\_items%}  <button  type="button"  class="list-group-item list-group-item-action"  data-bs-toggle="collapse"  href="#cart"  role="button"  aria-expanded="false"  aria-controls="collapseExample"  >  清單  </button>  {% endif%}  </div>  <!--list-->  </div>  <div class="col-md-1"></div>  <!--餐點頁面-->  <div class="col-md-8">  <nav aria-label="breadcrumb">  <ol class="breadcrumb">  {% if selected\_category == 'all'%}  <li class="breadcrumb-item">  <a href="{{url\_for('home')}}">全部餐點</a>  </li>  {%else%}  <li class="breadcrumb-item">  <a href="{{url\_for('home')}}">全部餐點</a>  </li>  <li class="breadcrumb-item active" aria-current="page">  {{selected\_category}}  </li>  {%endif%}  </ol>  </nav>  <div class="row g-3 align-items-center row-cols-3">  <div class="collapse col-12" id="cart">  <!--訂單-->  <div class="card">  <div class="card-title">點餐列表</div>  <div class="card-body">  <table class="table">  <thead>  <tr>  <th scope="col">餐點</th>  <th scope="col">數量</th>  <th scope="col">金額</th>  <th scope="col">刪除</th>  </tr>  </thead>  <tbody>  {% for item in cart\_items %}  <tr>  <th scope="col">{{item.0}}</th>  <th scope="col">{{item.1}}</th>  <th scope="col">{{item.2}}</th>  <th scope="col">  <form  action="{{ url\_for('delete\_cart\_item', cart\_id=item.3) }}"  method="POST"  >  <input type="hidden" name="page" value="home" />  <button class="btn btn-danger" type="submit">  刪除  </button>  </form>  </th>  </tr>  {% endfor %}  </tbody>  </table>  <a href="{{ url\_for('cart\_detail') }}" class="btn btn-primary"  >購物車</a  >  </div>  </div>  <!--訂單-->  </div>  {% for item in menu\_items %}  <div class="col">  <div class="card shadow-sm bg-body rounded">  <img  src="/{{item.3}}"  style="max-height: 10rem"  class="card-img-top"  alt="{{item.2}}"  />  <div class="card-body">  <div class="card-title">{{item.2}}　　${{item.4}}</div>  <p class="card-text">{{item.5}}</p>  <form action="{{ url\_for('add\_to\_cart') }}" method="POST">  <div class="row">  <div class="col-md-3">  <select  class="form-select"  name="quantity"  style="width: 4rem"  >  <option value="0" selected>0</option>  {% for num in range(1, 10) %}  <option value="{{ num }}">{{ num }}</option>  {% endfor %}  </select>  </div>  <input type="hidden" name="item\_id" value="{{ item.0 }}" />  <div class="col-md-9">  <button type="submit" class="btn btn-primary">  加入清單  </button>  </div>  </div>  </form>  </div>  </div>  </div>  {% endfor %}  </div>  </div>  <!--餐點頁面-->  </div>  </div> |

* 加入清單Function

|  |
| --- |
| @app.route('/add\_to\_cart', methods=['POST'])  def add\_to\_cart():  item\_id = request.form['item\_id']  quantity = int(request.form['quantity'])  conn = create\_connection()  cursor = conn.cursor()  try:  # 查找 orders 表中 table\_id 的最大值  cursor.execute('SELECT MAX(table\_id) FROM orders')  max\_table\_id = cursor.fetchone()[0]  if max\_table\_id is not None:  table\_id = max\_table\_id + 1  else:  table\_id = 1  # 檢查購物車中是否已經存在該餐點，若存在則更新數量，否則新增一筆資料  cursor.execute('SELECT \* FROM carts WHERE item\_id = ? AND table\_id = ? LIMIT 1', (item\_id, table\_id))  existing\_item = cursor.fetchone()  if existing\_item:  new\_quantity = existing\_item[3] + quantity  cursor.execute('UPDATE carts SET quantity = ? WHERE item\_id = ? AND table\_id = ?', (new\_quantity, item\_id,table\_id))  else:  cursor.execute('INSERT INTO carts (table\_id, item\_id, quantity) VALUES (?, ?, ?)', (table\_id, item\_id, quantity))    conn.commit()  except Error as e:  conn.rollback()  flash(f'Error adding item to cart: {e}')  finally:  conn.close()  return redirect(url\_for('home')) |

* + 1. 顯示購買清單

一張含有 螢幕擷取畫面, 文字 的圖片

自動產生的描述建立一個showcart()，此函式先連線至資料庫取的取得已轉成訂單的table\_id最大值，再將該值累加，成為新的table\_id，會顯示所購買的項目至清單的網頁內。

* 網頁程式碼

|  |
| --- |
| <!--訂單-->  <div class="card">  <div class="card-title">點餐列表</div>  <div class="card-body">  <table class="table">  <thead>  <tr>  <th scope="col">餐點</th>  <th scope="col">數量</th>  <th scope="col">金額</th>  <th scope="col">刪除</th>  </tr>  </thead>  <tbody>  {% for item in cart\_items %}  <tr>  <th scope="col">{{item.0}}</th>  <th scope="col">{{item.1}}</th>  <th scope="col">{{item.2}}</th>  <th scope="col">  <form  action="{{ url\_for('delete\_cart\_item', cart\_id=item.3) }}"  method="POST"  >  <input type="hidden" name="page" value="home" />  <button class="btn btn-danger" type="submit">  刪除  </button>  </form>  </th>  </tr>  {% endfor %}  </tbody>  </table>  <a href="{{ url\_for('cart\_detail') }}" class="btn btn-primary"  >購物車</a  >  </div>  </div>  <!--訂單--> |

* 使用Function

建立一個showcart()，此函式先連線至資料庫取的取得已轉成訂單的table\_id最大值，再將該值累加，成為新的table\_id，會顯示所購買的項目至清單的網頁內。

|  |
| --- |
| def show\_cart():  conn = create\_connection()  cursor = conn.cursor()  cursor.execute('SELECT MAX(table\_id) FROM orders')  max\_table\_id = cursor.fetchone()[0]  if max\_table\_id is not None:  table\_id = max\_table\_id + 1  else:  table\_id = 1  cursor.execute('''  SELECT menu\_items.name, carts.quantity, menu\_items.price,carts.id  FROM carts  INNER JOIN menu\_items ON carts.item\_id = menu\_items.id  WHERE carts.table\_id = ?  ''', (table\_id,))  cart\_items = cursor.fetchall()  conn.close()  return cart\_items |

* + 1. 刪除餐點

購買清單中，建立一個delete\_cart\_item ()的函式，以尋找購買清單中該項目的id，並對該資料進行刪除，最後依據所點選刪除的頁面進行網頁刷新。

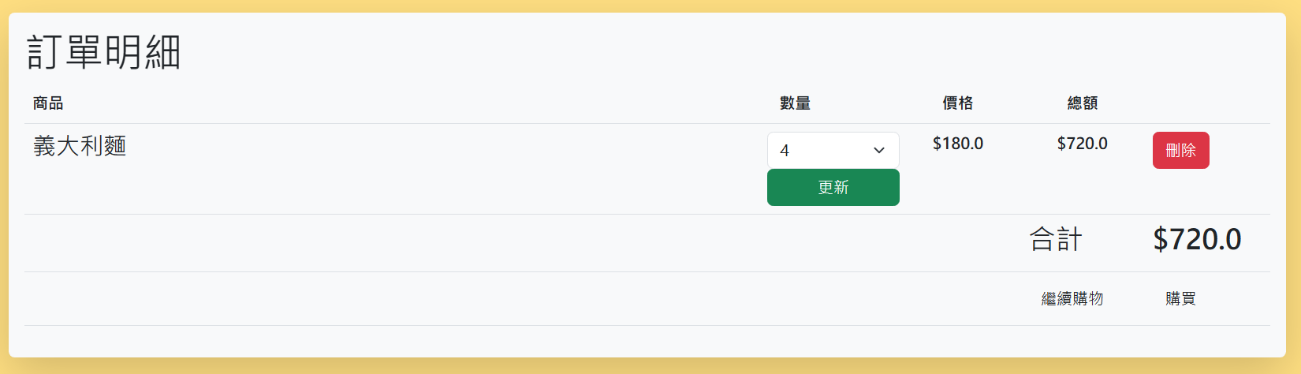
* 網頁程式碼

|  |
| --- |
| <form  action="{{ url\_for('delete\_cart\_item', cart\_id=item.3) }}"  method="POST"  >  <input type="hidden" name="page" value="home" />  <button class="btn btn-danger" type="submit">  刪除  </button>  </form> |

* 使用Function

|  |
| --- |
| @app.route('/delete\_cart\_item/<cart\_id>', methods=['POST'])  def delete\_cart\_item(cart\_id,page='home'):    # 刪除餐點  conn = create\_connection()  cursor = conn.cursor()  cursor.execute("DELETE FROM carts WHERE id = ?",(cart\_id,))  # 提交變更到資料庫  conn.commit()  conn.close()  if request.form["page"] =='for\_order':  return redirect(url\_for('cart\_detail'))  elif request.form["page"] =='home':  return redirect(url\_for('home'))  return redirect(url\_for('edit\_order' , order\_id = request.form['page'])) |

* 1. 購買清單



* + 1. 餐點數量更新

在購買清單中，可針對該項目的數量進行更改，透過連線資料庫找到carts\_id，即可進行數量的變更。

* + - * 網頁程式碼

|  |
| --- |
| <form  action="{{ url\_for('update\_cart\_item', cart\_id=item.3 ) }}"  method="post"  >  <div class="row">  <select name="quantity" class="col-m-1 form-select">  {% for num in range(1,10)%} {% if num == item.1%}  <option value="{{item.1}}" selected>{{item.1}}</option>  {%else%}  <option value="{{num}}">{{num}}</option>  {%endif%}  {%endfor%}  </select>  <input type="hidden" name="page" value="for\_order" />  <button type="submit" class="col-m-1 btn btn-success btn-product">  更新  </button>  </div>    </form> |

* + - * 使用Function

|  |
| --- |
| @app.route('/update\_cart\_item/<cart\_id>', methods=['POST'])  def update\_cart\_item(cart\_id):  conn = create\_connection()  cursor = conn.cursor()  quantity = int(request.form['quantity'])  cursor.execute("UPDATE carts SET quantity = ? WHERE id = ?",(quantity,cart\_id))  conn.commit()  conn.close()  if request.form['page'] == "for\_order":  return redirect(url\_for('cart\_detail'))  return redirect(url\_for('edit\_order' , order\_id = request.form['page'])) |

* + 1. 餐點刪除

同點餐頁面中的購物車都是透過delete\_cart\_item()進行刪除。

* + - * 網頁程式碼

|  |
| --- |
| <form  action="{{ url\_for('delete\_cart\_item', cart\_id=item.3) }}"  method="POST"  >  <button  class="btn btn-danger"  type="submit"  name="page" value="for\_order"/>  刪除  </button>  </form> |

* + 1. 進行購買

確認後，點選購買，會透過cookies機制判斷有無登入，若有登入即成立訂單；若沒有則要求登入。

透過cart\_to\_order()

* + - * 網頁程式碼

|  |
| --- |
| {% if request.cookies.get('user\_phone') %}  <a href="{{url\_for('cart\_to\_order' , phone =request.cookies.get('user\_phone') )}}" class="btn btn-default"  >購買<i class="glyphicon glyphicon-play" aria-hidden="true"></i>  </a>  {% else %}  <a href="{{url\_for('login' , status="to\_order")}}" class="btn btn-default"  >購買<i class="glyphicon glyphicon-play" aria-hidden="true"></i>  </a>  {% endif %} |

* + - * 使用Function

|  |
| --- |
| @app.route('/cart\_to\_order/<phone>')  def cart\_to\_order(phone):  user\_id = create\_order\_cart(phone)  conn = create\_connection()  cursor = conn.cursor()  cursor.execute('SELECT MAX(id) FROM orders WHERE user\_id=?',(user\_id,))  order\_id = cursor.fetchone()[0]  return redirect(url\_for('created\_order',order\_id=order\_id)) |

* + 1. 金額計算

透過jinja2的模板，將資料導入至網頁後，取得total變數的資料。

* + - * 網頁程式碼

|  |
| --- |
| <tr>  <td> </td>  <td> </td>  <td> </td>  <td><h3>合計</h3></td>  <td class="text-right">  <h3><strong>${{total}}</strong></h3>  </td>  </tr> |

* 1. 會員機制
     1. 註冊、登入功能介紹

會員主要是紀錄會員名稱、手機號碼、密碼，另外在訂單當中，也會記錄會員的ID，以便做查詢。

註冊頁面：

一張含有 文字, 螢幕擷取畫面, 數字, 字型 的圖片

自動產生的描述

|  |
| --- |
| # 註冊會員頁面  @app.route('/Register', methods=['GET', 'POST'])  def register():  if request.method == 'POST':  username = request.form.get('username')  phone = request.form.get('phone')  password = request.form.get('password')    # 儲存會員資料到資料庫  save\_user\_to\_db(username, phone, password)    return redirect(url\_for('login'))    return render\_template('register.html') |

登入頁面：

一張含有 文字, 螢幕擷取畫面, 數字, 字型 的圖片

自動產生的描述

|  |
| --- |
| # 登入頁面  @app.route('/Login', methods=['GET', 'POST'])  @app.route('/Login/<status>', methods=['GET', 'POST'])  def login(status= 'to\_history'):    if request.method == 'POST':  phone = request.form.get('phone')  password = request.form.get('password')  status = request.form.get('status')  # 檢查會員資料庫是否有該手機號碼和密碼  if check\_user\_credentials(phone, password):  # 登入成功，執行成立訂單  if status == "to\_order":  ''' 設定 Cookie '''  # order\_id = cart\_to\_order(phone)  response = make\_response(cart\_to\_order(phone))  response.set\_cookie('islogin',value='true')  response.set\_cookie('user\_id', phone)  response.set\_cookie('user\_phone', phone)  return response  elif status == "to\_history":  response = make\_response(redirect('/'))  response.set\_cookie('islogin',value='true')  response.set\_cookie('user\_id', phone)  response.set\_cookie('user\_phone', phone)  return response  # 登入失敗，顯示錯誤訊息  error\_message = '手機號碼或密碼不正確'  return render\_template('login.html', error\_message=error\_message,status=status)  return render\_template('login.html',status=status) |

* + 1. Cookies機制
       - 登入:

會依據所在頁面，判斷要引導至哪個頁面，而設定的cookies都有所不同，若登入成功，會設定islogin的值為true、user\_phone儲存手機號碼，user\_id儲存會員id。

|  |
| --- |
| if status == "to\_order":  ''' 設定 Cookie '''  # order\_id = cart\_to\_order(phone)  response = make\_response(cart\_to\_order(phone))  response.set\_cookie('islogin',value='true')  response.set\_cookie('user\_id', phone)  response.set\_cookie('user\_phone', phone)  return response  elif status == "to\_history":  response = make\_response(redirect('/'))  response.set\_cookie('islogin',value='true')  response.set\_cookie('user\_id', phone)  response.set\_cookie('user\_phone', phone)  return response |

* + - * 餐點頁面:

當載入頁面時，會判斷有無isLogin的cookies，若沒有則會預設isLogin為false

|  |
| --- |
| ''' 檢查 Cookie '''  response = make\_response(render\_template('index.html', categories=categories, menu\_items=menu\_items,selected\_category=selected\_category,cart\_items=cart\_items))  if 'islogin' not in request.cookies:  response.set\_cookie('islogin',value='false') |

* 1. 訂單顯示

透過created\_order找尋特定orders\_id，顯示已經成立的訂單資料，並透過table\_id、user\_id，取得訂單資料及會員資料。

* + - 網頁程式碼

|  |
| --- |
| {% extends 'base.html' %} {% block content %}  <div class="row justify-content-center">  <div class="col-2"></div>  <div class="col-8">  <!--訂單編輯-->  <div class="card">  <div class="card-body text-start">  <h5 class="card-title">訂單成立!!</h5>  <p class="card-text">  <div class="col justify-content-start">  <div class="col-6">  訂單編號:#{{ order\_number[0] }}    </div>  <div class="col-6">  會員名稱:{{ username }}  </div>  <div class="col-6">  連絡電話: {{ phone }}  </div>  <div class="col-6">  成立時間:{{ order\_number[3]| custom\_datetimeformat }}  </div>  </div>  </p>  <h6 class="card-subtitle mb-2 text-body-secondary">訂單餐點</h6>  <div class="card-text">  <table class="table">  <thead>  <tr>  <th scope="col">餐點</th>  <th scope="col">數量</th>  <th scope="col">金額</th>  </tr>  </thead>  <tbody>  {% for item in items %}  <tr>  <th scope="row">{{ item[0] }}</th>  <td>  {{item[2] }}  </td>  <td>{{ item[1]\*item[2]}}</td>  </tr>  {% endfor %}  <tr>  <td>  <h1>總額:$ {{ total\_price }}</h1>  </td>  </tr>  </tbody>  </table>  </div>  <div class="row justify-content-end">  <div class="col-4">  <a href="" class="btn btn-outline-secondary card-link">歷史紀錄</a>  <a href="{{url\_for('home')}}" class="btn btn-secondary card-link">回首頁</a>  </div>  </div>  </div>  </div>  <!--訂單編輯-->  </div>  <div class="col-2"></div>  </div>  {% endblock content%} |

* + - Function

|  |
| --- |
| """  訂單成立資料顯示  - 顯示:訂單編號、會員名稱、手機號碼、餐點名稱、數量訂單成立時間、總額  """  @app.route('/order/<order\_id>', methods=['GET', 'POST'])  def created\_order(order\_id):  conn = create\_connection()  cursor = conn.cursor()  # 訂單資料  cursor.execute('SELECT id, user\_id, table\_id, timestamp FROM orders WHERE id = ?', (order\_id,))  order = cursor.fetchone()  # 會員資料  cursor.execute('SELECT username, phone FROM users WHERE id = ?', (order[1],))  user = cursor.fetchone()  # 餐點資訊  cursor.execute('''  SELECT menu\_items.name, menu\_items.price ,carts.quantity  FROM carts  INNER JOIN menu\_items ON carts.item\_id = menu\_items.id  WHERE carts.table\_id = ?  ''', (order[2],))  cart\_items = cursor.fetchall()  # 總額  total\_price = 0  for item in cart\_items:  item\_price = item[1]  total\_price += item\_price \* item[2]  conn.close()  return render\_template('order.html', order\_number=order, username=user[0], phone=user[1], items=cart\_items, total\_price=total\_price) |

* 1. 歷史訂單

一張含有 文字, 螢幕擷取畫面, 字型, Rectangle 的圖片

自動產生的描述

* + - 程式碼

|  |
| --- |
| @app.route('/Order\_history', methods=['GET'])  def order\_history():  conn = create\_connection()  cursor = conn.cursor()  phone = request.cookies.get('user\_phone')  user\_id = cursor.execute('SELECT users.id FROM users WHERE users.phone = ?', (phone,)).fetchone()  print(phone,user\_id[0])  try:  # 歷史訂單  cursor.execute('SELECT orders.id, SUM(carts.quantity \* menu\_items.price), orders.timestamp , carts.quantity , menu\_items.name , menu\_items.price FROM orders INNER JOIN carts ON orders.table\_id = carts.table\_id INNER JOIN menu\_items ON carts.item\_id = menu\_items.id WHERE orders.user\_id = ? GROUP BY orders.id', (user\_id[0],))  orders = cursor.fetchall()  conn.close()  return render\_template('/restaurant/order\_history.html', orders=orders)  except Error as e:  print(f'Error retrieving order history: {e}')  conn.close()  return 'Error retrieving order history' |

* + - 網頁程式碼

|  |
| --- |
| {% extends 'base.html' %} {% block content %}  <div class="row">  {% for order in orders %}  <div class="col-lg-2 col-md-3 col-sm-4 col-6">  <div class="card shadow-lg p-3 mb-5 bg-body-tertiary">  <div class="card-body">  <h5 class="card-title">訂單編號: #{{ order[0] }}</h5>  <p class="card-text">總額: ${{ order[1] }}</p>  <p class="card-text">  成立時間: <br />{{ order[2] | custom\_datetimeformat }}  </p>  </div>  </div>  </div>  {% endfor %}  </div>  {% endblock %} |

1. 餐點管理系統
   1. 餐點管理流程圖
   2. 餐點管理

一張含有 文字, 螢幕擷取畫面, 數字, Rectangle 的圖片

自動產生的描述顯示餐點的資料，可進行新增、刪除、修改。

* + 1. 新增餐點

一張含有 文字, 螢幕擷取畫面, 軟體, 數字 的圖片

自動產生的描述如圖，可設定圖片、餐點種類、名稱、金額、介紹進行餐點新增。

|  |
| --- |
| @app.route('/admin/add\_menu\_item', methods=['GET', 'POST'])  def add\_menu\_item():  conn = create\_connection()  cursor = conn.cursor()  if request.method == 'POST':  file = request.files['image']  if file:  filename = secure\_filename(file.filename)  print(filename)  file\_path = f".{app.config['UPLOAD\_FOLDER']}/{filename}"  file.save(file\_path)  category = request.form['category']  name = request.form['name']  image = file\_path  price = request.form['price']  description = request.form['description']  cursor.execute('INSERT INTO menu\_items (categories, name, image, price, description) VALUES (?, ?, ?, ?, ?)',  (category, name, image, price, description))  conn.commit()  return redirect(url\_for('admin'))  cursor.execute('SELECT \* FROM categories')  categories = cursor.fetchall()  conn.close()  return render\_template('admin/add\_item.html', categories=categories) |

* + 1. 刪除餐點

|  |
| --- |
| """  刪除餐點  """  @app.route('/delete\_menu\_item/<item\_id>', methods=['POST'])  def delete\_menu\_item(item\_id):  conn = create\_connection()  cursor = conn.cursor()  # 執行刪除操作  cursor.execute("DELETE FROM menu\_items WHERE id=?", (item\_id,))  # 提交變更並關閉連接  conn.commit()  conn.close()  # 跳轉回顯示餐點的頁面  return redirect(url\_for('admin')) |

* + 1. 編輯餐點

一張含有 食物, 菜餚 的圖片

自動產生的描述

|  |
| --- |
| """  編輯餐點  """  @app.route('/edit\_menu\_item/<item\_id>', methods=['GET', 'POST'])  def edit\_menu\_item(item\_id):  if request.method == 'POST':  # 從表單中獲取修改後的餐點資料    file = request.files['image']  if file:  filename = secure\_filename(file.filename)  print(filename)  file\_path = f".{app.config['UPLOAD\_FOLDER']}/{filename}"  file.save(file\_path)  name = request.form['name']  category = request.form['category']  price = request.form['price']  description = request.form['description']  image = file\_path  # 連接到 SQLite 資料庫  conn = sqlite3.connect('database.db')  cursor = conn.cursor()  # 更新資料庫中對應的餐點資料  cursor.execute("UPDATE menu\_items SET name=?, categories=? , image=?, price=?, description=? WHERE id=?", (name, category, image, price, description, item\_id , ))  conn.commit()  # 關閉資料庫連接  conn.close()  # 跳轉回顯示餐點的頁面  return redirect(url\_for('admin'))  # 連接到 SQLite 資料庫  conn = sqlite3.connect('database.db')  cursor = conn.cursor()  # 根據 item\_id 從資料庫中獲取餐點資料  cursor.execute("SELECT \* FROM menu\_items WHERE id=?", (item\_id,))  item = cursor.fetchone()  #品項種類  cursor.execute('SELECT \* FROM categories')  categories = cursor.fetchall()  # 關閉資料庫連接  conn.close()  # 將餐點資料傳遞給編輯頁面進行顯示  return render\_template('admin/edit\_item.html', item\_id=item\_id, item=item,categories=categories) |

* 1. 訂單顯示
     1. 訂單管理

一張含有 文字, 螢幕擷取畫面, 數字, 軟體 的圖片

自動產生的描述連線至資料庫，取得訂單資料，並可針對清單進行更改。

|  |
| --- |
| @app.route('/edit\_order/<order\_id>', methods=['GET', 'POST'])  def edit\_order(order\_id):  conn = create\_connection()  cursor = conn.cursor()  # 取得訂單資料  cursor.execute('SELECT id, user\_id, timestamp FROM orders WHERE id = ?', (order\_id,))  order = cursor.fetchone()  # 取得會員資料  cursor.execute('SELECT username, phone FROM users WHERE id = ?', (order[1],))  user = cursor.fetchone()  # 取得購物車資料  cursor.execute('''  SELECT carts.id, menu\_items.name, carts.quantity, menu\_items.price  FROM carts  INNER JOIN menu\_items ON carts.item\_id = menu\_items.id  WHERE carts.table\_id = ?  ''', (order\_id,))  cart\_items = cursor.fetchall()  conn.close()  #總額  total\_price = 0  for item in cart\_items:  total\_price += item[3] \* item[2]  return render\_template('admin/edit\_orders.html', order\_id=order[0], username=user[0], phone=user[1], timestamp=order[2], cart\_items=cart\_items,total= total\_price) |

* + 1. 刪除定單

取得訂單ID，進行刪除，對應的table\_id也會連同刪除。

|  |
| --- |
| @app.route('/delete\_order/<order\_id>', methods=['GET','POST'])  def delete\_order(order\_id):  conn = create\_connection()  cursor = conn.cursor()  try:  # 删除订单及关联的购物车项  cursor.execute('DELETE FROM orders WHERE id = ?', (order\_id,))  cursor.execute('DELETE FROM carts WHERE table\_id = ?', (order\_id,))  conn.commit()  print('Order deleted successfully')  except Error as e:  conn.rollback()  print(f'Error deleting order: {e}')  finally:  conn.close()  return redirect(url\_for('manage\_orders')) |

* + 1. 訂單管理列表



* + - 網頁程式碼

|  |
| --- |
| {% extends '/sample/base.html' %} {% block content %}  <div class="row align-items-center justify-content-center">  <div class="col">  <div class="card">  <table class="table table-striped table-hover">  <thead>  <tr>  <th scope="col">訂單編號</th>  <th scope="col">會員</th>  <th scope="col">連絡電話</th>  <th scope="col">訂單成立時間</th>  <th scope="col"></th>  <th scope="col"></th>  </tr>  </thead>  <tbody class="table-group-divider">  {% for order in orders %}  <tr>  <th scope="row">{{ order[0] }}</th>  <td>{{ order[1] }}</td>  <td>{{ order[2] }}</td>  <td>{{ order[3]| custom\_datetimeformat }}</td>  <td colspan="2">  <button  type="button"  class="btn btn-primary"  onclick="window.location.href = '{{ url\_for('edit\_order', order\_id=order[0]) }}'"  >  編輯  </button>  </td>  <td colspan="2">  <form  action="{{ url\_for('delete\_order', order\_id=order[0]) }}"  method="POST"  onsubmit="return confirm('確定要刪除該訂單嗎？')"  >  <button type="submit" class="btn btn-danger">刪除訂單</button>  </form>  </td>  </tr>  {% endfor%}  </tbody>  </table>  </div>  </div>  </div>  {% endblock content%} |

* + - 程式碼

|  |
| --- |
| """  列出所有成立的訂單:  訂單編號、會員名稱、會員手機號碼、成立時間、編輯按鈕  """  @app.route('/manage\_orders', methods=['GET'])  def manage\_orders():  conn = create\_connection()  cursor = conn.cursor()  # 取得訂單資料  cursor.execute('''  SELECT orders.id, users.username, users.phone, orders.timestamp  FROM orders  INNER JOIN users ON orders.user\_id = users.id  ''')  orders = cursor.fetchall()  conn.close()  return render\_template('admin/manage\_orders.html', orders=orders) |