|  |  |  |  |
| --- | --- | --- | --- |
| **Subject :** | QPlay Backend Platform | **Latest Ver :** | 1.4.1 |
| **Latest Release Date :** | 2018-07-26 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Change History** | | | | |
| Date | Author | Version | Type | Comment |
| 2018-05-23 | Darren.K.Ti | 1.0 | Function  DB | 功能修改記錄 : 4.1  1. QPlay 後台新增功能 [ 公司管理 ]  2. QPlay Login 流程修改  3. DB `qplay` 新增 Table  4. QPlay Backend Login - Remember Me |
| 2018-06-19 | Darren.K.Ti | 1.1 | Function  DB | 功能修改記錄 : 4.2  1. QPlay 後台新增功能 [ 消費券管理 ]  2. DB `qplay` 新增 QPay 相關 Table  3. QPlay API 新增   1. QPay 相關 API 2. 更新 QAccount Password   4. APP QPlay   1. 整合 QPay 2. 新增功能 [ 更新 QAccount Password ]   5. QPay Web 查詢頁面 |
| 2018-06-25 | Darren.K.Ti | 1.2 | Function DB  API | 功能修改記錄 : 4.3  1. QPlay 後台新增功能 [ Function Management ]  2. DB `qplay` 新增 Function Manage 相關 Table  3. QPlay API 新增 Function Manage 相關 API |
| 2018-07-06 | Darren.K.Ti | 1.3 | Function  API | 功能修改記錄 : 4.4  1. Plugin QLogin 頁面 支援 QAccount  2. QPlay API login 支援 QAccount  3. QPlay API register 支援 QAccount  3. APP QPlay 使用 Function Manage |
| 2018-07-16 | Darren.K.Ti | 1.3.1 | Function  DB  API | 功能修改記錄 : 4.3  1. Function Detail 的權限管理, 要處理 company\_label  2. 新增設定 QAccount 的權限等級 |
| 2018-07-20 | Darren.K.Ti | 1.4 | Function  DB  API | 功能修改記錄 : 4.2  1. 完成 2018-06-25 尚未完成的項目 |
| 2018-07-26 | Darren.K.Ti | 1.4.1 | Function  API | 功能修改記錄 : 4.2  1. QPay Web API : 修正執行流程  2. 依據功能，個別評估時程 |

# 目錄

[**目錄**](#_30j0zll) **3**

[**1.0 系統設計 (System Design)**](#_3znysh7) **4**

[**2.0 資料庫設計 (Database Design)**](#_tyjcwt) **5**

[**3.0 Implementation**](#_3dy6vkm) **5**

[**4.0 功能修改記錄**](#_1t3h5sf) **6**

[4.2 [ QPlay 後台新增功能 [ 消費券管理 ]，DB `qplay` 新增 QPay 相關 Table，QPlay API 新增 QPay 相關 API 以及 更新 QAccount Password，APP QPlay 整合 QPay 以及新增功能 [ 更新 QAccount Password ]，QPay Web 查詢頁面 ]](#_4d34og8) 6

# 

# 1.0 系統設計 (System Design)

**1.1 系統架構圖 (System Framework Diagram)**

* QPlay Backend Platform 整體的系統架構圖，待補上

**1.2 系統流程圖 (System Flow)**

* QChat API 基於 laravel 5.2 框架進行開發，並使用 Repository + Service 模式之程式架構。
* 使用 index.php 作為進入點，route 為路由分配，midware 作為中介層，進行 request 和 response 的驗證，以及寫入 API Log。



**1.3 類別圖 (Package / Class Diagram)**

* QPlay Backend Platform 整體的類別圖，待補上

# 2.0 資料庫設計 (Database Design)

請參考資料庫設計文件 : QPlay\_DB\_Schema.docx

2.1 資料庫物件清單 (Database Object List)

* 每個修改項目 4.X 各自描述

2.2 實體關聯圖 (Entity-Relation Chart)

* 每個修改項目 4.X 各自描述

2.3 資料表結構 (Table Layout)

* 每個修改項目 4.X 各自描述

# 3.0 Implementation

3.1 公用元件 (Public Component)

* 每個修改項目 4.X 各自描述

3.2 共用介面 (Public Interface)

* 每個修改項目 4.X 各自描述

# 4.0 功能修改記錄

## 4.2 [ QPlay 後台新增功能 [ 消費券管理 ]，DB `qplay` 新增 QPay 相關 Table，QPlay API 新增 QPay 相關 API 以及 更新 QAccount Password，APP QPlay 整合 QPay 以及新增功能 [ 更新 QAccount Password ]，QPay Web 查詢頁面 ]

4.2.1 需求說明

* **QPlay 後台新增功能 [ 消費券管理 ]**，依照 UIFlow，產生以下階層功能 :
  + 用戶管理
    - 管理員資料
      * 管理員的新增 / 移除。
    - 消費券類型
      * 消費券類型的新增 / 移除。
      * 顏色代碼設定。
    - 店家資料
      * 店家的新增 / 移除。
      * 密碼重置。
      * 使用狀態管理。
      * 交易狀態管理。
    - 員工資料
      * 有領取消費券的員工列表。
      * 交易密碼重置。
  + 儲值管理
    - 點數儲值
      * 使用 Excel 匯入員工名單，進行消費券發放。
      * 產生儲值代碼。
    - 儲值紀錄
      * 儲值紀錄列表。
      * 可以下載當初上傳的 Excel 檔案。
    - 員工發放紀錄
      * 有領取消費券的員工，查詢其點數的儲值紀錄。
  + 報銷管理
    - 財務請款
      * 依據 消費券類型 + 店家 + 月份，產生當月份的交易紀錄。
      * 承上，將資料以日期做結算，匯出為 Excel 檔案。
    - 員工消費記錄
      * 查詢指定員工的消費記錄。
  + 報表分析
    - 依據需求，產生圖表 (功能未定，Phase 1 不實作)。
  + 備註
    - 由於目前 QPlay 後台的選單階層，只有兩階層，但是，如果 QPay 要加入，必須要有三階層。
    - 可能的處理方式 :
      * 修改 QPlay 後台的選單階層，程式碼跟 DB Table 都需要同步配合修改。
      * 針對 QPay，將第三階層的選單內容寫在程式碼中，另外透過 Javascript 去產生第三階層的選單。
* **DB `qplay` 新增 QPay 相關 Table**
* **QPlay API 新增**
  + **QPay 相關 API** 
    - **API getQPayInfo** ( 取得基本資料 )
      * Post : UUID / emp\_type。
      * Return : 點數總額 / 店家列表 / 消費券類型。
    - **API newTrade** ( 交易處理 )
      * Post : UUID / shop\_id / price / trade\_pwd。
      * Return : 交易結果/ 消費券餘額 / 交易代碼 / 交易時間。
    - **API getStoreRecord** ( 儲值紀錄 )
      * Post : UUID / start\_date / end\_date
      * Return : 儲值紀錄。
    - **API getTradeRecord** ( 交易紀錄 for 員工 or 店家 )
      * Post : UUID / start\_date / end\_date / emp\_type / point\_type\_id。
      * Return : 交易紀錄。
    - **API checkTradeRecord** ( 當天交易紀錄 for 店家)
      * Post : UUID。
      * Return : 當天交易紀錄。
    - **API resetTradePwdForAPP** ( 更新交易密碼 for APP )
      * Post : UUID / old\_trade\_pwd / new\_trade\_pwd。
      * Return : 結果。
    - **API resetTradePwdForWeb** ( 更新交易密碼 for Web )
      * Post : emp\_no / old\_trade\_pwd / new\_trade\_pwd。
      * Return : 結果。
    - **API loginQPayWeb** ( 登入 QPay Web )
      * Post : emp\_no / trade\_pwd。
      * Return : 結果。
    - **API logoutQPayWeb** ( 登出 QPay Web )
      * Post : emp\_no。
      * Return : 結果。
    - **API tradeRecordQPayWeb** ( 交易紀錄 和 儲值紀錄 for QPay Web )
      * Post : emp\_no / start\_date / end\_date。
      * Return : 結果。
  + **更新 QAccount Password**
    - User 的 company = shop，可以進行 QAccount Password 的更新。
    - **API resetQAccountPassword** ( 更新 QAccount 登入密碼 )
      * Post : login\_id(UUID) / old\_qaccount\_pwd / new\_qaccount\_pwd。
      * Return : 結果。
* **APP QPlay**
  + **整合 QPay**
    - 餘額查詢
    - 店家列表
    - 進行交易
    - 交易紀錄
    - 更新交易密碼
  + **新增功能 [ 更新 QAccount Password ]**
    - 依照討論，目前只有 company = shop 的 User 可以修改 QAccount Password。
    - QPlay APP 可以依據 API getFunctionList 的資料，判斷當前 User 是否有權限使用此功能。
  + **這份 PESPIS 文件不包含 APP QPlay 程式碼修改的規劃。**
* **QPay Web 查詢頁面**
  + 使用工號 / 交易密碼登入。
  + 查詢儲值紀錄 / 交易紀錄。
  + 更新交易密碼。
  + **這份 PESPIS 文件不包含 QPay Web 查詢頁面 程式碼修改的規劃。**

4.2.2 時程評估 (依照開發順序)

* [ QPlay 後台- 消費券管理 - 用戶管理 - 管理員資料 ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 用戶管理 - 消費券類型 ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 用戶管理 - 店家資料 ]
  + 預估時程 : ? ~ ? (2.5 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 儲值管理 - 點數儲值 UIFlow 1 ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 儲值管理 - 點數儲值 UIFlow 2 ]
  + 預估時程 : ? ~ ? (1 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 儲值管理 - 點數儲值 UIFlow 3 ]
  + 預估時程 : ? ~ ? (4 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 儲值管理 - 點數儲值 UIFlow 4 ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 儲值管理 - 儲值紀錄 ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 用戶管理 - 員工資料 ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 用戶管理 - 員工資料 - 重置員工的交易密碼 ]
  + 預估時程 : ? ~ ? (1 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 儲值管理 - 員工發放紀錄 ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 報銷管理 - 財務請款 ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 報銷管理 - 財務請款 - 匯出 Excel ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 報銷管理 - 員工消費記錄 UIFlow 1 ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ QPlay 後台- 消費券管理 - 報銷管理 - 員工消費記錄 UIFlow 2 ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ API - getQPayInfo ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ API - getStoreRecord ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ API - newTrade - UIFlow 1 ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ API - newTrade - UIFlow 2 ]
  + 預估時程 : ? ~ ? (2.5 day)
  + 實際時程 :
* [ API - newTrade - UIFlow 3 ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ API - newTrade - UIFlow 4 ]
  + 預估時程 : ? ~ ? (0.5 day)
  + 實際時程 :
* [ API - getTradeRecord ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ API - checkTradeRecord ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ API - resetTradePwdForAPP ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ API - loginQPayWeb ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ API - logoutQPayWeb ]
  + 預估時程 : ? ~ ? (1 day)
  + 實際時程 :
* [ API - tradeRecordQPayWeb ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ API - resetTradePwdForWeb ]
  + 預估時程 : ? ~ ? (1 day)
  + 實際時程 :
* [ API - resetQAccountPassword ]
  + 預估時程 : ? ~ ? (1 day)
  + 實際時程 :

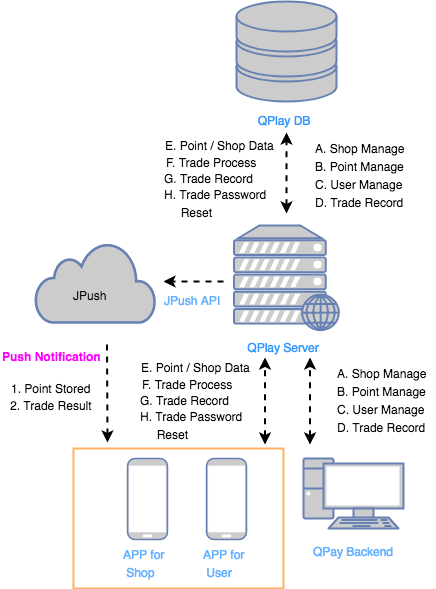
4.2.3 對應文件

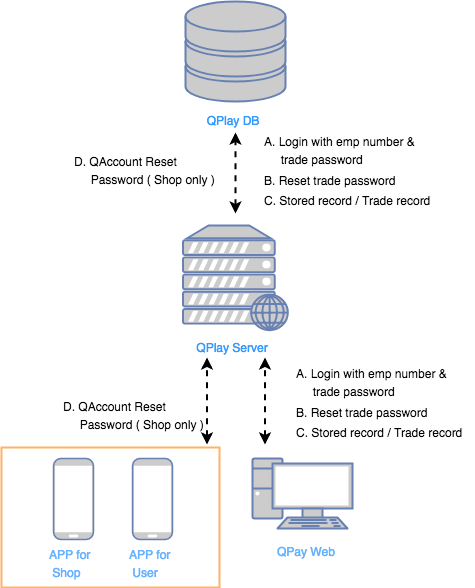
* QPlay\_API\_1.3.6\_20180726.docx
* QPlay\_DB\_Schema\_3.8\_20180720.docx

4.2.4 修改項目

* DB `**qplay**`
  + 新增 Table
    - `**qpay\_manager**`
    - `**qpay\_shop**`
    - `**qpay\_point\_shop**`
    - `**qpay\_point\_store**`
    - `**qpay\_member**`
    - `**qpay\_member\_point**`
    - `**qpay\_point\_trade\_log**`
* qplayApi/ qplayApi/
  + 修改
    - app/ Http/ **routes.php**
    - app/ Http/ Controllers/ **qplayController.php**
    - app/ Repositories/ **UserRepository.php**
    - app/ lib/
      1. **ResultCode.php**
      2. **Verify.php**
  + 新增
    - app/ Model/
      1. **QPay\_Shop.php**
      2. **QPay\_Point\_Type.php**
      3. **QPay\_Point\_Store.php**
      4. **QPay\_Member.php**
      5. **QPay\_Member\_Point.php**
      6. **QPay\_Point\_Trade\_Log.php**
    - app/ Repositories/
      1. **QPayShopRepository.php**
      2. **QPayPointTypeRepository.php**
      3. **QPayMemberRepository.php**
      4. **QPayMemberPointRepository.php**
      5. **QPayPointTradeLogRepository.php**
    - app/ Services/ **qpayService.php**
* qplay/
  + 修改
    - app/ Http/ **routes.php**
    - app/ lib/ **ResultCode.php**
    - app/ Model/ **QP\_User.php**
    - resources/ lang
      1. en/ **messages.php**
      2. zh-cn/ **messages.php**
      3. zh-tw/ **messages.php**
  + 新增
    - app/ Model/
      1. **QPay\_Manager.php**
      2. **QPay\_Shop.php**
      3. **QPay\_Point\_Type.php**
      4. **QPay\_Point\_Store.php**
      5. **QPay\_Member.php**
      6. **QPay\_Member\_Point.php**
      7. **QPay\_Point\_Trade\_Log.php**
    - app/ Repositories/
      1. **QPayManagerRepository.php**
      2. **QPayShopRepository.php**
      3. **QPayPointTypeRepository.php**
      4. **QPayPointStoreRepository.php**
      5. **QPayMemberRepository.php**
      6. **QPayMemberPointRepository.php**
      7. **QPayPointTradeLogRepository.php**
    - app/ Http/ Controllers/ **qpayController.php**
    - app/ Services/ **qpayService.php**
    - resources/ views/ **qpay\_maintain**/
      1. **manager.blade.php**
      2. **pointType.blade.php**
      3. **shop.blade.php**
      4. **member.blade.php**
      5. **pointStore.blade.php**
      6. **pointStoreRecord.blade.php**
      7. **pointGetRecord.blade.php**
      8. **reimburseFinance.blade.php**
      9. **reimbursePurchase.blade.php**
* QPlay Backend > APP 管理 > QPlay > 錯誤代碼 **Error.json**

4.2.5 功能架構圖

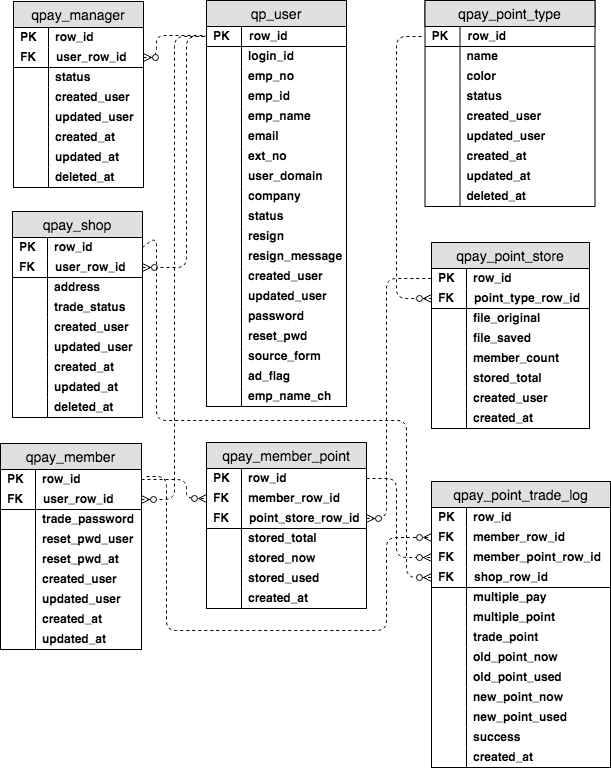




4.2.6 資料庫物件清單

|  |  |  |  |
| --- | --- | --- | --- |
| **DB Type** | **Type** | **Object Name** | **Description** |
| MySQL | Table | qp\_user | 取得員工的基本資料 ( 身分證後 6 碼 / QAccount Password / Status / Resgin )。 |
| MySQL | Table | qpay\_manager | 消費券管理員的資料。 |
| MySQL | Table | qpay\_shop | 店家資料 ( 地址 / 交易狀態 )，其他資料比照員工，紀錄於 `qp\_user`。 |
| MySQL | Table | qpay\_point\_type | 消費券類型的資料 ( 名稱 / 顏色代碼 )。 |
| MySQL | Table | qpay\_point\_store | 消費券的儲值紀錄 ( 總人數 / 總點數 / 上傳的 Excel 檔案名稱 )。 |
| MySQL | Table | qpay\_member | 領取到消費券點數的員工資料 ( 交易密碼 )，其他基本資料紀錄於 `qp\_user`。 |
| MySQL | Table | qpay\_member\_point | 員工領取的消費券點數資料 :   * 從哪一位管理員上傳的哪一份 Excel 獲得多少點數。 * 以及這份點數剩餘多少，使用多少。 |
| MySQL | Table | qpay\_point\_trade\_log | 消費券點數的交易紀錄 :   * 此筆交易是否跨越兩筆不同的消費券點數。 * 交易金額。 * 交易之前的點數紀錄。 * 交易之後的點數紀錄。 * 交易是否成功。 |

4.2.7 實體關聯圖



4.2.8 資料表結構

* 參考文件 : QPlay\_DB\_Schema\_3.8\_20180720.docx

4.2.9 公用元件

* 內部模組

4.2.10 共用介面

* QPlay\_API\_1.3.6\_20180726.docx

4.2.11 資料流程圖 Data Flow Diagram

* 消費券管理 > 用戶管理 > 管理者資料 - 新增管理員



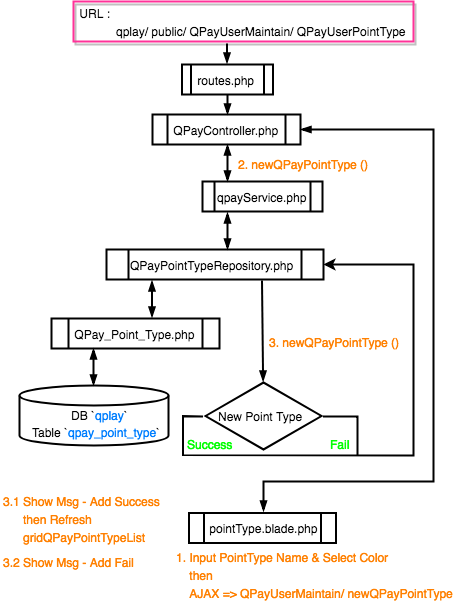
* 消費券管理 > 用戶管理 > 管理者資料 - 管理員列表



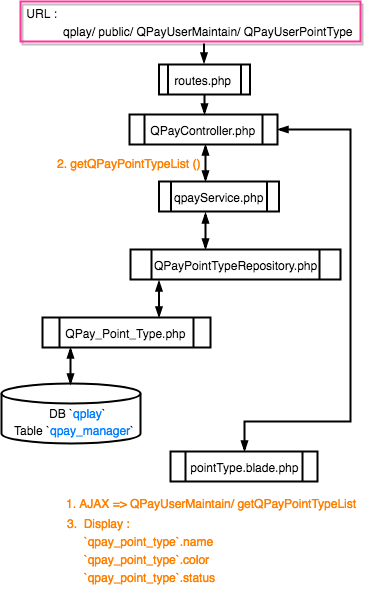
* 消費券管理 > 用戶管理 > 管理者資料 - 編輯管理員



* 消費券管理 > 用戶管理 > 消費券資料 - 新增消費券



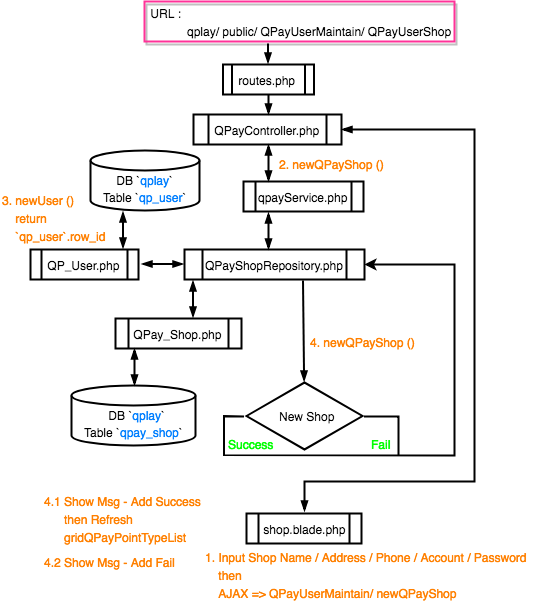
* 消費券管理 > 用戶管理 > 消費券資料 - 消費券列表



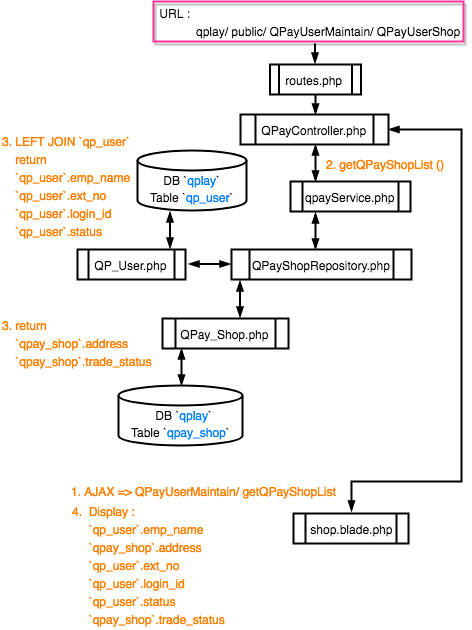
* 消費券管理 > 用戶管理 > 消費券資料 - 編輯消費券



* 消費券管理 > 用戶管理 > 店家資料 - 新增店家



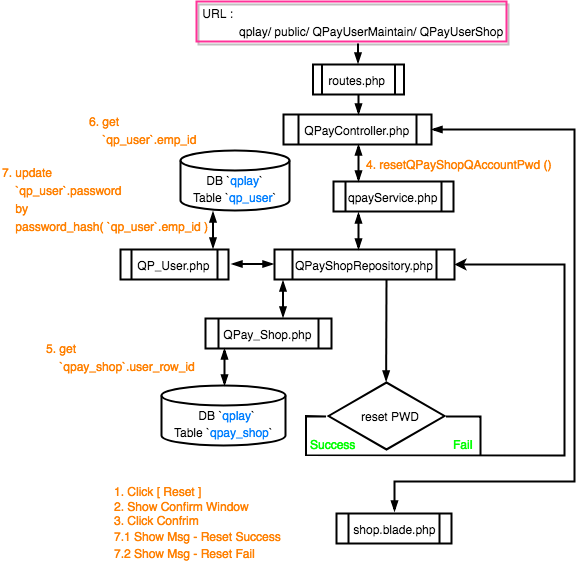
* 消費券管理 > 用戶管理 > 店家資料 - 店家列表



* 消費券管理 > 用戶管理 > 店家資料 - 編輯店家



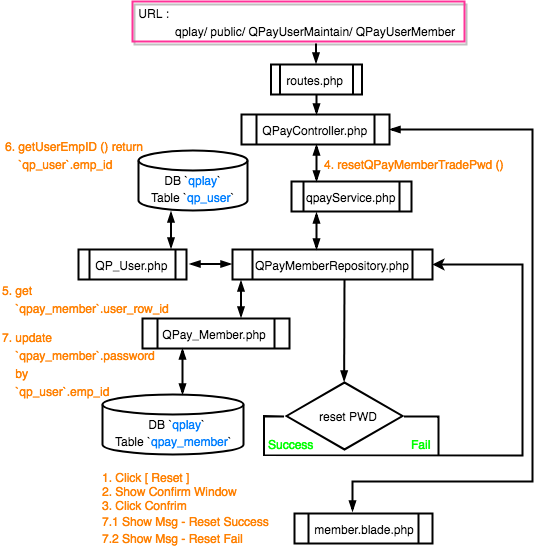
* 消費券管理 > 用戶管理 > 店家資料 - 重置店家的 QAccount 登入密碼



* 消費券管理 > 用戶管理 > 員工資料 - 員工列表



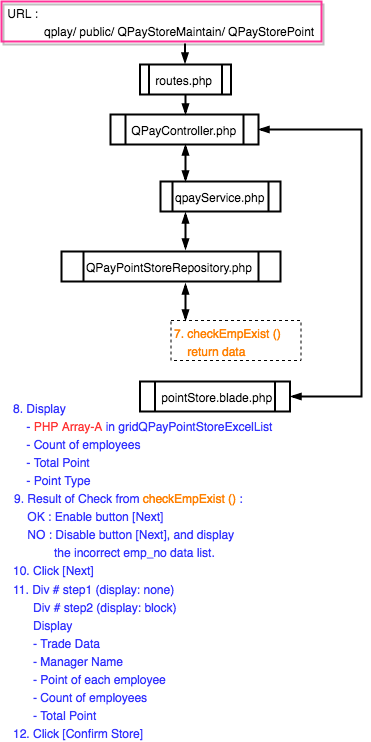
* 消費券管理 > 用戶管理 > 員工資料 - 重置員工的交易密碼



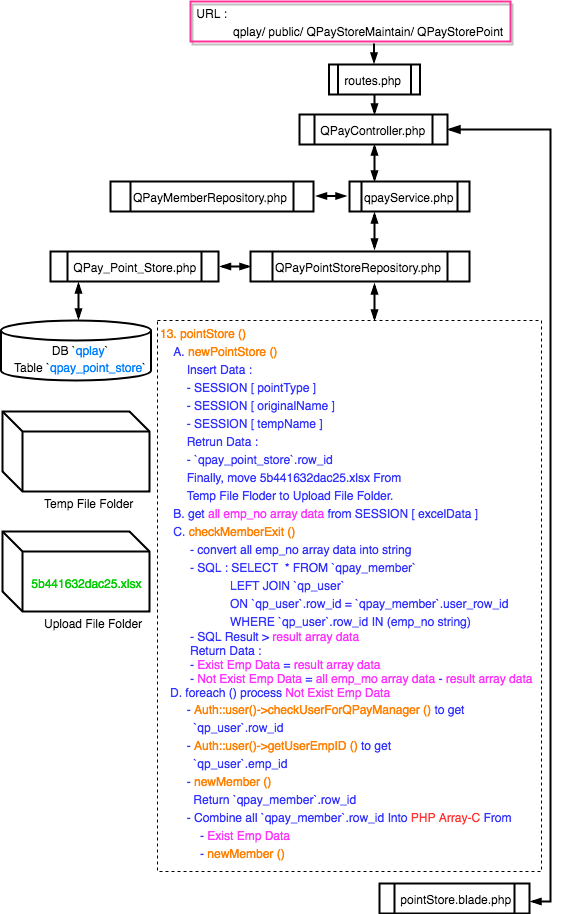
* 消費券管理 > 儲值管理 > 點數儲值 - UIFlow 1



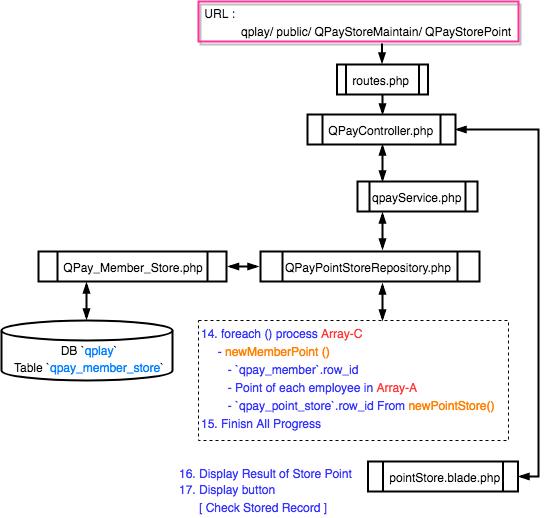
* 消費券管理 > 儲值管理 > 點數儲值 - UIFlow 2



* 消費券管理 > 儲值管理 > 點數儲值 - UIFlow 3



* 消費券管理 > 儲值管理 > 點數儲值 - UIFlow 4



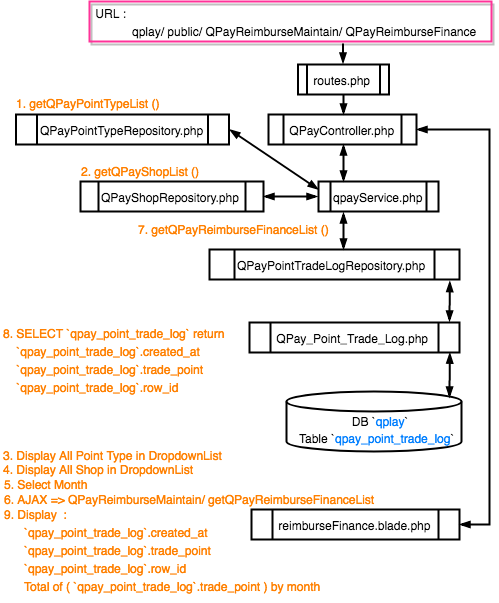
* 消費券管理 > 儲值管理 > 儲值紀錄



* 消費券管理 > 儲值管理 > 員工發放紀錄



* 消費券管理 > 報銷管理 > 財務請款



* 消費券管理 > 報銷管理 > 財務請款 - 匯出 Excel



* 消費券管理 > 報銷管理 > 員工消費記錄 - UIFLow 1



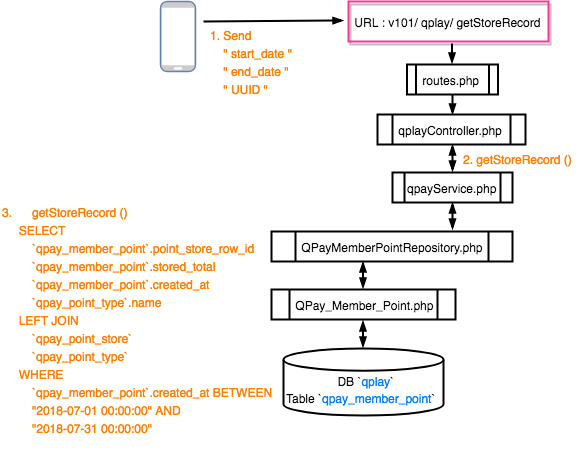
* 消費券管理 > 報銷管理 > 員工消費記錄 - UIFLow 2



* API - getQPayInfo



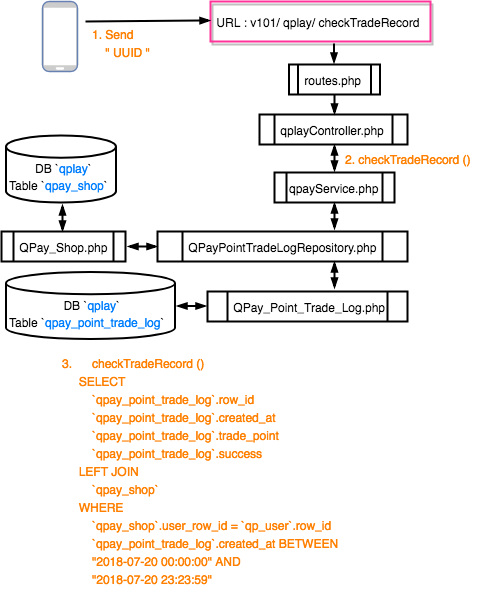
* API getStoreRecord



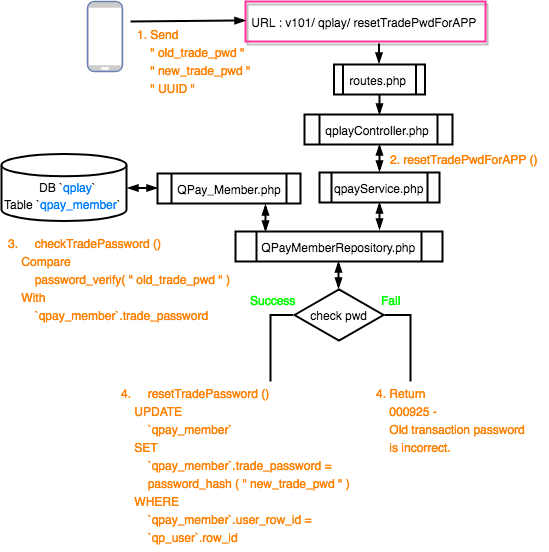
* API getTradeRecord



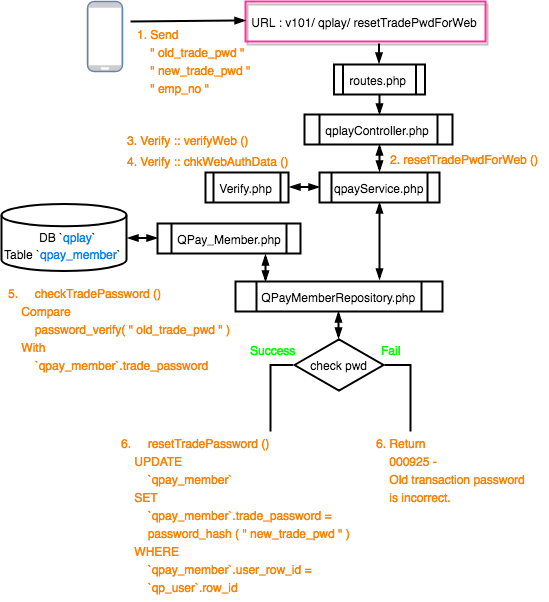
* API checkTradeRecord



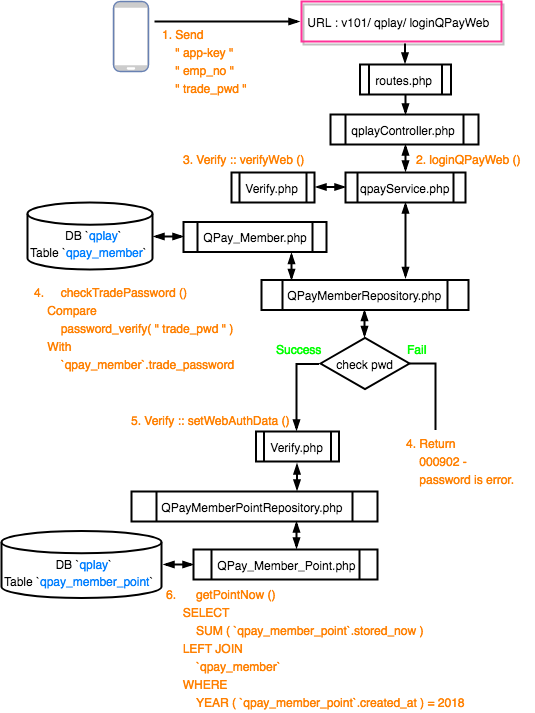
* API resetTradePwdForAPP



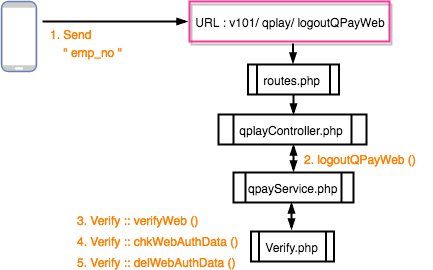
* API resetTradePwdForWeb



* API loginQPayWeb



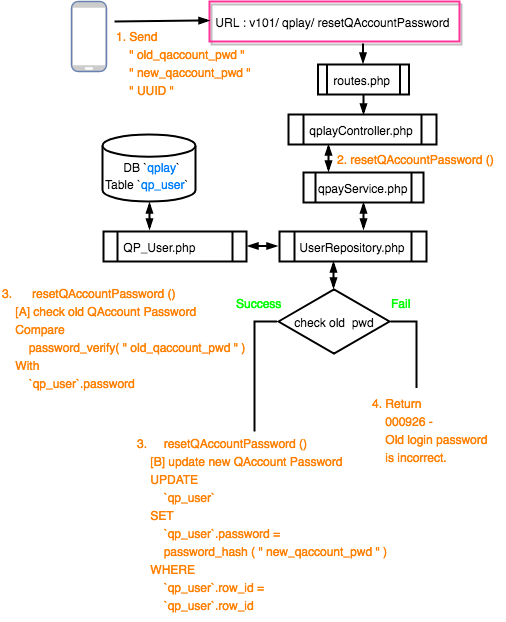
* API logoutQPayWeb



* API tradeRecordQPayWeb



* API resetQAccountPassword



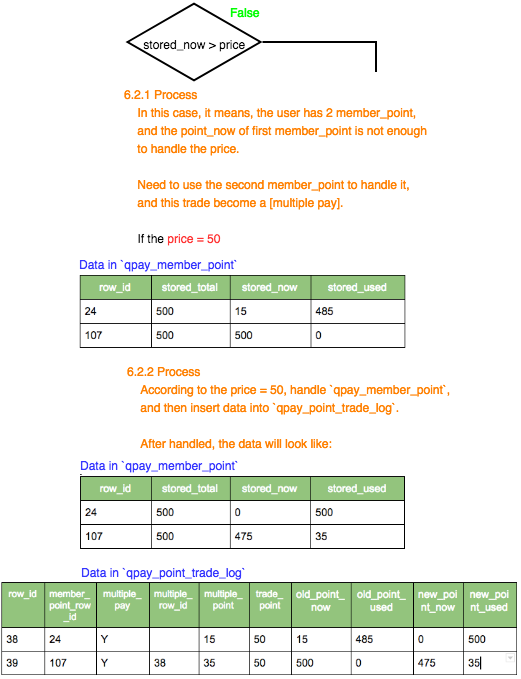
* API newTrade - UIFlow 1



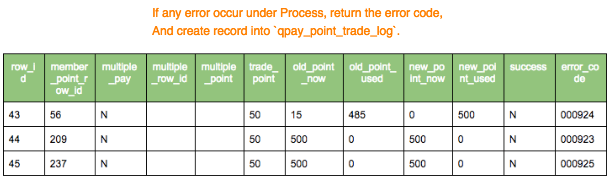
* API newTrade - UIFlow 2



* API newTrade - UIFlow 3



* API newTrade - UIFlow 4



4.2.12 程式碼修改記錄

* qpalyApi/ qplayApi/
  + app/ Http/ **routes.php**
    - URL Request Handle :
      * 新增
        + getQPayInfo，取得 QPay 基本資料 ( 點數總額 / 店家列表 / 消費券型態 )。
        + newTrade，進行交易處理。
        + getStoreRecord，查詢儲值紀錄。
        + getTradeRecord，查詢交易紀錄。
        + checkTradeRecord，查詢店家的當天交易紀錄。
        + resetTradePwdForAPP，更新交易密碼 For APP。
        + resetTradePwdForWeb，更新交易密碼 For Web。
        + loginQPayWeb，登入 QPay Web。
        + logoutQPayWeb，登出 QPay Web。
        + tradeRecordQPayWeb，查詢交易紀錄 For QPay Web。
        + resetQAccountPassword，更新 QAccount 登入密碼。
  + app/ Http/ Controllers/ **qplayController.php**
    - 定義 function
      * getQPayInfo ()。
      * newTrade ()。
      * getStoreRecord ()。
      * getTradeRecord ()。
      * checkTradeRecord ()。
      * resetTradePwdForAPP ()。
      * resetTradePwdForWeb ()。
      * loginQPayWeb ()。
      * logoutQPayWeb ()。
      * tradeRecordQPayWeb ()。
      * resetQAccountPassword ()。
  + app/ Model/
    - **QPay\_Point\_Type.php**
      * 定義 DB `qplay` Table `qpay\_point\_type`。
    - **QPay\_Point\_Store.php**
      * 定義 DB `qplay` Table `qpay\_point\_store`。
    - **QPay\_Shop.php**
      * 定義 DB `qplay` Table `qpay\_shop`。
    - **QPay\_Member.php**
      * 定義 DB `qplay` Table `qpay\_member`。
    - **QPay\_Member\_Point.php**
      * 定義 DB `qplay` Table `qpay\_member\_point`。
    - **QPay\_Point\_Trade\_Log.php**
      * 定義 DB `qplay` Table `qpay\_point\_trade\_log`。
  + app/ Repositories/
    - **QPayShopRepository.php**
      * 定義 function getEnableShopList ()
        + 取得消費券店家的資料
        + 條件

`qp\_user`.status = Y

`qpay\_shop`.trade\_status = Y

* + - * + 回傳

`qpay\_shop`.row\_id

`qp\_user`.emp\_name as shop\_name

* + - * 定義 function checkShopStatus ()
        + 檢查店家是否可以正常進行交易
        + 帶入

`qpay\_shop`.row\_id

* + - * + 條件

`qp\_user`.status = Y

`qpay\_shop`.trade\_status = Y

* + - * + 回傳

狀態正常

狀態異常

* + - **QPayPointTypeRepository.php**
      * 定義 function getEnablePointTypeList ()
        + 取得消費券類型的資料
        + 條件

`qpay\_point\_type`.status = Y

* + - * + 回傳

`qpay\_point\_type`.row\_id

`qpay\_point\_type`.name

* + - **QPayMemberRepository.php**
      * 定義 function checkTradePassword ()
        + 驗證交易密碼是否正確
        + 帶入

`qpay\_member`.trade\_password

* + - * + 回傳

密碼正確

密碼錯誤

* + - * + 備註 :

`qpay\_member`.trade\_password 使用 PHP password\_verify () 處理之後進行驗證

* + - * 定義 function resetTradePassword ()
        + 更新交易密碼
        + 帶入

`qp\_user`.row\_id

new\_trade\_pwd

* + - * + 回傳

更新成功

更新失敗

* + - * + 備註 :

使用 PHP password\_hash () 進行加密

algorithms : PASSWORD\_BCRYPT

$options [ 'cost' => 10 ]

存入`qpay\_member`.trade\_password

* + - * 定義 function loginQPayWeb ()
        + 登入 QPay Web
        + 帶入

`qp\_user`.emp\_no

`qpay\_member`.trade\_password

* + - * + 回傳

登入成功

登入失敗

* + - **QPayMemberPointRepository.php**
      * 定義 function getPointNow ()
        + 取得今年度可用的點數餘額
        + 帶入

`qpay\_member`.row\_id

* + - * + 條件

`qpay\_member\_point`.created\_at 時間範圍在今年度

* + - * + 回傳

SUM ( `qpay\_member\_point`.stored\_now )

* + - * 定義 function processPoint ()
        + 進行交易處理
        + 帶入

`qpay\_member`.row\_id

price

* + - * + 回傳

交易成功

`qp\_user`.emp\_no

`qpay\_point\_trade\_log`.row\_id

`qpay\_point\_trade\_log`.trade\_point

`qpay\_point\_trade\_log`.created\_at

SUM ( `qpay\_member`.stored\_now )

交易失敗

系統錯誤代碼

* + - * 定義 fucntion getStoreRecord ()
        + 取得儲值紀錄
        + 帶入

emp\_no(UUID)

start\_date

end\_date

* + - * + 回傳

`qpay\_member\_point`.point\_store\_row\_id

`qpay\_member\_point`.stored\_total

`qpay\_member\_point`.created\_at

`qpay\_point\_type`.name

* + - * + 備註 :

儲值碼

使用 `qpay\_member\_point`.point\_store\_row\_id 組成 6 位數的數字，左邊自動補 0

前方加上字串 [ S ]

ex :

`qpay\_member\_point`.point\_store\_row\_id = 1

儲值碼 : S000001

* + - **QPayPointTradeLogRepository.php**
      * 定義 function getTradeRecord ()
        + 取得交易紀錄
        + 帶入

emp\_no(UUID)

start\_date

end\_date

emp\_type

point\_type\_id (emp\_type = shop 時帶入此參數)

* + - * + 回傳 :

emp\_type = emp

`qpay\_point\_trade\_log`.row\_id

`qp\_user`.emp\_name as shop\_name

`qpay\_point\_trade\_log`.created\_at

`qpay\_point\_trade\_log`.trade\_point

`qpay\_point\_trade\_log`.success

emp\_type = shop

只撈取指定 point\_type\_id (消費券類型的 row\_id)

`qpay\_point\_type`.name

SUM ( `qpay\_point\_trade\_log`.trade\_point )

`qpay\_point\_trade\_log`.row\_id

`qpay\_point\_trade\_log`.created\_at

`qpay\_point\_trade\_log`.trade\_point

`qpay\_point\_trade\_log`.success

* + - * + 備註 :

交易碼

使用 `qpay\_point\_trade\_log`.row\_id 組成 6 位數的數字，左邊自動補 0

前方加上字串 [ T ]

ex :

`qpay\_point\_trade\_log`.row\_id = 1

交易碼 : T000001

* + - * 定義 function checkTradeRecord ()
        + 店家取得當天的所有交易紀錄
        + 帶入

emp\_no(UUID)

* + - * + 回傳 :

`qpay\_point\_trade\_log`.row\_id

`qpay\_point\_trade\_log`.created\_at

`qpay\_point\_trade\_log`.trade\_point

`qpay\_point\_trade\_log`.success

* + - * + 備註 :

交易碼

使用 `qpay\_point\_trade\_log`.row\_id 組成 6 位數的數字，左邊自動補 0

前方加上字串 [ T ]

ex :

`qpay\_point\_trade\_log`.row\_id = 1

交易碼 : T000001

* + - * 定義 function newTradeRecord ()
        + 新增交易紀錄
        + 帶入

`qpay\_member`.row\_id

`qpay\_member\_point`.row\_id

`qpay\_member\_point`.store\_now (舊)

`qpay\_member\_point`.store\_used (舊)

`qpay\_member\_point`.store\_now (新)

`qpay\_member\_point`.store\_used (新)

`qpay\_shop`.row\_id

multiple\_pay = Y or N

multiple\_point

price

success = Y or N

* + - * + 回傳

`qpay\_point\_trade\_log`.row\_id

* + - * + 備註 :

交易碼

使用 `qpay\_point\_trade\_log`.row\_id 組成 6 位數的數字，左邊自動補 0

前方加上字串 [ T ]

ex :

`qpay\_point\_trade\_log`.row\_id = 1

交易碼 : T000001

* + - **UserRepository.php**
      * 定義 function resetQAccountPassword ()
        + 更新 QAccount 登入密碼
        + 帶入

login\_id(UUID)

old\_qaccount\_pwd

new\_qaccount\_pwd

* + - * + 回傳

更新成功

更新失敗

* + - * + 備註 :

執行流程 :

先檢查 old\_qaccount\_pwd 是否正確，

使用 PHP password\_verify () 處理之後進行驗證

接著，使用 PHP password\_hash () 進行加密

algorithms : PASSWORD\_BCRYPT

$options [ 'cost' => 10 ]

存入`qp\_user`.password

* + app/ Services/ **qpayService.php**
    - 定義 function
      * getQPayInfo ()
        + 帶入

emp\_type

UUID

* + - * + 依據 API 帶入的 emp\_type :

emp

使用QPayShopRepository.php 定義的

getEnableShopList ()。

使用QPayMemberPointRepository.php 定義的

getPointNow ()。

shop

使用QPayPointTypeRepository.php 定義的

getEnablePointTypeList ()。

* + - * + 回傳

emp

店家列表

點數總額

shop

消費券類型列表

* + - * newTrade ()
        + 帶入

UUID

shop\_id

price

trade\_pwd

* + - * + 使用QPayMemberRepository.php 定義的 checkTradePassword ()。
        + 使用 QPayShopRepository.php 定義的 checkShopStatus ()。
        + 使用QPayMemberPointRepository.php 定義的 getPointNow ()。
        + 使用QPayMemberPointRepository.php 定義的 processPoint ()。
        + 使用QPayPointTradeLogRepository.php 定義的 newTradeRecord ()。
        + 備註 :

處理流程 :

checkTradePassword () 檢查成功，往下繼續處理

checkShopStatus () 檢查成功，往下繼續處理

getPointNow () 的結果 跟 price 比較

getPointNow () < price

餘額不足

getPointNow () > price

往下繼續處理

processPoint ()

計算點數餘額

交易處理的結果傳送至 newTradeRecord ()

newTradeRecord ()

紀錄交易結果，產生交易代碼

* + - * + 回傳

交易結果

* + - * getStoreRecord ()
        + 帶入

UUID

start\_date

end\_date

* + - * + 使用QPayMemberPointRepository.php 定義的 getStoreRecord ()。
        + 回傳

儲值紀錄

* + - * getTradeRecord ()
        + 帶入

UUID

start\_date

end\_date

emp\_type

point\_type\_id (emp\_type = shop 時帶入此參數)

* + - * + 使用QPayPointTradeLogRepository.php 定義的 getTradeRecord ()。
        + 回傳

交易紀錄

* + - * checkTradeRecord ()
        + 帶入

UUID

* + - * + 使用QPayPointTradeLogRepository.php 定義的 checkTradeRecord ()。
        + 回傳

店家在當天的所有交易紀錄

* + - * resetTradePwdForAPP ()
        + 帶入

UUID

old\_trade\_pwd

new\_trade\_pwd

* + - * + 使用QPayMemberRepository.php 定義的 checkTradePassword ()。
        + 使用QPayMemberRepository.php 定義的 resetTradePassword ()。
        + 回傳

更新結果

* + - * resetTradePwdForWeb ()
        + 帶入

emp\_no

old\_trade\_pwd

new\_trade\_pwd

* + - * + 使用 Verify.php 定義的 verifyWeb () 檢查。
        + 使用 Verify.php 定義的 chkWebAuthData () 檢查。
        + 使用QPayMemberRepository.php 定義的 checkTradePassword ()。
        + 使用QPayMemberRepository.php 定義的 resetTradePassword ()。
        + 回傳

更新結果

* + - * loginQPayWeb ()
        + 帶入

emp\_no

trade\_pwd

* + - * + 使用 Verify.php 定義的 verifyWeb () 檢查。
        + 使用QPayMemberRepository.php 定義的 loginQPayWeb ()。

登入成功

使用 Verify.php 定義的 setWebAuthData () 產生資料並且回傳。

使用QPayMemberPointRepository.php 定義的

getPointNow ()。

回傳

點數總額

登入失敗

回傳

登入失敗

* + - * logoutQPayWeb ()
        + 帶入

emp\_no

* + - * + 使用 Verify.php 定義的 verifyWeb () 檢查。
        + 使用 Verify.php 定義的 chkWebAuthData () 檢查。
        + 使用 Verify.php 定義的 delWebAuthData () 清除 SESSION。
        + 回傳

登出成功

登出失敗

* + - * tradeRecordQPayWeb ()
        + 帶入

emp\_no

start\_date

end\_date

* + - * + 使用 Verify.php 定義的 verifyWeb () 檢查。
        + 使用 Verify.php 定義的 chkWebAuthData () 檢查。
        + 使用QPayMemberPointRepository.php 定義的 getStoreRecord ()。
        + 使用 QPayPointTradeLogRepository.php 定義的 getTradeRecord()。
        + 備註 :

處理流程 :

依序取得 getStoreRecord () 和 getTradeRecord() 回傳的資料，並且合併 :

每筆資料註明 record\_type = store or trade

使用 created\_at (timestamp) 做排序

* + - * + 回傳交易/儲值紀錄
      * resetQAccountPassword ()
        + 帶入

login\_id(UUID)

old\_qaccount\_pwd

new\_qaccount\_pwd

* + - * + 使用UserRepository.php 定義的 resetQAccountPassword ()。
        + 回傳

更新結果

* + app/ lib/ **ResultCode.php**
    - 新增
      * \_000923\_shopTradeStatusDisabled
      * \_000924\_tradePointInsufficientqplay
      * \_000925\_oldTradePasswordIncorrect
      * \_000926\_oldLoginPasswordIncorrect
  + app/ lib/ **Verify.php**
    - 新增
      * function verifyWeb ()
        + 使用 Web 呼叫 API，因為不具有 UUID，必須有額外的驗證流程
        + 驗證

Content-Type

App-Key

* + - * + 備註

參考 function verify () 的驗證流程，檢查這兩個資料

* + - * function setWebAuthData ()
        + Web 透過 API 進行 User 身份認證，完成檢查後，產生驗證資料，回傳給 Web
        + 產生

SignatureTime

Signature

token

tokenValid

* + - * + 執行流程

SignatureTime : 使用 time() 產生當下的 timestamp

Signature : 參考 function getSignature ()，結果 APP Secret-Key 加上 SignatureTime，使用 base64 進行加密

token : 使用 uniqid () 生成 token

tokenValid : 使用 time() + (60\*5) 產生當下的 timestamp + 5分鐘

最後，使用 SESSION，將 token / tokenValid 綁定 emp\_no 做紀錄

* + - * function chkWebAuthData ()
        + 使用 Web 呼叫 API，驗證 Header 帶入的資料，檢查 Signature / token 是否過期
        + 驗證

SignatureTime

Signature

token

tokenValid

* + - * + 執行流程

SignatureTime : 驗證 time() - SignatureTime 是否大於 5分鐘

Signature : 驗證資料是否正確，參考 function getSignature ()

檢查 SESSION[ emp\_no ] 的 token / tokenValid

1. 確認 SESSION[ emp\_no ] 存在，未被清除

2. 確認 SESSION 的 token / tokenValid 和 Header 帶入的 token / tokenValid 資料內容相同

3. 確認 SESSION 的 tokenValid 小於 time ()

* + - * fucntion delWebAuthData ()
        + 清除 SESSION[ emp\_no ]
* qplay/
  + app/ Http/ **routes.php**
    - URL Request Handle :
      * 新增 QPayUserMaintain/
        + QPayUserManager，顯示消費券管理員的 UI。
        + newQPayManager，新增消費券管理員。
        + editQPayManager，編輯消費券管理員。
        + getQPayManagerList，取得消費券管理員的資料。
        + QPayUserPointType，取得消費券類型的 UI。
        + newQPayPointType，新增消費券類型。
        + editQPayPointType，編輯消費券類型。
        + getQPayPointTypeList，取得消費券類型的資料。
        + QPayUserShop，顯示消費券店家的 UI。
        + newQPayShop，新增消費券店家。
        + editQPayShop，編輯消費券店家。
        + resetQPayShopQAccountPwd，重置消費券店家的 QAccount Login Pwd。
        + getQPayShopList，取得消費券店家的資料。
        + QPayUserMember，顯示消費券員工的 UI。
        + resetQPayMemberTradePwd，重置消費券員工的交易密碼。
        + checkMemberExist，檢查消費券員工是否存在於資料表中。
        + newMember，新增消費券員工資料。
        + getQPayMemberList，取得消費券員工的資料。
      * 新增 QPayStoreMaintain
        + QPayStorePoint，顯示點數儲值的 UI。
        + uploadPointExcel，上傳消費券點數儲值名單 Excel。
        + newPointStore，新增消費券儲值紀錄。
        + newMemberPoint，新增單一員工的點數配發紀錄。
        + checkEmpExist，檢查 消費券點數儲值名單 Excel 中的員工資料，是否所有 emp\_no 皆存在於 DB Table `qp\_user`。
        + QPayStoreRecord，顯示儲值紀錄的 UI。
        + downloadPointExcel，下載消費券點數儲值名單 Excel。
        + getQPayStoreRecordList，取得儲值紀錄的資料。
        + QPayStoreEmployee，顯示員工發放紀錄的 UI。
        + getQPayPointGetRecordList，取得員工發放紀錄的資料。
      * 新增 QPayReimburseMaintain
        + QPayReimburseFinance，顯示財務請款的 UI。
        + downloadReimburseFinanceExcel，下載財務請款 Excel。
        + getQPayReimburseFinanceList，取得財務請款的資料。
        + QPayReimbursePurchase，顯示員工消費記錄的 UI。
        + getQPayReimbursePurchaseList，取得員工消費記錄的資料。
  + app/ Http/ Controllers/ **qpayController.php**
    - 定義 function
      * QPayUserManager ()
        + return view “qpay\_maintain/ manager”。
      * newQPayManager ()。
      * editQPayManager ()。
      * getQPayManagerList ()。
      * QPayUserPointType ()
        + return view “qpay\_maintain/ pointType”。
      * newQPayPointType ()。
      * editQPayPointType ()。
      * getQPayPointTypeList ()。
      * QPayUserShop ()
        + return view “qpay\_maintain/ shop”。
      * newQPayShop ()。
      * editQPayShop ()。
      * resetQPayShopQAccountPwd ()。
      * getQPayShopList ()。
      * QPayUserMember
        + return view “qpay\_maintain/ member”。
      * resetQPayMemberTradePwd ()。
      * checkMemberExist ()。
      * newMember ()。
      * getQPayMemberList ()。
      * QPayStorePoint
        + return view “qpay\_maintain/ pointStore”。
      * uploadPointExcel ()。
      * newPointStore()。
      * newMemberPoint()。
      * checkEmpExist ()。
      * QPayStoreRecord
        + return view “qpay\_maintain/ pointStoreRecord”。
      * downloadPointExcel ()。
      * getQPayStoreRecordList ()。
      * QPayStoreEmployee
        + return view “qpay\_maintain/ pointGetRecord”。
      * getQPayPointGetRecordList ()。
      * QPayReimburseFinance
        + return view “qpay\_maintain/ reimburseFinance”。
      * downloadReimburseFinanceExcel ()。
      * getQPayReimburseFinanceList ()。
      * QPayReimbursePurchase
        + return view “qpay\_maintain/ reimbursePurchase”。
      * getQPayReimbursePurchaseList ()。
  + app/ Model/
    - **QP\_User.php**
      * 新增定義 function checkUserForQPayManager ()
        + 檢查此 User Domain + Emp No 是否存在
        + 帶入

`qp\_user`.emp\_no

`qp\_user`.user\_domain

* + - * + 回傳

Exist return =>`qp\_user`.row\_id

Not Exist return => null

* + - * 新增定義 function newUser ()
        + 新增資料至 `qp\_user`
        + 帶入

login\_id

emp\_no

emp\_name

password

emp\_id

email

ext\_no

user\_domain

site\_code

company

department

status

resign

ad\_flag

source\_from

* + - * + 預設

resign\_message = N

reset\_pwd = N

created\_user = 當前 User

created\_at = 當前 Timestamp

updated\_at = 當前 Timestamp

* + - * + 回傳

`qp\_user`.row\_id

* + - * + 備註 :

可以參考 AppRepository.php > function insertAppHead ()

的 SQL 用法，直接取得 insertGetId。

* + - * 新增定義 function editUser ()
        + 編輯 `qp\_user`
        + 帶入

row\_id

emp\_name

ext\_no

status

* + - * 新增定義 function resetQAccountPwd ()
        + 重置 QAccount Login Password
        + 帶入

`qp\_user`.row\_id

* + - * + 執行

取得 `qp\_user`.emp\_id

使用 PHP password\_hash () 進行加密

algorithms : PASSWORD\_BCRYPT

$options [ 'cost' => 10 ]

存入`qp\_user`.password

* + - * + 回傳

重置成功

重置失敗

* + - * + 備註 :

之後一般員工的 Reset QAccount Login Password 也是使用此 function。

* + - * 新增定義 function getUserEmpID ()
        + 取得員工的身份證後 6 碼
        + 帶入

`qp\_user`.row\_id

* + - * + 回傳

`qp\_user`.emp\_id

* + - * 新增定義 function getAllDepartment ()
        + 取得員工資料中所有的部門代碼
        + 回傳

`qp\_user`.department

* + - **QPay\_Manager.php**
      * 定義 DB `qplay` Table `qpay\_manager`。
    - **QPay\_Point\_Type.php**
      * 定義 DB `qplay` Table `qpay\_point\_type`。
    - **QPay\_Point\_Store.php**
      * 定義 DB `qplay` Table `qpay\_point\_store`。
    - **QPay\_Shop.php**
      * 定義 DB `qplay` Table `qpay\_shop`。
    - **QPay\_Member.php**
      * 定義 DB `qplay` Table `qpay\_member`。
    - **QPay\_Member\_Point.php**
      * 定義 DB `qplay` Table `qpay\_member\_point`。
    - **QPay\_Point\_Trade\_Log.php**
      * 定義 DB `qplay` Table `qpay\_point\_trade\_log`。
  + app/ Repositories/
    - **QPayManagerRepository.php**
      * 定義 function checkQPayManager ()
        + 檢查此消費券管理員是否已經存在
        + 帶入

`qp\_user`.row\_id

* + - * + 回傳

已經存在

不存在

* + - * 定義 function newQPayManager ()
        + 新增消費券管理員
        + 帶入

`qp\_user`.row\_id

* + - * + 回傳

新增成功

新增失敗

* + - * 定義 function editQPayManager ()
        + 編輯消費券管理員
        + 帶入

`qpay\_manager`.row\_id

status

* + - * + 回傳

更新成功

更新失敗

* + - * 定義 function getQPayManagerList ()
        + 取得消費券管理員的資料
        + 回傳

`qp\_user`.emp\_name

`qp\_user`.company

`qpay\_manager`.status

`qpay\_manager`.row\_id

* + - **QPayPointTypeRepository.php**
      * 定義 function newQPayPointType ()
        + 新增消費券類別
        + 帶入

名稱

顏色代碼

* + - * + 回傳

新增成功

新增失敗

* + - * 定義 function editQPayPointType ()
        + 編輯消費券類別
        + 帶入

`qpay\_point\_type`.row\_id

name

color

status

* + - * + 回傳

更新成功

更新失敗

* + - * 定義 function getQPayPointTypeList ()
        + 取得消費券類別的資料
        + 回傳

`qpay\_point\_type`.row\_id

`qpay\_point\_type`.name

`qpay\_point\_type`.color

`qpay\_point\_type`.status

* + - **QPayPointStoreRepository.php**
      * 定義 function uploadPointExcel ()
        + 上傳 消費券點數儲值名單 Excel

將消費券類型 row\_id 儲存於 SESSION [消費券類型]

將 Excel 檔案名稱儲存於 SESSION [原始檔名]

另外透過 PHP uniqid () 產生 UNIQUE 的 SESSION [暫存檔名]

存放至暫存資料夾

* + - * + 帶入

`qpay\_point\_type`.row\_id

* + - * + 回傳

處理成功後，執行 checkEmpExist ()

* + - * 定義 function downloadPointExcel ()
        + 下載 消費券點數儲值名單 Excel

檔案位於 上傳資料夾

下載的時候，使用 `qpay\_point\_store`.file\_original 取代

`qpay\_point\_store`.file\_saved

* + - * + 帶入

`qpay\_point\_store`.row\_id

* + - * 定義 function checkEmpExist ()
        + 檢查 Excel 內的所有 emp\_no 是否存在 DB Table `qp\_user`
        + 回傳

return view(qpay\_maintain/ pointStore.blade.php)

並且帶入下列資料

PHP 陣列 A

每人匯入的點數

總共人數

點數總額

比對的結果

OK

NO 以及有問題的 emp\_no 名單

* + - * + 備註 :

處理流程 :

透過 Laravel-Excel 讀取暫存資料夾內的 Excel :

SESSION [暫存檔名]

將 Excel 資料轉換為 PHP 陣列 A，儲存於 SESSION [Excel Data]

取得所有工號資料，以逗號區隔，組成字串

ex: 1609009, 11004100, 11111108

使用 SQL : SELECT emp\_no, company

FROM `qp\_user` WHERE

emp\_no IN (1609009, 11004100, 11111108) AND resign = Y 取得回傳值，將資料轉換為 PHP 陣列 B，

key : emp\_no，value : company

比對 陣列 A 和 陣列 B 的 emp\_no + comapny

透過 foreach () 處理 陣列 A

先使用 PHP in\_array (needle, array, true) 檢查是否存在

接著再檢查 陣列B [emp\_no] 的 comapny 是否等於 陣列A [emp\_no] 的 comapny

兩者完全一致，表示所有 Excel 內的員工名單，皆存在 `qp\_user`，回傳 OK，並且回傳以下資料 :

每人匯入的點數 (全部員工應該一樣)

總共人數

點數總額 (每人匯入的點數 \* 總共人數)

兩者有差異，表示 Excel 內的員工名單，有部分員工不存在於 `qp\_user`，回傳 NO，並且回傳有問題的員工 emp\_no

* + - * 定義 function newPointStore ()
        + 新增消費券儲值紀錄
        + 帶入

SESSION [消費券類型]

SESSION [原始檔名]

SESSION [暫存檔名]

* + - * + 回傳

Y : 新增成功，以及 `qpay\_point\_store`.row\_id

N : 新增失敗

* + - * + 備註 :

處理流程 :

將剛剛位於暫存資料夾的 Excel 檔案，

繼續使用 SESSION [暫存檔名]，並且將檔案移轉到上傳資料夾。

刪除暫存資料夾的 Excel 檔案。

將資料新增至 DB Table `qpay\_point\_store`。

回傳新增的 row\_id。

* + - * 定義 function newMemberPoint ()
        + 新增單一員工的點數配發紀錄
        + 帶入

`qpay\_member`.row\_id

`qpay\_point\_store`.row\_id

* + - * + 回傳

新增成功

新增失敗

* + - * + 備註 :

處理流程 :

新增資料時

`qpay\_member\_point`.stored\_total =

`qpay\_member\_point`.stored\_now

`qpay\_member\_point`.stored\_used = 0

* + - * 定義 function pointStore ()
        + 將 消費券點數儲值名單 Excel 的資料進行儲值
        + 使用 SESSION [Excel Data] 進行資料的處理
        + 回傳

儲值成功

儲值失敗

* + - * + 備註 :

處理流程 :

DB Table `qpay\_point\_store`

file\_original : SESSION [原始檔名]

file\_saved : SESSION [暫存檔名]

新增資料，使用 QPayPointStoreRepository.php 定義的 newPointStore ()，

取得 `qpay\_point\_store`.row\_id。

DB Table `qpay\_member`

將 SESSION [Excel Data] 中的所有 emp\_no 取出

檢查所有 emp\_no 是否存在於 `qpay\_member`， 使用 QPayMemberRepository.php 定義

的 checkMemberExist ()

回傳資料 > 已經存在的 `qpay\_member`.row\_id

回傳資料 > 不存在的 emp\_no :

透過 foreach () 處理陣列資料

使用 Auth::user()->checkUserForQPayManager() 取得 `qp\_user`.row\_id。

Auth::user()->getUserEmpID

取得 `qp\_user`.emp\_id。

透過 QPayMemberRepository.php 定義

的 newMember () 新增消費券員工資料。

取得 `qpay\_member`.row\_id

所有員工的皆存在 or 新增至 `qpay\_member` 之後，將所有的 `qpay\_member`.row\_id 儲存至陣列 C

DB Table `qpay\_member\_point

透過 foreach() 一筆一筆處理上面第二步驟取得陣列 C 資料中的

`qpay\_member`.row\_id

依據第一步驟取得 `qpay\_point\_store`.row\_id

依據 SESSION [Excel Data] 紀錄的

每人匯入的點數

使用 QPayPointStoreRepository.php 定義

的 newMemberPoint ()，新增單一員工的點數配發紀錄。

所有員工的資料處理完成後回傳結果。

* + - * 定義 function getQPayStoreRecordList ()
        + 取得消費券點數儲值紀錄的資料
        + 帶入

開始日期

結束日期

* + - * + 回傳

`qpay\_point\_store`.row\_id

`qpay\_point\_store`.created\_at

`qp\_user`.emp\_name

`qpay\_point\_type`.name

`qpay\_point\_store`.member\_count

`qpay\_point\_store`.stored\_total

* + - * + 備註 :

儲值碼

使用 `qpay\_point\_store`.row\_id 組成 6 位數的數字，左邊自動補 0

前方加上字串 [ S ]

ex :

`qpay\_point\_store`.row\_id = 1

儲值碼 : S000001

* + - * 定義 function getQPayPointGetRecordList ()
        + 取得員工發放紀錄的資料
        + 帶入

`qpay\_point\_type`.row\_id

`qp\_user`.department

`qp\_user`.emp\_no

開始日期

結束日期

* + - * + 回傳

`qp\_user`.emp\_no

`qp\_user`.emp\_name

`qpay\_point\_type`.name

`qp\_user`.department

`qpay\_member\_point`.created\_at

`qpay\_member\_point`.stored\_total

`qpay\_member\_point`.point\_store\_row\_id

* + - * + 備註 :

儲值碼

使用 `qpay\_member\_point`.point\_store\_row\_id 組成 6 位數的數字，左邊自動補 0

前方加上字串 [ S ]

ex :

`qpay\_member\_point`.point\_store\_row\_id = 1

儲值碼 : S000001

* + - **QPayShopRepository.php**
      * 定義 function newQPayShop ()
        + 新增消費券店家
        + 帶入

名稱

地址

電話

帳號

密碼

* + - * + 回傳

新增成功

新增失敗

* + - * + 備註 :

處理流程 :

先將以下資料，透過 QP\_User.php 的 newUser ()，

Insert `qp\_user`，並且取得新產生 row\_id:

名稱 > `qp\_user`.emp\_name

電話 > `qp\_user`.ext\_no

帳號 > `qp\_user`.emp\_no

帳號 > `qp\_user`.login\_id

密碼 (明碼) > `qp\_user`.emp\_id

密碼 (加密) > `qp\_user`.password

字串 ‘shop’ > `qp\_user`.email

字串 ‘shop’ > `qp\_user`.user\_domain

字串 ‘shop’ > `qp\_user`.site\_code

字串 ‘shop’ > `qp\_user`.company

字串 ‘shop’ > `qp\_user`.department

字串 ‘shop’ > `qp\_user`.source\_from

字串 ‘Y’ > `qp\_user`.status

字串 ‘Y’ > `qp\_user`.resign

字串 ‘N’ > `qp\_user`.ad\_flag

HR 給予店家的 QAccount 預設登入密碼 :

要以明碼的形式，存入`qp\_user`.emp\_id，之後進行 Reset QAccount Password，會使用此資料做處理。

另外，使用 PHP password\_hash () 進行加密

algorithms : PASSWORD\_BCRYPT

$options [ 'cost' => 10 ]

存入`qp\_user`.password

* + - * 定義 function editQPayShop ()
        + 編輯消費券店家
        + 帶入

`qpay\_shop`.row\_id

`qp\_user`.emp\_name

`qpay\_shop`.address

`qp\_user`.ext\_no

`qp\_user`.status

`qpay\_shop`.trade\_status

* + - * + 回傳

更新成功

更新失敗

* + - * + 備註 :

處理流程 :

先將以下資料，透過 QP\_User.php 的 editUser ()，

Update `qp\_user` :

`qp\_user`.emp\_name

`qp\_user`.ext\_no

`qp\_user`.status

接著更新 `qpay\_shop` 的以下資料 :

`qpay\_shop`.address

`qpay\_shop`.trade\_status

* + - * 定義 function resetQPayShopQAccountPwd ()
        + 重置消費券店家的 QAccount Login Pwd
        + 帶入

`qpay\_shop`.row\_id

* + - * + 回傳

重置成功

重置失敗

* + - * + 備註 :

處理流程 :

先透過 `qpay\_shop`.row\_id 至 `qpay\_shop` 取得店家的 `qp\_user`.row\_id。

接著再透過 QP\_User.php 的 resetQAccountPwd () 進行重置。

* + - * 定義 function getQPayShopList ()
        + 取得消費券店家的資料
        + 回傳

`qpay\_shop`.row\_id

`qp\_user`.emp\_name

`qpay\_shop`.address

`qp\_user`.ext\_no

`qp\_user`.login\_id

`qp\_user`.status

`qpay\_shop`.trade\_status

* + - **QPayMemberRepository.php**
      * 定義 function getQPayMemberList ()
        + 取得消費券員工的資料
        + 帶入

`qpay\_point\_type`.row\_id

`qp\_user`.department

`qp\_user`.emp\_no

* + - * + 回傳

`qp\_user`.emp\_no

`qp\_user`.emp\_name

`qpay\_point\_type`.name

`qp\_user`.department

* + - * 定義 function resetQPayMemberTradePwd ()
        + 重置消費券密碼
        + 帶入

`qpay\_member`.row\_id

* + - * + 回傳

重置成功

重置失敗

* + - * + 備註 :

處理流程 :

先透過 `qpay\_member`.row\_id 至 `qpay\_member` 取得員工的 `qp\_user`.row\_id。

接著再透過 QP\_User.php 的 getUserEmpID () 取得員工的 `qp\_user`.emp\_id。

然後，將 `qp\_user`.emp\_id 的後 4 碼

使用 PHP password\_hash () 進行加密

algorithms : PASSWORD\_BCRYPT

$options [ 'cost' => 10 ]

存入`qpay\_member`.trade\_password。

* + - * 定義 function checkMemberExist ()
        + 檢查消費券員工是否存在於 DB Table `qpay\_member` 中
        + 帶入

emp\_no 的陣列資料

* + - * + 回傳

已經存在的 `qpay\_member`.row\_id 陣列資料

不存在 `qpay\_member` 中的 emp\_no 陣列資料

* + - * + 備註 :

處理流程 :

將 emp\_no 的陣列資料 處理為字串

1609009, 11004100, 11111108

DB Table `qpay\_member` LEFT JOIN `qp\_user`

ON `qp\_user`.row\_id = `qpay\_member`.user\_row\_id

WHERE `qp\_user`.row\_id IN (emp\_no 字串)

將結果與帶入的 emp\_no 的陣列資料 做比對

比對成功 : 表示存在 `qpay\_member`

比對失敗 : 表示不存在 `qpay\_member`

將兩種結果都處理為陣列

* + - * 定義 function newMember ()
        + 新增消費券員工資料
        + 帶入

`qp\_user`.row\_id

`qp\_user`.emp\_id

* + - * + 回傳

`qpay\_member`.row\_id

* + - * + 備註 :

處理流程 :

`qp\_user`.emp\_id 只取用後面 4 位資料。

接著使用 PHP password\_hash () 進行加密

algorithms : PASSWORD\_BCRYPT

$options [ 'cost' => 10 ]

存入`qpay\_member`.trade\_password

回傳剛剛新增成功的 row\_id。

* + - **QPayMemberPointRepository.php**
      * 定義 function getQPayPointGetRecordList ()
        + 取得員工發放紀錄的資料
        + 帶入

`qpay\_point\_type`.row\_id

`qp\_user`.department

`qp\_user`.emp\_no

開始日期

結束日期

* + - * + 回傳

`qp\_user`.emp\_no

`qp\_user`.emp\_name

`qpay\_point\_type`.name

`qp\_user`.department

`qpay\_member\_point`.stored\_total

`qpay\_member\_point`.point\_store\_row\_id

* + - * + 備註 :

儲值碼

使用 `qpay\_member\_store`.point\_store\_row\_id 組成 6 位數的數字，左邊自動補 0

前方加上字串 [ S ]

ex :

`qpay\_member\_store`.point\_store\_row\_id = 1

儲值碼 : S000001

* + - * 定義 function getQPayPointNow ()
        + 取得員工剩餘點數的資料
        + 帶入

`qp\_user`.emp\_no

`qp\_user`.user\_domain

* + - * + 回傳

SUM ( `qpay\_member\_point`.stored\_now )

`qp\_user`.emp\_name

* + - **QPayPointTradeLogRepository**
      * 定義 function downloadReimburseFinanceExcel ()
        + 下載財務請款的資料
        + 帶入

`qpay\_point\_type`.name

`qp\_user`.emp\_name (店家名稱)

月份

* + - * + 回傳

`qpay\_point\_trade\_log`.created\_at

Total of ( `qpay\_point\_trade\_log`.trade\_point ) by day

`qp\_user`.emp\_name (店家名稱)

`qpay\_point\_type`.name

Total of ( `qpay\_point\_trade\_log`.trade\_point ) by month

* + - * + 備註 :

處理流程 :

透過 SQL Command 取得資料之後

將資料依據同一天 / 同月份做整合

透過 Laravel-Excel 將資料匯出為 Excel

最後將 Excel 進行下載

* + - * 定義 function getQPayReimburseFinanceList ()
        + 取得財務請款的資料
        + 帶入

`qpay\_point\_type`.row\_id

`qpay\_shop`.row\_id

月份

* + - * + 回傳

`qpay\_point\_trade\_log`.created\_at

`qpay\_point\_trade\_log`.trade\_point

`qpay\_point\_trade\_log`.row\_id

`qp\_user`.emp\_name (店家名稱)

`qpay\_point\_type`.name

Total of ( `qpay\_point\_trade\_log`.trade\_point ) by month

* + - * + 備註 :

交易碼

使用 `qpay\_point\_trade\_log`.row\_id 組成 6 位數的數字，左邊自動補 0

前方加上字串 [ T ]

ex :

`qpay\_point\_trade\_log`.row\_id = 1

交易碼 : T000001

* + - * 定義 function getQPayReimbursePurchaseList ()
        + 取得員工消費紀錄的資料
        + 帶入

`qp\_user`.emp\_no

`qp\_user`.user\_domain

開始日期

結束日期

* + - * + 回傳

From getQPayReimbursePurchaseList ()

`qpay\_point\_trade\_log`.created\_at

`qp\_user`.emp\_name (店家名稱)

`qpay\_point\_trade\_log`.trade\_point

`qpay\_point\_trade\_log`.row\_id

From getQPayPointNow ()

SUM ( `qpay\_member\_point`.stored\_now )

`qp\_user`.emp\_name (員工姓名)

* + - * + 備註 :

交易碼

使用 `qpay\_point\_trade\_log`.row\_id 組成 6 位數的數字，左邊自動補 0

前方加上字串 [ T ]

ex :

`qpay\_point\_trade\_log`.row\_id = 1

交易碼 : T000001

取得 SQL 回傳的資料之後，再執行

QPayMemberPointRepository.php 定義的

function getQPayPointNow () 取得員工剩餘點數的資料，處理完成之後，在將所有資料一併回傳至 View。

* + app/ Services/ **qpayService.php**
    - 定義 function
      * newQPayManager ()
        + 使用 Auth::user()->checkUserForQPayManager() 驗證使用者是否存在。

是 : 使用QPayManagerRepository.php

定義的 newQPayManager ()。

否 : 回傳錯誤訊息 [使用者不存在]。

* + - * editQPayManager ()
        + 使用QPayManagerRepository.php 定義的 editQPayManager ()。
      * getQPayManagerList ()
        + 使用QPayManagerRepository.php 定義的 getQPayManagerList ()。
      * newQPayPointType ()
        + 使用QPayPointTypeRepository.php 定義的 newQPayPointType ()。
      * editQPayPointType ()
        + 使用QPayPointTypeRepository.php 定義的 editQPayPointType ()。
      * getQPayPointTypeList ()
        + 使用QPayPointTypeRepository.php 定義的 getQPayPointTypeList ()。
      * newQPayShop ()
        + 使用QPayShopRepository.php 定義的 newQPayShop ()。
      * editQPayShop ()
        + 使用QPayShopRepository.php 定義的 editQPayShop ()。
      * resetQPayShopQAccountPwd ()
        + 使用QPayShopRepository.php 定義的 resetQPayShopQAccountPwd ()。
      * getQPayShopList ()
        + 使用QPayShopRepository.php 定義的 getQPayShopList ()。
      * resetQPayMemberTradePwd ()
        + 使用QPayMemberRepository.php 定義的 resetQPayMemberTradePwd ()。
      * checkMemberExist ()
        + 使用QPayMemberRepository.php 定義的 checkMemberExist ()。
      * newMember ()
        + 使用QPayMemberRepository.php 定義的 newMember ()。
      * getQPayMemberList ()
        + 使用QPayMemberRepository.php 定義的 getQPayMemberList ()。
      * uploadPointExcel ()
        + 使用QPayPointStoreRepository.php 定義的 uploadPointExcel ()。
      * newPointStore ()
        + 使用QPayPointStoreRepository.php 定義的 newPointStore ()。
      * newMemberPoint ()
        + 使用QPayPointStoreRepository.php 定義的 newMemberPoint ()。
      * checkEmpExist ()
        + 使用QPayPointStoreRepository.php 定義的 checkEmpExist ()。
      * downloadPointExcel ()
        + 使用QPayPointStoreRepository.php 定義的 downloadPointExcel ()。
      * getQPayStoreRecordList
        + 使用QPayPointStoreRepository.php 定義的 getQPayStoreRecordList ()。
      * getQPayPointGetRecordList ()
        + 使用QPayPointStoreRepository.php 定義的 getQPayPointGetRecordList ()。
      * downloadReimburseFinanceExcel ()
        + 使用QPayPointTradeLogRepository.php 定義的

downloadReimburseFinanceExcel ()。

* + - * getQPayReimburseFinanceList ()
        + 使用QPayPointTradeLogRepository.php 定義的

getQPayReimburseFinanceList ()。

* + - * getQPayReimbursePurchaseList ()
        + 使用QPayPointTradeLogRepository.php 定義的

getQPayReimbursePurchaseList ()。

* + resources/ views/ **qpay\_maintain**/
    - **manager.blade.php**
      * 下拉選單 - 公司
        + 使用 CompanyService.php 定義的

function geEnableCompanyList ()，取得公司資料

* + - * + Option - Value : `qp\_company`.user\_domain
        + Option - Text : `qp\_company`.name
      * 新增消費券管理員 :
        + 選擇公司，輸入工號，點擊 [新增管理員]，透過 AJAX 將資料送到 Route : newQPayManager 做處理，顯示 AJAX 回傳的結果訊息。
        + 如果新增成功，更新 gridQPayManagerList。
      * 編輯消費券管理員的狀態 :
        + 點擊管理員的名字，開啟 dialog\_content。
        + 編輯

狀態 : 正常 / 停用

* + - * + 透過 AJAX 將資料送到 Route : QPayUserMaintain/ editQPayManager 做處理，顯示 AJAX 回傳的結果訊息。
        + 如果編輯成功，更新 gridQPayManagerList。
      * 消費券管理員資料列表 :
        + 使用 bootstrap table

id = gridQPayManagerList

data-url = QPayUserMaintain/ getQPayManagerList

data-field

`qp\_user`.emp\_name

`qp\_user`.company

`qpay\_manager`.status

* + - **pointType.blade.php**
      * 新增消費券類別 :
        + 選擇顏色，輸入名稱點擊 [新增]，透過 AJAX 將資料

送到 Route : newQPayPointType 做處理，顯示 AJAX 回傳的結果訊息。

* + - * + 如果新增成功，更新 gridQPayPointTypeList。
      * 編輯消費券類別 :
        + 點擊消費券類別的名字，開啟 dialog\_content。
        + 編輯

名稱

狀態 : 正常 / 停用

顏色

* + - * + 透過 AJAX 將資料送到 Route : QPayUserMaintain/ editQPayPointType 做處理，顯示 AJAX 回傳的結果訊息。
        + 如果編輯成功，更新 gridQPayPointTypeList。
      * 消費券類別資料列表 :
        + 使用 bootstrap table

id = gridQPayPointTypeList

data-url = QPayUserMaintain/ getQPayPointTypeList

data-field

`qpay\_point\_type`.name

`qpay\_point\_type`.color

`qpay\_point\_type`.status

* + - * 顏色選擇器
        + 產生 3 X 3 共 9 種顏色的表格，每一格的顏色代表

一種 HTML Color Code。

* + - * + ex: #08298A
        + 將此 HTML Color Code 存入 DB。
    - **shop.blade.php**
      * 新增消費券店家 :
        + 輸入名稱 / 地址 / 電話 / 帳號 / 密碼，點擊 [新增店家]，透過 AJAX 將資料

送到 Route : newQPayShop 做處理，顯示 AJAX 回傳的結果訊息。

* + - * + 如果新增成功，更新 gridQPayShopList。
        + 備註 :

密碼的長度限制為 6。

* + - * 編輯消費券店家 :
        + 點擊消費券店家的名稱 or 使用狀態 or 交易狀態，開啟 dialog\_content。
        + 編輯

名稱

地址

電話

使用狀態 : 正常 / 停用

交易狀態 : 正常 / 停用

* + - * + 透過 AJAX 將資料送到 Route : QPayUserMaintain/ editQPayShop 做處理，顯示 AJAX 回傳的結果訊息。
        + 如果編輯成功，更新 gridQPayShopList。
      * 消費券類別資料列表 :
        + 使用 bootstrap table

id = gridQPayShopList

data-url = QPayUserMaintain/ getQPayShopList

data-field

`qp\_user`.emp\_name

`qpay\_shop`.address

`qp\_user`.ext\_no

`qp\_user`.login\_id

`qp\_user`.status

`qpay\_shop`.trade\_status

* + - * 重置店家的 QAccount 登入密碼
        + 點擊 [Reset]，跳出提示確認視窗，如果按下確認 :

透過 AJAX 將資料送到

Route : QPayUserMaintain/ resetQPayShopQAccountPwd 做處理，顯示 AJAX 回傳的結果訊息。

* + - **member.blade.php**
      * 消費券員工資料列表 :
        + 使用 bootstrap table

id = gridQPayMemberList

data-url = QPayUserMaintain/ getQPayMemberList

data-field

`qp\_user`.emp\_no

`qp\_user`.emp\_name

`qpay\_point\_type`.name

`qp\_user`.department

* + - * + 下拉選單 - 消費券類型

使用 QPayPointTypeRepository.php 定義的

function getQPayPointTypeList ()，取得消費券類型資料

Option - Value : `qpay\_point\_type`.row\_id

Option - Text : `qpay\_point\_type`.name

* + - * + 部門代碼的 Autocomplete

使用 QP\_User.php 定義的 fucntion getAllDepartment () ，

取得所有的部分代碼。

將資料傳到前端，儲存至 Javascript 陣列中。

每此輸入任何文字，即時監聽事件，比對陣列的資料，在輸入框的下方顯示過濾之後的選項。

* + - * 重置員工的 QAccount 登入密碼
        + 點擊 [Reset]，跳出提示確認視窗，如果按下確認 :

透過 AJAX 將資料送到

Route : QPayUserMaintain/ resetQPayMemberTradePwd 做處理，顯示 AJAX 回傳的結果訊息。

* + - **pointStore.blade.php**
      * 匯入員工名單
        + 依據操作步驟，產生 2 個 Div

Div #step1 (display: block)

Div #step2 (display: none)

* + - * + 下拉選單 - 消費券類型

使用 QPayPointTypeRepository.php 定義的

function getQPayPointTypeList ()，取得消費券類型資料

Option - Value : `qpay\_point\_type`.row\_id

Option - Text : `qpay\_point\_type`.name

* + - * + 選擇檔案，點擊按鈕 [匯入員工名單]

透過 input type=file 讓 User 選擇 Excel 檔案，點擊 [匯入員工名單] 之後，透過 QPayPointStoreRepository.php

定義的 uploadPointExcel () 進行上傳。

承上，上傳成功後，透過 QPayPointStoreRepository.php 定義的 checkEmpExist () 進行檢查，完成之後，將 SESSION [Excel Data] 傳送到此頁面 Reload，將陣列資料呈現在 bootstrap table

id = gridQPayPointStoreExcelList

data-url =

data-field

emp\_no

emp\_name

company

department

承上，依據 checkEmpExist () 的結果，跳出提示視窗:

OK : 資料核對無誤

按鈕 [下一步] Enable

NO : 以下工號核對有誤!! 請查明後重新匯入

工號 12085556 工號 1424545 …..

按鈕 [下一步] Disable

* + - * + 按鈕 [下一步]

Div #step1 (display: none)

Div #step2 (display: block)

checkEmpExist () 假設結果為 OK，會回傳以下資料 :

每人匯入的點數

總共人數

點數總額 (每人匯入的點數 \* 總共人數)

顯示上述資料於頁面

另外顯示 交易日期 / 管理員 emp\_name 於頁面

* + - * + 按鈕 [確認儲值]

點擊之後，跳出提示視窗:

取消 : 關閉視窗

確認 : 執行 QPayPointStoreRepository.php 定義的

pointStore ()

依據 pointStore () 的執行結果，顯示

成功

失敗

* + - * + 按鈕 [查看儲值紀錄]

點擊之後，轉址到 QPayStoreMaintain/ QPayStoreRecord

* + - **pointStoreRecord.blade.php**
      * 消費券點數儲值紀錄的資料列表 :
        + 使用 bootstrap table

id = gridQPayStoreRecordList

data-url = QPayStoreMaintain/ getQPayStoreRecordList

data-field

`qpay\_point\_store`.row\_id

`qpay\_point\_store`.created\_at

`qp\_user`.emp\_name

`qpay\_point\_type`.name

`qpay\_point\_store`.member\_count

`qpay\_point\_store`.stored\_total

* + - * + 日期區間選擇

開始日期

結束日期

* + - * + 按鈕 [ 查詢 ]

點擊之後，透過設定 bootstrap table 的 queryParams，帶入日期參數至後端

* + - * + 按鈕 [ 下載 ]

點擊之後，執行 QPayPointStoreRepository.php 定義

downloadPointExcel ()，帶入 `qpay\_point\_store`.row\_id

* + - **pointGetRecord.blade.php**
      * 員工發放紀錄的資料列表 :
        + 使用 bootstrap table

id = gridQPayPointGetRecordList

data-url = QPayStoreMaintain/ getQPayPointGetRecordList

data-field

`qp\_user`.emp\_no

`qp\_user`.emp\_name

`qpay\_point\_type`.name

`qp\_user`.department

`qpay\_member\_point`.stored\_total

`qpay\_member\_point`.point\_store\_row\_id

* + - * + 下拉選單 - 消費券類型

使用 QPayPointTypeRepository.php 定義的

function getQPayPointTypeList ()，取得消費券類型資料

Option - Value : `qpay\_point\_type`.row\_id

Option - Text : `qpay\_point\_type`.name

* + - * + 部門代碼的 Autocomplete

使用 QP\_User.php 定義的 fucntion getAllDepartment () ，

取得所有的部分代碼。

將資料傳到前端，儲存至 Javascript 陣列中。

每此輸入任何文字，即時監聽事件，比對陣列的資料，在輸入框的下方顯示過濾之後的選項。

* + - * + 工號的輸入框
        + 日期區間選擇

開始日期

結束日期

* + - * + 按鈕 [ 查詢 ]

點擊之後，透過設定 bootstrap table 的 queryParams，帶入

消費券類型

部門代碼

員工工號

日期參數

這些資料傳送至後端

* + - **reimburseFinance.blade.php**
      * 財務請款的資料列表 :
        + 使用 bootstrap table

id = gridQPayReimburseFinanceList

data-url = QPayReimburseMaintain/

getQPayReimburseFinanceList

data-field

`qpay\_point\_trade\_log`.created\_at

`qpay\_point\_trade\_log`.trade\_point

`qpay\_point\_trade\_log`.row\_id

備註 :

透過 bootstrap table 的 responseHandler 取得 Server Side 回傳的店家名稱 / 交易日期 / 交易總額 / 消費券類型，顯示在資料列表的上方

* + - * + 下拉選單 - 消費券類型

使用 QPayPointTypeRepository.php 定義的

function getQPayPointTypeList ()，取得消費券類型資料

Option - Value : `qpay\_point\_type`.row\_id

Option - Text : `qpay\_point\_type`.name

* + - * + 下拉選單 - 店家

使用 QPayPointShopRepository.php 定義的

function getQPayShopList ()，取得店家資料

Option - Value : `qpay\_shop`.row\_id

Option - Text : `qpay\_shop`.name

* + - * + 下拉選單 - 月份
        + 按鈕 [ 查詢 ]

點擊之後，透過設定 bootstrap table 的 queryParams，帶入

`qpay\_point\_type`.row\_id

`qpay\_shop`.row\_id

月份

這些資料傳送至後端

* + - * + 按鈕 [ 下載 ]

點擊之後，執行 QPayPointTradeLogRepository.php 定義

downloadReimburseFinanceExcel()，帶入

`qpay\_point\_type`.row\_id

`qpay\_shop`.row\_id

月份

* + - **reimbursePurchase.blade.php**
      * 員工消費紀錄的資料列表 :
        + 下拉選單 - 公司

使用 CompanyService.php 定義的

function geEnableCompanyList ()，取得公司資料

Option - Value : `qp\_company`.user\_domain

Option - Text : `qp\_company`.name

* + - * + 工號的輸入框
        + 日期區間選擇

開始日期

結束日期

* + - * + 使用 bootstrap table

id = gridQPayReimbursePurchaseList

data-url = QPayReimburseMaintain/

getQPayReimbursePurchaseList

data-field

`qpay\_point\_trade\_log`.created\_at

`qp\_user`.emp\_name

`qpay\_point\_trade\_log`.trade\_point

`qpay\_point\_trade\_log`.row\_id

備註 :

透過 bootstrap table 的 responseHandler 取得 Server Side 回傳的員工姓名 / 工號 / 剩餘金額，顯示在資料列表的上方

* + 第三層的選單
    - [ 用戶管理 ]
      * 名稱 : 管理員資料
        + 鏈結 : QPayUserManager
      * 名稱 : 消費券類型
        + 鏈結 : QPayUserPointType
      * 名稱 : 店家資料
        + 鏈結 : QPayUserShop
      * 名稱 : 員工資料
        + 鏈結 : QPayUserEmployee
    - [ 儲值管理 ]
      * 名稱 : 點數儲值
        + 鏈結 : QPayStorePoint
      * 名稱 : 儲值紀錄
        + 鏈結 : QPayStoreRecord
      * 名稱 : 員工發放紀錄
        + 鏈結 : QPayStoreEmployee
    - [ 報銷管理 ]
      * 名稱 : 財務請款
        + 鏈結 : QPayReimburseFinance
      * 名稱 : 員工消費記錄
        + 鏈結 : QPayReimbursePurchase
    - [ 報表分析 ]
* QPlay Backend Platform
  + [ 系統管理 ] > [ 選單維護 ] > [ 新增 ]
    - 菜單名 : QPAY\_MAINTAIN
    - 英文名 : QPay Maintain
    - 簡體中文名 : 消费券管理
    - 繁體中文名 : 消費券管理
  + [ 系統管理 ] > [ 選單維護 ] > 點擊 ( **功能管理** ) > [ 新增 子選單 ]
    - 菜單名 : QPAY\_USER\_MAINTAIN
      * 鏈結 : QPayUserMaintain
      * 英文名 : User Maintain
      * 簡體中文名 : 用户管理
      * 繁體中文名 : 用戶管理
    - 菜單名 : QPAY\_STORE\_MAINTAIN
      * 鏈結 : QPayStoreMaintain
      * 英文名 : Store Maintain
      * 簡體中文名 : 储值管理
      * 繁體中文名 : 儲值管理
    - 菜單名 : QPAY\_REIMBURSE\_MAINTAIN
      * 鏈結 : QPayReimburseMaintain
      * 英文名 : Reimburse Maintain
      * 簡體中文名 : 报销管理
      * 繁體中文名 : 報銷管理
    - 菜單名 : QPAY\_REPORT\_ANALYSIS
      * 鏈結 : QPayReportAnalysis
      * 英文名 : Report Analysis
      * 簡體中文名 : 报表分析
      * 繁體中文名 : 報表分析
  + [ 系統管理 ] > [ 群組維護 ]
    - Administrator : 預設有最高權限，可以使用全部群組。
  + 上述動作完成之後，必須修改以下檔案，以 (菜單名) 為變數名稱，建立相對應的資料
    - qplay/ resources/ lang/ en/ **messages.php**
      * 'TITLE\_QPAY\_USER\_MAINTAIN' => 'User Maintain'
      * 'TITLE\_QPAY\_STORE\_MAINTAIN' => 'Store Maintain'
      * 'TITLE\_QPAY\_REIMBURSE\_MAINTAIN' => 'Reimburse Maintain'
      * 'TITLE\_QPAY\_REPORT\_ANALYSIS' => 'Report Analysis'
    - qplay/ resources/ lang/ zh-cn/ **messages.php**
      * 'TITLE\_QPAY\_USER\_MAINTAIN' => '用户管理'
      * 'TITLE\_QPAY\_STORE\_MAINTAIN' => '储值管理'
      * 'TITLE\_QPAY\_REIMBURSE\_MAINTAIN' => '报销管理'
      * 'TITLE\_QPAY\_REPORT\_ANALYSIS' => '报表分析'
    - qplay/ resources/ lang/ zh-tw/ **messages.php**
      * 'TITLE\_QPAY\_USER\_MAINTAIN' => '用户管理'
      * 'TITLE\_QPAY\_STORE\_MAINTAIN' => '儲值管理'
      * 'TITLE\_QPAY\_REIMBURSE\_MAINTAIN' => '報銷管理'
      * 'TITLE\_QPAY\_REPORT\_ANALYSIS' => '報表分析'
  + 最後，要看 User 的操作權限設定選項中的 [功能管理] 是否被勾起。
    - 備註 : 就算是 Administrator 群組，個人的操作權限不一定跟群組一樣。
  + [ APP 管理 ] > [ App 管理 ] > QPlay > 錯誤代碼 **Error.json** 新增代碼
    - 000923
      * zh-tw => 店家未啟動交易，請聯絡相關人員
      * zh-cn => 店家未启动交易，请联络相关人员
      * en-us => The shop is disabled to be traded, please contact to staff.
    - 000924
      * zh-tw => 消費券餘額不足
      * zh-cn => 消费券余额不足
      * en-us => Insufficient balance of coupons.
    - 000925
      * zh-tw => 舊的交易密碼不正確
      * zh-cn => 旧的交易密码不正确
      * en-us => Old transaction password is incorrect.
    - 000926
      * zh-tw => 舊的登入密碼不正確
      * zh-cn => 旧的登入密码不正确
      * en-us => Old login password is incorrect.

4.2.13 UI 示意圖

* 參考 QPay(後台).180619pdf.pdf