|  |  |  |  |
| --- | --- | --- | --- |
| **Subject :** | QPlay Backend Platform | **Latest Ver :** | 1.3 |
| **Latest Release Date :** | 2018-07-02 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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 Manage ]  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 |

# 目錄

[**目錄**](#_fvukxvw609q9) **3**

[**1.0 系統設計 (System Design)**](#_17qb5z9rsywe) **5**

[**2.0 資料庫設計 (Database Design)**](#_svpws9pqxbdm) **6**

[**3.0 Implementation**](#_l3yah7s6jx9k) **6**

[**4.0 功能修改記錄**](#_y9sxdk33wuqt) **7**

[4.1 [ QPlay 後台新增功能 [ 公司管理 ]，以及 QPlay Login Page 使用動態公司選項，和 QPlay Login 流程修改，另外修正 QPlay Backend Login - Remember Me ]](#_f9crrxf6b8wb) 7

[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 ]](#_k9pqb9346spl) 19

[4.3 [ QPlay 後台新增功能 [ Function Manage ]，DB `qplay` 新增 Function Manage 相關 Table，QPlay API 新增 Function Manage 相關 API ]](#_xfzx9vmshw0k) 26

[4.4 [ Plugin QLogin 頁面 支援 QAccount，QPlay API login 支援 QAccount，APP QPlay 使用 Function Manage ]](#_hfvgfgw3nbq9) 42

# 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

## 4.1 [ QPlay 後台新增功能 [ 公司管理 ]，以及 QPlay Login Page 使用動態公司選項，和 QPlay Login 流程修改，另外修正 QPlay Backend Login - Remember Me ]

4.1.1 需求說明

* APP QPlay Login UI 上面的公司選項，目前為 Hard Code 寫死，考量到之後陸續會有新公司加入集團，公司選項需要變更為動態產生，由 QPlay Backend 提供資料。
* QPlay Backend Platform Login UI 上面的公司選項，目前為 Hard Code 寫死，考量到之後陸續會有新公司加入集團，公司選項需要變更為動態產生。
* QPlay Backend Platform 新增功能 [ 公司管理 ]，管理公司名稱，以及驗證 AD 的方式和 Server IP。
* QPlay Backend Login 的 Remember Me 無法正確運作，修正。

4.1.2 時程評估

* [ QPlay 後台新增功能 (公司管理) ]
  + 預估時程 : 5/29 ~ 6/1 (4天)
  + 實際時程 : 5/29 ~ 5/31 (3天)
* [ QPlay Login Page 使用動態公司選項 ]
  + 預估時程 : 6/4 ~ 6/5 (2天)
  + 實際時程 : 6/1 (3hr)
* [ QPlay Login 流程修改 ]
  + 預估時程 : 6/6 ~ 6/8 (3天)
  + 實際時程 :6/1 (3hr)
* [ QPlay Backend Login - Remember Me ]
  + 預估時程 : 6/6 ~ 6/8 (3天)
  + 實際時程 :6/1 (2hr)

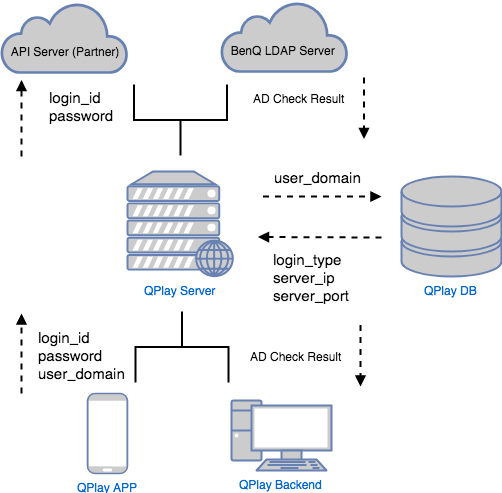
4.1.3 對應文件

* QPlay\_DB\_Schema\_3.0\_20180523.docx

4.1.4 修改項目

1. DB `qplay`
   1. 新增
      1. Table `**qp\_company**`
      2. Table `**qp\_company\_log**`
2. qplayApi/ qplayApi/
   1. 修改
      1. app/ Http/ **routes.php**
      2. app/ Http/ Controllers/ **qplayController.php**
      3. resources/ views/ **login.blade.php**
   2. 新增
      1. app/ Model/
         1. **QP\_User.php**
         2. **QP\_Company.php**
      2. app/ Repositoreies/ **CompanyRepository.php**
      3. app/ Services/ **CompanyService.php**
3. qplay/
   1. 修改
      1. app/ Http/ **routes.php**
      2. app/ Http/ Controllers/ **AuthController.php**
      3. app/ lib/ **ResultCode.php**
      4. resources/ views/ auth/ **login.blade.php**
      5. resources/ lang
         1. en/ **messages.php**
         2. zh-cn/ **messages.php**
         3. zh-tw/ **messages.php**
   2. 新增
      1. app/ Model/
         1. **QP\_Company.php**
         2. **QP\_Company\_Log.php**
      2. app/ Repositoreies/
         1. **CompanyRepository.php**
         2. **CompanyLogRepository.php**
      3. app/ Http/ Controllers/ **companyController.php**
      4. app/ Services/ **CompanyService.php**
      5. resources/ views/ company\_maintain/
         1. **company\_list.blade.php**
4. 系統操作
   1. [ 系統管理 ] > [ 選單維護 ] > [ 新增 ] 建立 ( **公司管理** )
   2. [ 系統管理 ] > [ 選單維護 ] > 點擊 ( **公司管理** ) > [ 新增 子選單 ] 建立 ( **公司管理** )
   3. [ 系統管理 ] > [ 群組維護 ] 設定可以使用 ( **公司管理** ) 的 QPlay Backend User

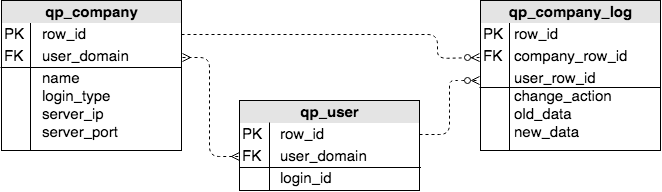
4.1.5 功能架構圖



4.1.6 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，判定屬於哪家公司，決定使用哪種登入驗證方式。 |
| MySQL | Table | qp\_company | 維護公司資料，設定 LDAP 或 API 登入，以及 Server IP。 |
| MySQL | Table | qp\_company\_log | 維護公司資料，操作紀錄 Log。 |

4.1.7 實體關聯圖



4.1.8 資料表結構

* 參考文件 : QPlay\_DB\_Schema\_3.0\_20180523.docx

4.1.9 公用元件

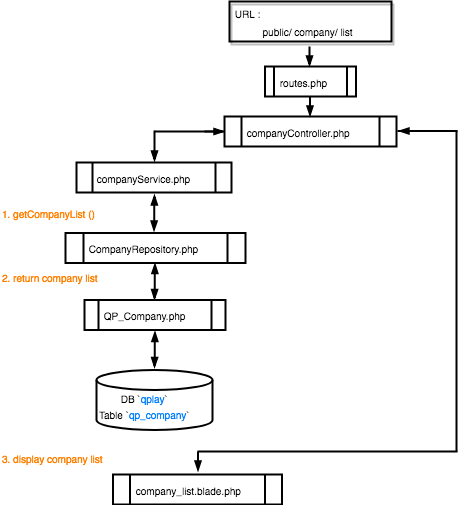
* 內部模組

4.1.10 共用介面

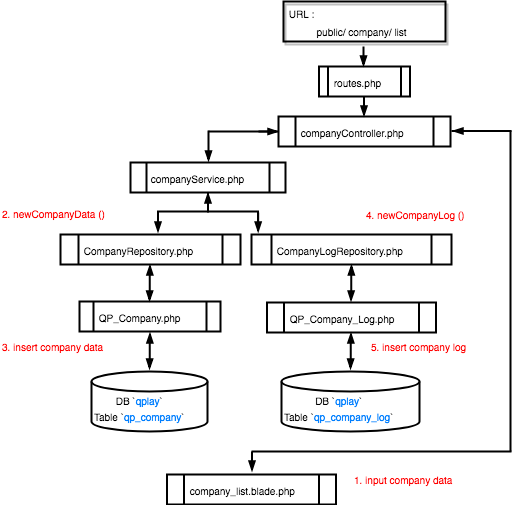
* 無

4.1.11 資料流程圖 Data Flow Diagram

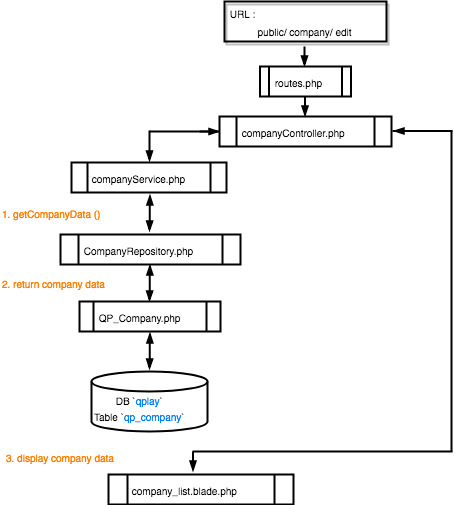
* QPlay 後台的新功能 [ 公司管理 ] - 資料列表



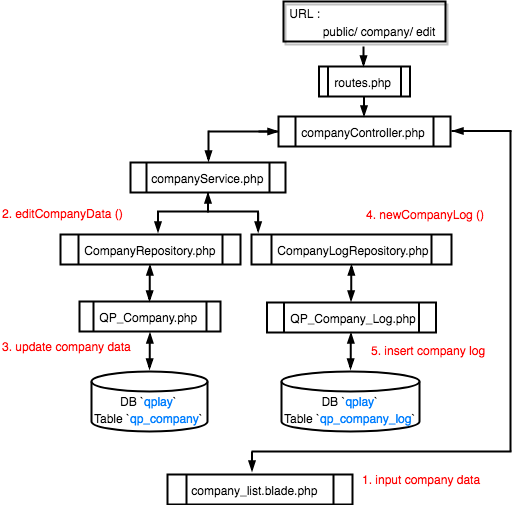
* QPlay 後台的新功能 [ 公司管理 ] - 新增資料



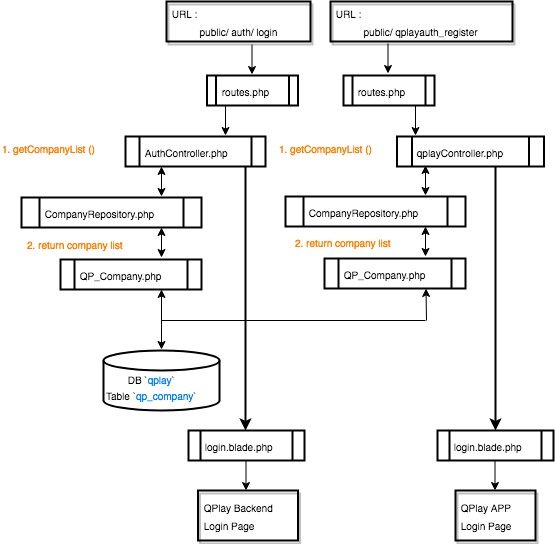
* QPlay 後台的新功能 [ 公司管理 ] - 資料編輯 - 顯示資料



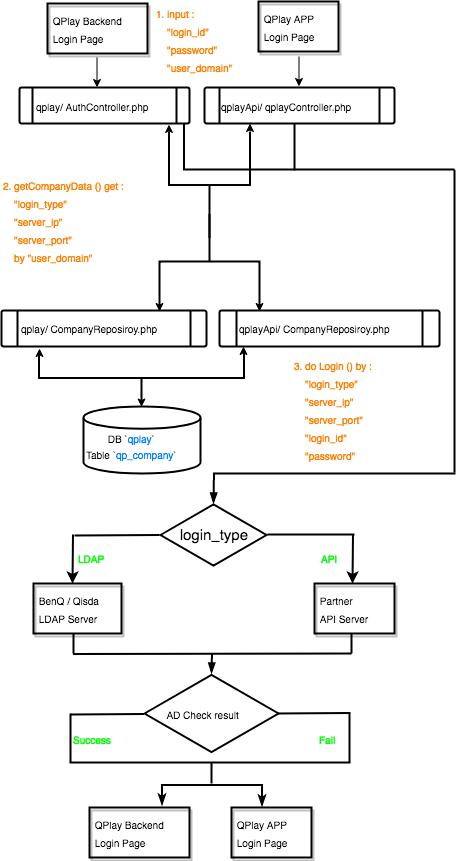
* QPlay 後台的新功能 [ 公司管理 ] - 資料編輯 - 更新資料



* QPlay Login Page 使用動態公司選項



* QPlay Login 流程修改



4.1.12 程式碼修改記錄

1. qplayApi/ qplayApi/
   1. app/ Http/ **routes.php**
      1. 由先 URL Request : /qplayauth\_register 會直接指派 login.blade.php 呈現 View。現在修改流程，先經過 qplayController.php 處理。
   2. app/ Http/ Controllers/ **qplayController.php**
      1. 動態公司選項 :
         1. 透過 ComponyRepository.php 取得 公司資料。
         2. 指派 login.blade.php 呈現 QPlay APP Login View。
      2. Login 流程修改 :
         1. 依據 User 輸入的 login\_id，透過 QP\_User.php 取得 user\_domain。
         2. 依據 user\_domain，透過 CompanyRepository.php 取得 login\_type 以及 server\_ip 和 server\_port。
         3. 使用 LDAP or API 驗證 User AD。
   3. resources/ views/ **login.blade.php**
      1. 下拉選單 id = ddlCompany 的 選項內容，原先 Hard Code 寫為 BENQ 跟 QGROUP，現在要改為由資料庫提供動態資料。
   4. app/ Model/ **QP\_User.php**
      1. 定義 DB `qplay` Table `qp\_user`。
   5. app/ Model/ **QP\_Company.php**
      1. 定義 DB `qplay` Table `qp\_company`。
   6. app/ Services/ **CompanyService.php**
      1. 透過 CompanyRepository.php 取得公司資料。
   7. app/ Repositoreies/ **CompanyRepository.php**
      1. 取得 company list。
      2. 取得欄位資料 “login\_type” “server\_ip” “server\_port”。
2. qplay/
   1. app/ Http/ **routes.php**
      1. 由先 URL Request : auth/ login 會直接指派 views/ auth/ login.blade.php 呈現 View。現在修改流程，先經過 authController.php 處理。
      2. 新增 URL Request Handle : companyMaintain，經由 CompanyController.php 處理 [ 公司管理 ]。
   2. app/ Http/ Controllers/ **AuthController.php**
      1. 動態公司選項 :
         1. 透過 ComponyRepository.php 取得 公司資料。
         2. 指派 views/ auth/ login.blade.php 呈現 QPlay Backend Login View。
      2. Login 流程修改 :
         1. 依據 User 輸入的 login\_id，透過 QP\_User.php 取得 user\_domain。
         2. 依據 user\_domain，透過 CompanyRepository.php 取得 login\_type 以及 server\_ip 和 server\_port。
         3. 使用 LDAP or API 驗證 User AD。
      3. Login Page - Remember Me
         1. 增加 Session 記憶 lang / loginid / domain / remember。
   3. app/ lib/ **ResultCode.php**
      1. 針對 [公司管理] 的資料維護結果，新增 Result code。
   4. app/ Http/ Controllers/ **CompanyController.php**
      1. 依據 URL Request，決定如何使用 CompanyService.php。
   5. app/ Services/ **CompanyService.php**
      1. 公司資料列表 :
         1. 透過 ComponyRepository.php 取得 公司資料 做列表呈現。
      2. 公司資料新增 :
         1. 透過 ComponyRepository.php 新增 公司資料。
      3. 公司資料編輯 :
         1. 透過 ComponyRepository.php 編輯 公司資料。
      4. 公司資料操作 Log :
         1. 只要新增 or 編輯公司資料，透過 ComponyLogRepository.php 寫入 Log。
   6. resources/ views/ auth/ **login.blade.php**
      1. 下拉選單 id = domain 的 選項內容，原先 Hard Code 寫為 BenQ 跟 Qgroup，現在要改為由資料庫提供動態資料。
      2. 使用 Session 來處理 Remember Me 功能。
   7. app/ Model/ **QP\_Company.php**
      1. 定義 DB `qplay` Table `qp\_company`。
   8. app/ Model/ **QP\_Company\_log.php**
      1. 定義 DB `qplay` Table `qp\_company\_log`。
   9. app/ Repositoreies/ **CompanyRepository.php**
      1. 取得 所有公司資料。
      2. 取得欄位資料 “login\_type” “server\_ip” “server\_port”。
      3. 新增 公司資料。
      4. 編輯 公司資料。
   10. app/ Repositoreies/ **CompanyLogRepository.php**
       1. 新增 company 資料異動時的 Log。
   11. resources/ views/ company\_maintain/ **company\_list.blade.php**
       1. 呈現 所有公司的列表資料。
       2. 使用 Javascript 動態產生視窗，新增公司資料，送到 Server 端做處理。
       3. 必須設定 $menu\_name，才能在選單看到 [公司管理]。
   12. resources/ views/ company\_maintain/ **company\_detail.blade.php**
       1. 呈現 單一公司的詳細資料。
       2. 編輯 公司資料，送到 Server 端做處理。
3. QPlay Backend Platform
   1. [ 系統管理 ] > [ 選單維護 ] > [ 新增 ]
      1. 菜單名 : COMPANY\_MAINTAIN
      2. 英文名 : Company Maintain
      3. 簡體中文名 : 公司管理
      4. 繁體中文名 : 公司管理
   2. [ 系統管理 ] > [ 選單維護 ] > 點擊 ( **公司管理** ) > [ 新增 子選單 ]
      1. 菜單名 : COMPANY\_MAINTAIN
      2. 鏈結 : companyMaintain
      3. 英文名 : Company Maintain
      4. 簡體中文名 : 公司管理
      5. 繁體中文名 : 公司管理
   3. [ 系統管理 ] > [ 群組維護 ]
      1. Administrator : 預設有最高權限，可以使用全部群組。
   4. 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
      1. qplay/ resources/ lang/ en/ **messages.php**
      2. qplay/ resources/ lang/ zh-cn/ **messages.php**
      3. qplay/ resources/ lang/ zh-tw/ **messages.php**
   5. 最後，要看 User 的操作權限設定選項中的 [公司管理] 是否被勾起。
      1. 備註 : 就算是 Administrator 群組，個人的操作權限不一定跟群組一樣。

## 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 ]

4.1.1 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 : emp\_no。
      * Return : 點數總額 / 店家列表。
    - API newTrade ( 交易處理 )
      * Post : emp\_no / shop\_id / price / trade\_pwd。
      * Return : 交易結果/ 點數剩餘量 / 交易代碼。
    - API getTradeRecord ( 交易紀錄 )
      * Post : emp\_no。
      * Return : 交易紀錄/ 儲值紀錄。
    - API updateTradePassword ( 更新交易密碼 )
      * Post : emp\_no / old trade\_pwd / new trade\_pwd。
      * Return : 結果。
  + **更新 QAccount Password**
    - User 的 company = shop，可以進行 QAccount Password 的更新。
* **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。

4.1.2 時程評估

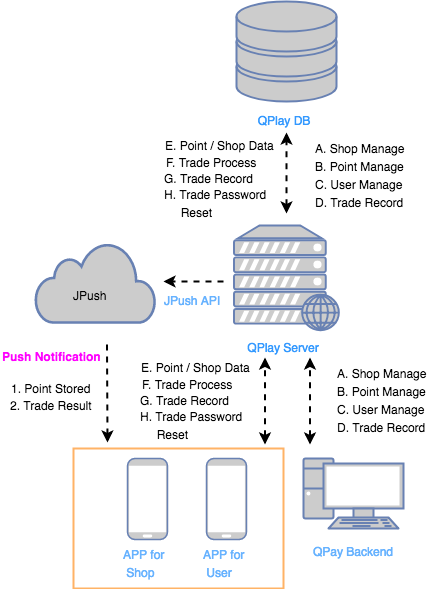
4.1.3 對應文件

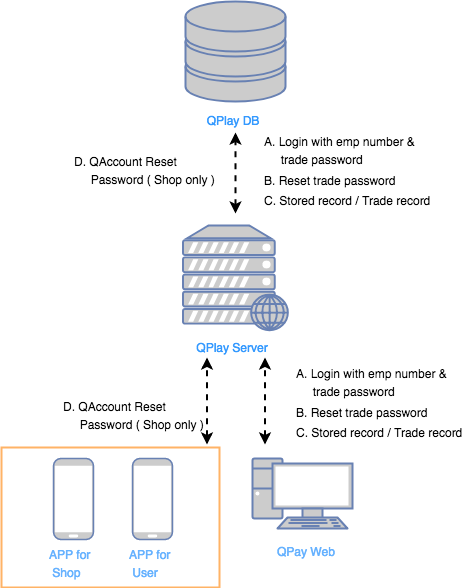
* QPlay\_API\_1.3.1\_20180619.docx
* QPlay\_DB\_Schema\_3.4\_20180619.docx

4.1.4 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/ Controllers/ **qplayController.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. **financeReport.blade.php**
      9. **tradeRecord.blade.php**

4.1.5 功能架構圖

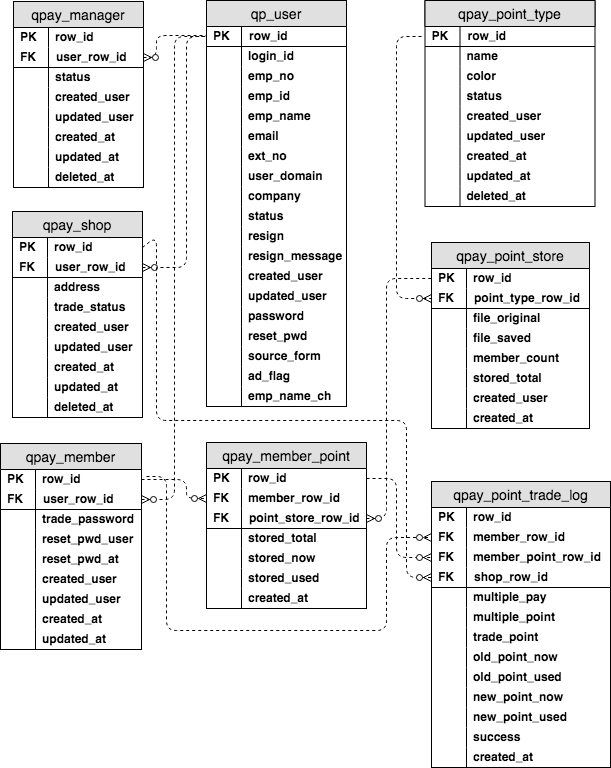




4.1.6 資料庫物件清單

|  |  |  |  |
| --- | --- | --- | --- |
| **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。 |

4.1.7 實體關聯圖



4.1.8 資料表結構

* 參考文件 : QPlay\_DB\_Schema\_3.4\_20180619.docx

4.1.9 公用元件

* 內部模組

4.1.10 共用介面

* QPlay\_API\_1.3.1\_20180619.docx

4.1.11 資料流程圖 Data Flow Diagram

4.1.12 程式碼修改記錄

* qplay/
  + app/ Http/ **routes.php**
    - URL Request Handle :
      * 新增 QPayUserMaintain/
        + QPayUserManagerr，顯示消費券管理員的資料列表。
        + 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，取得消費券類型的資料。
        + 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，顯示消費券店家的資料列表。
        + 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。
  + 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 ()。
  + app/ Model/
    - **QP\_User.php**
      * 新增定義 function checkUserForQPayManager ()
        + 檢查此 Company + Emp No 是否存在
        + 帶入

emp\_no

user\_domain

* + - * + 回傳

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

Not Exist return => null

* + - **QPay\_Manager.php**
      * 定義 DB `qplay` Table `qpay\_manager`。
    - **QPay\_Point\_Type.php**
      * 定義 DB `qplay` Table `qpay\_point\_type`。
    - **QPay\_Shop.php**
      * 定義 DB `qplay` Table `qpay\_shop`。
  + 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

* + - **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 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

‘shop’ > `qp\_user`.email

‘shop’ > `qp\_user`.user\_domain

‘shop’ > `qp\_user`.site\_code

‘shop’ > `qp\_user`.company

‘shop’ > `qp\_user`.department

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`.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

* + - * 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

* + - * 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

* + 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 ()。
  + resources/ views/ **qpay\_maintain**/
    - **manager.blade.php**
      * 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/ editQPayShop做處理，顯示 AJAX 回傳的結果訊息。

* + 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。

## 4.3 [ QPlay 後台新增功能 [ Function Manage ]，DB `qplay` 新增 Function Manage 相關 Table，QPlay API 新增 Function Manage 相關 API ]

4.3.1 需求說明

* **QPlay 後台新增功能 [ Function Manage ]**，用來管理各個 APP 內部頁面功能的使用權限:
  + 產生以下操作功能
    - Function List
    - Function Add
      * 綁定 APP。
    - Function Edit
      * 不做刪除的動作，透過 Status 紀錄資料狀態。
  + 每個 Function 綁定一個 APP，依據 [QAccount] 或 [企業角色] 或 [使用者] 來設定使用權限。
    - 權限的比重 :
      * [QAccount] < [企業角色] < [使用者]
    - ex:
      * 重設 QAccount 密碼的權限
        + QAccount : false
        + [企業角色 Shop] : true
      * 消費券店家，屬於 QAccount 的成員，無法重設 QAccount 密碼，但是，同時屬於 [企業角色 Shop]，所以最後決定，可以重設 QAccount 密碼。
      * 沒有 AD 的員工，屬於 QAccount 的成員，最後決定，無法重設 QAccount 密碼。
* **DB `qplay` 新增 Function Manage 相關 Table**。
* **QPlay API 新增 Function Manage 相關 API** :
  + getFunctionList
    - 備註 :
      * 必須考慮到，之後會透過此 API 取得 APP List Data。
      * 以下項目，目前暫時不考慮 :
        + APP 分類。
        + APP 排序。
  + getFunctionDetail
    - 備註 :
      * 必須考慮到，之後會透過此 API 取得 APP Detail Data。

4.3.2 時程評估

* [ QPlay 後台新增功能 (Function Manage) ]
  + 後台選單參數設定 / 多國語系設定
    - 預估時程 : ? ~ ? (4 hr)
    - 實際時程 :
  + DB Table 新增
    - 預估時程 : ? ~ ? (2 hr)
    - 實際時程 :
  + Function List
    - 預估時程 : ? ~ ? (6 hr)
    - 實際時程 :
  + Function Edit
    - 預估時程 : ? ~ ? (2 Day)
    - 實際時程 :
* [ QPlay API 新增對應的 API ]
  + API getFunctionList
    - 預估時程 : ? ~ ? (3 Day)
    - 實際時程 :
  + API getFunctionDetail
    - 預估時程 : ? ~ ? (2 Day)
    - 實際時程 :

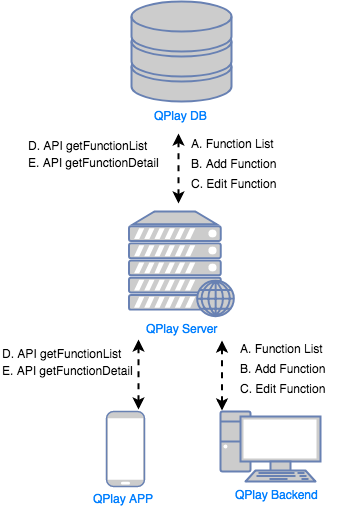
4.3.3 對應文件

* QPlay\_DB\_Schema\_3.5\_20180625.docx
* Qplay\_API\_1.3.1\_20180627.docx

4.3.4 修改項目

* DB `**qplay**`
  + 新增 Table
    - `**qp\_function**`
    - `**qp\_role\_function**`
    - `**qp\_user\_function**`
* qplayApi/ qplayApi/
  + 修改
    - app/ Http/ **routes.php**
    - app/ Http/ Controllers/ **qplayController.php**
    - app/ lib/ **CommonUtil.php**
  + 新增
    - app/ Model/
      * **QP\_Function.php**
      * **QP\_Role\_Function.php**
      * **QP\_User\_Function.php**
    - app/ Repositoreies/
      * **FunctionRepository.php**
      * **FunctionRoleRepository.php**
      * **FunctionUserRepository.php**
    - app/ Services/ **FunctionService.php**
* qplay/
  + 修改
    - app/ Http/ **routes.php**
    - app/ lib/ **ResultCode.php**
    - resources/ lang
      * en/ **messages.php**
      * zh-cn/ **messages.php**
      * zh-tw/ **messages.php**
  + 新增
    - app/ Model/
      * **QP\_Function.php**
      * **QP\_Role\_Function.php**
      * **QP\_User\_Function.php**
    - app/ Repositories/
      * **FunctionRepository.php**
      * **FunctionRoleRepository.php**
      * **FunctionUserRepository.php**
    - app/ Http/ Controllers/ **FunctionController.php**
    - app/ Services/ **FunctionService.php**
    - resources/ views/ **function\_maintain**/
      * **function\_list.blade.php**
      * **function\_detail.blade.php**

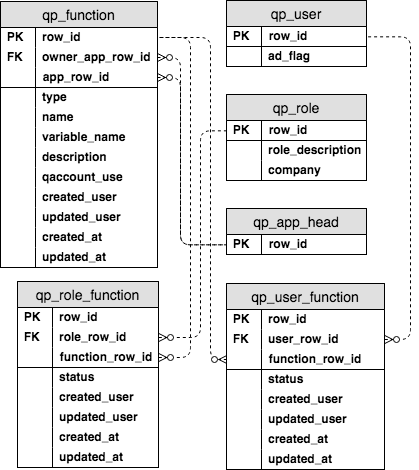
4.3.5 功能架構圖



4.3.6 資料庫物件清單

|  |  |  |  |
| --- | --- | --- | --- |
| **DB Type** | **Type** | **Object Name** | **Description** |
| MySQL | Table | qp\_function | 各個 APP 的頁面功能，可以依據 [企業角色] 或是 [使用者]，限制使用的權限。 |
| MySQL | Table | qp\_role\_function | 依據 [企業角色] 設定可以使用哪些 Function。 |
| MySQL | Table | qp\_user\_function | 依據 [使用者] 設定可以使用哪些 Function。 |
| MySQL | Table | qp\_user | 取得 User 的 row\_id 和 ad\_flag (判定是否為  QAccount)。 |
| MySQL | Table | qp\_app\_head | 取得 APP 的 row\_id。 |
| MySQL | Table | qp\_role | 取得 Role 的 row\_id。 |
| MySQL | Table | qp\_app\_line | 取得 APP 的語系資料。 |
| MySQL | Table | qp\_app\_version | 取得 APP 的版本資料。 |
| MySQL | Table | qp\_app\_pic | 取得 APP 的截圖資料。 |
| MySQL | Table | qp\_language | 取得 QPlay 的語系。 |

4.3.7 實體關聯圖



4.3.8 資料表結構

* 參考文件 : QPlay\_DB\_Schema\_3.5\_20180625.docx

4.3.9 公用元件

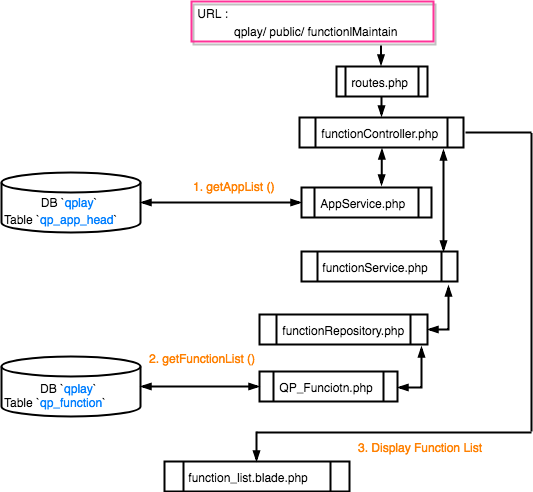
* 內部模組

4.3.10 共用介面

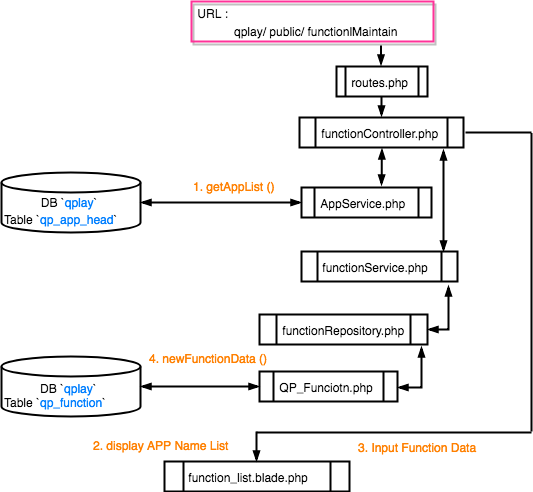
* 請參考 API 文件 : QPlay\_API\_1.3.1\_20180627.docx

4.3.11 資料流程圖 Data Flow Diagram

* QPlay 後台新增功能 [Function Manage] - Function List



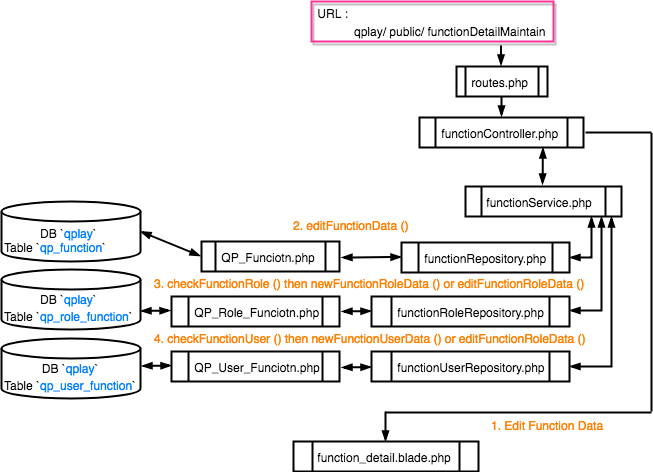
* QPlay 後台新增功能 [Function Manage] - Function Add



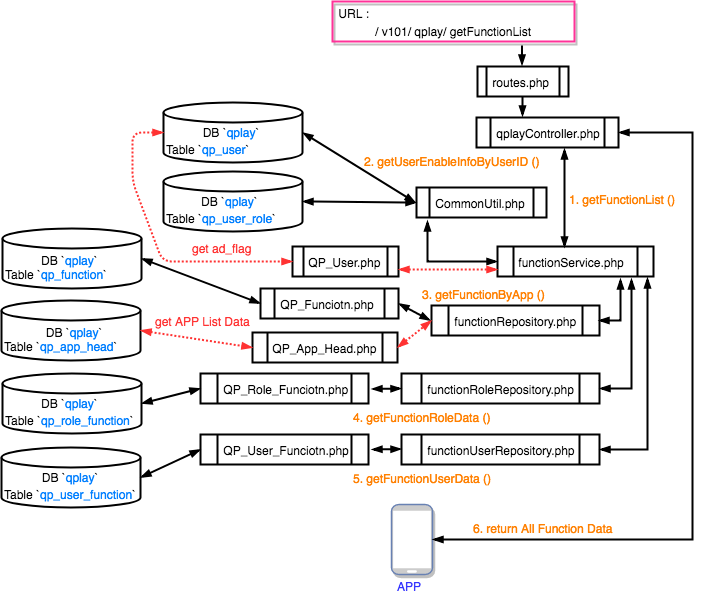
* QPlay 後台新增功能 [Function Manage] - Function Detail Data



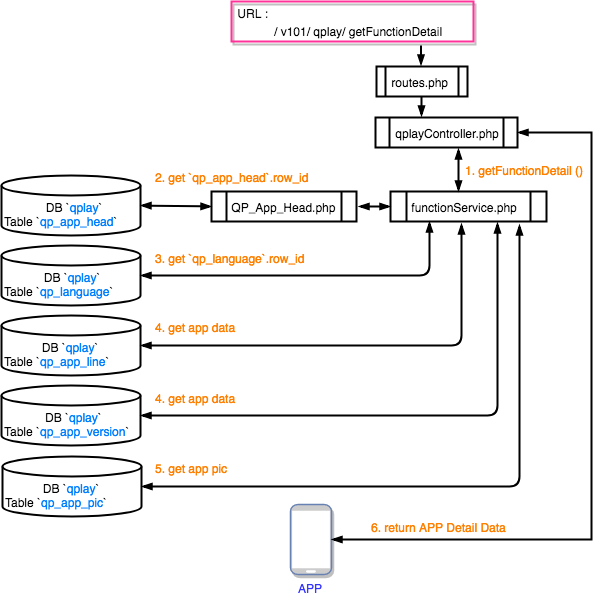
* QPlay 後台新增功能 [Function Manage] - Function Edit



* API getFunctionList



* API getFunctionDetail



4.3.12 程式碼修改記錄

1. qplayApi/ qplayApi/
   1. app/ Http/ **routes.php**
      1. 新增 URL Request Handle : getFunctionList，經由 qplayController.php 處理，

取得 User 在此 APP 的所有 Function List。

* + 1. 新增 URL Request Handle : getFunctionDetail，經由 qplayController.php 處理，

取得 User 在此 APP ，單一個 Function 的詳細資料。

* 1. app/ Http/ Controllers/ **qplayController.php**
     1. 依據 URL Request，決定如何使用 FunctionService.php。
     2. 定義 getFunctionList ()。
     3. 定義 getFunctionDetail ()。
  2. app/ lib/ **CommonUtil.php**
     1. 比照 qplay/ app/ lib/ CommonUtil.php :
        1. 新增 getUserEnableInfoByUserID ()
           1. 取得此 User 目前屬於哪些 企業角色 Role。
           2. 不需要取得以下資料 :

`qp\_user\_group`

`qp\_group\_menu`

`qp\_user\_menu`

* 1. app/ Model/
     1. **QP\_Function.php**
        1. 定義 DB `qplay` Table `qp\_function`。
     2. **QP\_Role\_Function.php**
        1. 定義 DB `qplay` Table `qp\_role\_function`。
     3. **QP\_User\_Function.php**
        1. 定義 DB `qplay` Table `qp\_user\_function`。
  2. app/ Repositoreies/
     1. **FunctionRepository.php**
        1. 定義 getFunctionByApp ()
           1. 依據 API 傳入的 APP Key 找到對應的 `qp\_app\_head`.row\_id，

在 `qp\_function` 找到相對應的 Function 列表。

* + - * 1. 依據 API 傳入的 User loginid，找到 `qp\_user`.ad\_flag，判定此 User 是否屬於 QAccount 用戶，決定此 User 是否

套用 `qp\_function`.qaccount\_use 這個資料欄位的設定值。

* + - * 1. 承上，之後 API Login 會修改回傳的資料內容，會告知 APP 端目前 User 的 ad\_flag，所以，之後可以省略上面這個步驟，會由 APP 端

告知 ad\_flag。

* + - * 1. ex:

Function [ News ] 的 `qp\_function`.qaccount\_use = N，如果是一般用戶，因為 ad\_flag = N，所以還是有權限可以使用 [ News ] ; 如果是 QAccount 用戶，因為 `qp\_user`.ad\_flag = Y，所以此 User 必須套用 qaccount\_user = N，沒有權限可以使用 [ News ]。

* + 1. **FunctionRoleRepository.php**
       1. 定義 getFunctionRoleData ()
          1. 依據 getUserEnableInfoByUserID () 的結果，透過

role\_row\_id 以及 funciton\_row\_id 在 `qp\_role\_function` 取得資料。

* + 1. **FunctionUserRepository.php**
       1. 定義 getFunctionRoleData ()
          1. 依據 User 的 `qp\_user`.row\_id 在 `qp\_user\_function` 取得資料。
  1. app/ Services/ **FunctionService.php**
     1. 定義 getFunctionList ()。
        1. 依據權限的權重，由小至大，進行下列流程處理資料，並且回傳 :
           1. FunctionRepository.php 的 getFunctionByApp () 取得此 APP 的

全部 Function，假設共有 3 個，預設全部權限皆為 N，放入陣列，

陣列的 key = `qp\_function`.variable\_name，

Array[ FunA : {right : N, type : APP}, FunB : {right : N, type : FUN}, FunC : {right : N, type : FUN}]。

* + - * 1. 檢查 User 是否為 QAccount 用戶，依據此 Function 的

`qp\_function`.qaccount\_use 決定使用權限。

* + - * 1. FunctionRoleRepository.php 的 getFunctionRoleData () 取得

Role 和 Function 之間的使用權限設定值，如果 User 所屬的 Role 可以使用 FunB，則陣列被更新為

Array[ FunA : {right : N, type : APP}, FunB : {right : Y, type : FUN}, FunC : {right : N, type : FUN}]。

* + - * 1. FunctionUserRepository.php 的 getFunctionRoleData () 取得

User 和 Function 之間的使用權限設定值，如果 User 被設定可以使用 FunC，則陣列被更新為

Array[ FunA : {right : N, type : APP}, FunB : {right : Y, type : FUN}, FunC : {right : Y, type : FUN}]。

* + - * 1. 最後，將處理過後的陣列資料，回傳至 APP 端，APP 端依據

variable\_name 的 right = Y / N，決定相對應的功能，是否可以使用。

* + - 1. 承上，如果要考慮加入 APP List :
         1. 執行 SQL 的時候，DB Table `qp\_function` 需要

LEFT JOIN DB Table `qp\_app\_head`，取得

`qp\_app\_head`.package\_name 和 `qp\_app\_head`.icon\_url。

* + - * 1. 此時的陣列資料會是下列格式 :

Array[ FunA : {right : N, type : APP, packageName : apprrs, icon : icon\_512.png}, FunB : {right : Y, type : FUN}, FunC : {right : Y, type : FUN}]。

* + - * 1. 承上，APP QPlay 可以在 APP List，依據 APP Name 和 Icon 顯示 APP 資料。
    1. 定義 getFunctionDetail ()。
       1. getFunctionList () 會告知 APP 端各個 Function 的 Type = FUN / APP，如果是 APP， 則透過此 API getFunctionDetail ()，取得 APP 的詳細資料。
       2. 依據 APP 端傳入的資料 packageName，透過 `qp\_app\_head` 找到對應

的 row\_id。

* + - 1. 使用 `qp\_app\_head`.row\_id 在 `qp\_app\_head`，`qp\_app\_line`，

`qp\_language`，`qp\_app\_version`，`qp\_app\_pic` 找到相對應的資料，整合之後回傳給 APP 端。

* + - 1. 承上，APP 的下載路徑，必須依據 `qp\_app\_line`.external\_app 的值，決定是否需要使用 FilePath::getApkDownloadUrl () 做處理，可以參考 qplayController.php。
      2. 承上，`qp\_app\_line` 和 `qp\_app\_pic` 的處理邏輯 :
         1. APP 端 Call API 時會在 URL 帶入 lang 參數，透過此參數

取得 `qp\_language`.row\_id 。

* + - * 1. 處理 `qp\_app\_head` 的時候，可以取得 default\_lang\_row\_id。
        2. 將 `qp\_app\_line` 和 `qp\_app\_pic` 中屬於此 APP 的資料都撈出來。
        3. 比對是否有 API 帶入的 lang 語系，回傳的資料中，只需要包含此語係。
        4. 如果 API 帶入的 lang 語系沒有資料存在，則使用 default\_lang\_row\_id 的資料做回傳。
        5. 總之，每次只會回傳一種語系的資料。
      1. 承上，由於之前 getAppList () 是直接在 qplayController.php 執行 SQL 撈取

`qp\_app\_head`，`qp\_language`，`qp\_app\_line`，`qp\_app\_version`，`qp\_app\_pic` 這些 Table 的資料，沒有依據這些 Table 額外新增

相關的 Repository 跟 Model，這次同樣依照此邏輯處理。

1. qplay/
   1. app/ Http/ **routes.php**
      1. 新增 URL Request Handle : functionMaintain，經由 functionController.php 處理

[ Function Manage ] 的 Function List 和 Function Add。

* + 1. 新增 URL Request Handle : functionDetailMaintain，經由 functionController.php 處理

[ Function Manage ] 的 Function Data Display 和 Function Edit。

* 1. app/ lib/ **ResultCode.php**
     1. 針對 [ Function Manage ] 的資料維護結果，新增 Result code。
        1. Function Variable Name 已經存在此 APP。
  2. resources/ lang
     1. en/ **messages.php**
     2. zh-cn/ **messages.php**
     3. zh-tw/ **messages.php**
        1. 三種語系，依據 UI 需求，增加對應的文字變數。
  3. app/ Model/
     1. **QP\_Function.php**
        1. 定義 DB `qplay` Table `qp\_function`。
     2. **QP\_Role\_Function.php**
        1. 定義 DB `qplay` Table `qp\_role\_function`。
     3. **QP\_User\_Function.php**
        1. 定義 DB `qplay` Table `qp\_user\_function`。
  4. app/ Repositories/
     1. **FunctionRepository.php**
        1. 定義 getFunctionList ()
           1. 回傳 Function 列表。
        2. 定義 newFunctionData ()
           1. 新增 Function 資料。
        3. 定義 editFunctionData ()
           1. 編輯 Function 資料。
     2. **FunctionRoleRepository.php**
        1. 定義 getFunctionRoleData ()
           1. 回傳 Role - Function 的使用權限。
        2. 定義 checkFunctionRole ()
           1. 檢查 Role - Function 是否已經存在。
        3. 定義 newFunctionRoleData ()
           1. 新增 Role - Function 使用權限。
        4. 定義 editFunctionRoleData ()
           1. 編輯 Role - Function 的使用權限。
     3. **FunctionUserRepository.php**
        1. 定義 getFunctionUserData ()
           1. 回傳 User - Function 的使用權限。
        2. 定義 checkFunctionUser ()
           1. 檢查 User - Function 是否已經存在。
        3. 定義 newFunctionUserData ()
           1. 新增 User - Function 的使用權限。
        4. 定義 editFunctionUserData ()
           1. 編輯 User - Function 的使用權限。
  5. app/ Http/ Controllers/ **FunctionController.php**
     1. 依據 URL Request，決定如何使用 FunctionService.php。
     2. 定義 functionMaintain ()。
     3. 定義 functionDetailMaintain ()。
     4. 定義 getFunctionList ()。
     5. 定義 newFunctionData ()。
     6. 定義 editFunctionData ()。
  6. app/ Services/ **FunctionService.php**
     1. 定義 getFunctionList () :
        1. 透過 FunctionRepository.php 取得 Function 資料 做列表呈現。
     2. 定義 newFunctionData () :
        1. 透過以下檔案，新增 Function 資料 :
           1. FunctionRepository.php
           2. AppService.php

使用 getAppList() 取得 APP List，每個 Function 綁定單一個 APP Key。

* + 1. 定義 editFunctionData () :
       1. 透過以下檔案，編輯 Function 資料 :
          1. FunctionRepository.php

顯示 / 編輯 Function 基本資料。

* + - * 1. FunctionRoleRepository.php

顯示 / 編輯 Role - Function 的使用權限。

如果 Role - Function 已經存在，改為變更 Status。

* + - * 1. FunctionUserRepository.php

顯示 / 編輯 User - Function 的使用權限。

如果 User - Function 已經存在，改為變更 Status。

* + - * 1. AppService.php

使用 getAppList() 取得 APP List，每個 Function 綁定單一個 APP Key。

* + - * 1. CommonUtil.php

使用 getAllCompanyRoleList() 取得 Role List，設定每個 Role - Function 的使用權限。

* 1. resources/ views/ **function\_maintain**/
     1. **function\_list.blade.php**
        1. Function 資料列表 :
           1. 使用 Bootstrap Table 產生資料列表。
           2. 顯示 APP Key / `qp\_function`.name / `qp\_function`.type

/ `qp\_function`.variable\_name / `qp\_function`.status 。

* + - 1. Function 資料新增 :
         1. 使用 Dialog UI 進行資料的新增。

透過下拉選單，選擇此項目是屬於哪個 APP Key 底下的項目。

選擇類型 Function / Appliction。

如果是 Application，另外出現下拉選單，必須選擇此

項目是對應到哪一個 APP。

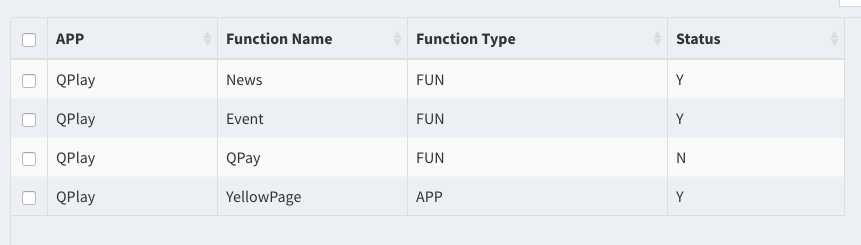
填入 Name / Variable Name / Description。

* + - * 1. DB Table `qp\_function`.qaccount\_use 預設為 N。
    1. **function\_detail.blade.php**
       1. 顯示 Function 詳細資料，進行編輯。
       2. 依據權限的權重，由小至大，UI 由上至下排列，依序設定下列項目:
          1. QAccount 的使用權限，透過 checkbox 設定。
          2. 企業角色 Role 的使用權限，透過 checkbox 設定。
          3. 單一使用者的使用權限，透過 checkbox 設定。

1. QPlay Backend Platform
   1. [ 系統管理 ] > [ 選單維護 ] > [ 新增 ]
      1. 菜單名 : FUNCTION\_MAINTAIN
      2. 英文名 : Function Maintain
      3. 簡體中文名 : 功能管理
      4. 繁體中文名 : 功能管理
   2. [ 系統管理 ] > [ 選單維護 ] > 點擊 ( **功能管理** ) > [ 新增 子選單 ]
      1. 菜單名 : FUNCTION\_MAINTAIN
      2. 鏈結 : functionMaintain
      3. 英文名 : Function Maintain
      4. 簡體中文名 : 功能管理
      5. 繁體中文名 : 功能管理
   3. [ 系統管理 ] > [ 群組維護 ]
      1. Administrator : 預設有最高權限，可以使用全部群組。
   4. 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
      1. qplay/ resources/ lang/ en/ **messages.php**
      2. qplay/ resources/ lang/ zh-cn/ **messages.php**
      3. qplay/ resources/ lang/ zh-tw/ **messages.php**
      4. 新增
         1. 'TITLE\_FUNCTION\_MAINTAIN' => Function Maintain'
   5. 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。
      1. 備註 : 就算是 Administrator 群組，個人的操作權限不一定跟群組一樣。

4.3.13 UI 示意圖

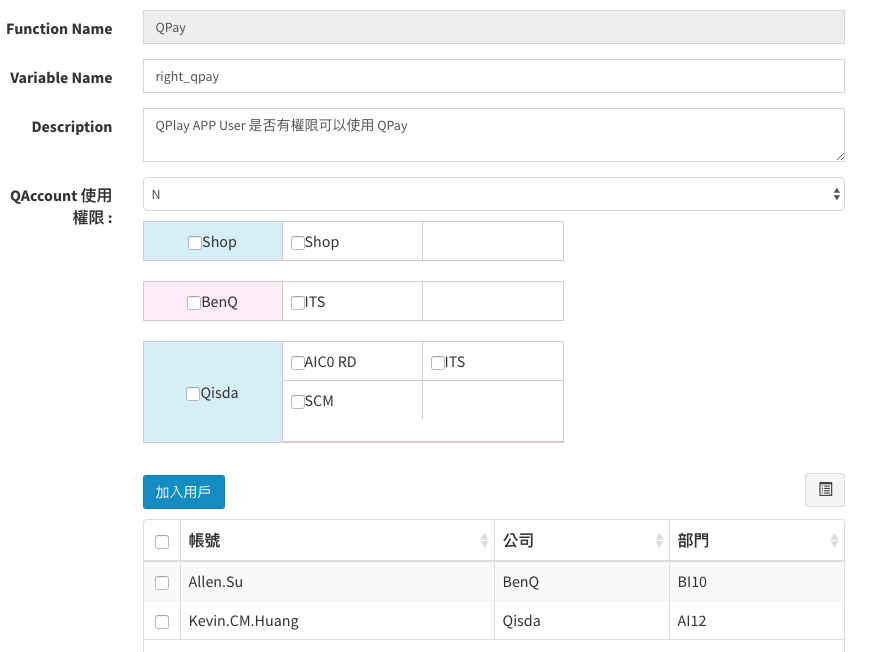
* Function Manage - Function List



* Function Manage - Function Add



* Function Manage - Function Edit



## 4.4 [ Plugin QLogin 頁面 支援 QAccount，QPlay API login 支援 QAccount，QPlay API register 支援 QAccount，APP QPlay 使用 Function Manage ]

4.4.1 需求說明

* **Plugin QLogin 頁面 支援 QAccount**
  + 修改登入頁面 UI
    - 第一階層 - 選擇公司
    - 第二階層 - 選擇登入方式
      * 使用桌機帳號
      * 其他
    - 備註 :
      * 如果第一階層選擇 [ 協力廠商 ]，則隱藏第二階層的下拉選單。
* **QPlay API register 支援 QAccount**
  + User 可以選擇 QAccount 完成登入。
  + 回傳的資料必須新增 :
    - `qp\_user`.ad\_flag
    - `qp\_user`.company
* **QPlay API login 支援 QAccount**
  + User 可以選擇 QAccount 完成登入。
  + 回傳的資料必須新增 :
    - `qp\_user`.ad\_flag
    - `qp\_user`.company
* **APP QPlay 使用 Function Manage**
  + 依據 API getFunctionList 回傳的資料，決定使用者是否可以使用下列功能 :
    - News
    - Event
    - QPay
    - Reset QAccount Password
    - 各個 APP 的使用權限
  + 備註 :
    - APP QPlay 暫時維持使用 API getAppList，之後才修改使用 API getFunctionDetail。
    - 這份 PESPIS 文件不包含 APP QPlay 程式碼修改的規劃。

4.4.2 時程評估

* [ Plugin QLogin 頁面 支援 QAccount ]
  + 預估時程 : ? ~ ? (1 Day)
  + 實際時程 :
* [ QPlay API login 支援 QAccount ]
  + 預估時程 : ? ~ ? (2 Day)
  + 實際時程 :

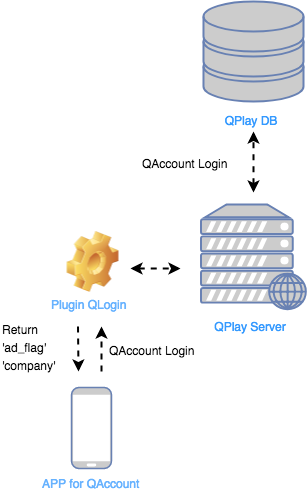
4.4.3 對應文件

* Qplay\_API\_1.3.2\_20180702.docx

4.4.4 修改項目

1. qplayApi/ qplayApi/
2. 修改
   1. app/ Http/ Controllers/ **qplayController.php**
   2. app/ lib/ **CommonUtil.php**
   3. app/ Repositoreies/ **UserRepository.php**
   4. resources/ views/ **login.blade.php**
   5. public/ js/ lang/
      1. **login-en-us.js**
      2. **login-zh-cn.js**
      3. **login-zh-tw.js**
3. 新增
   1. app/ Services/ **UserService.php**

4.4.5 功能架構圖



4.4.6 資料庫物件清單

* 無

4.4.7 實體關聯圖

* 無

4.4.8 資料表結構

* 無

4.4.9 公用元件

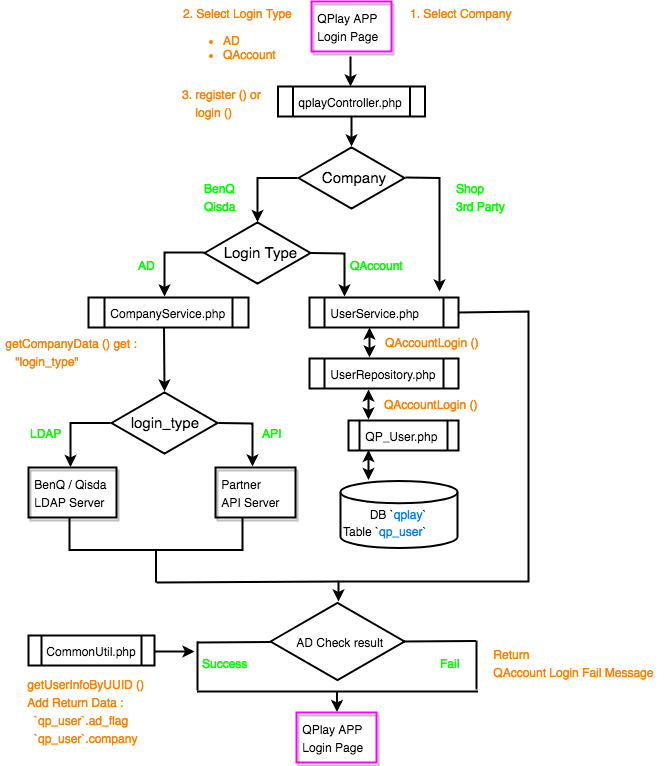
* 無

4.4.10 共用介面

* Qplay\_API\_1.3.2\_20180702.docx

4.4.11 資料流程圖 Data Flow Diagram

* Plugin QLogin 頁面 支援 QAccount
* QPlay API - register 支援 QAccount
* QPlay API - login 支援 QAccount



4.4.12 程式碼修改記錄

* qplayApi/ qplayApi/

1. app/ Http/ Controllers/ **qplayController.php**
   1. 加入 use App\ Services\ UserService.php
   2. function register () 修改執行流程，當以下條件成立時 :
      1. 登入頁面的選項 Company = 協力廠商 (shop)
      2. 登入頁面的選項 Login Type = 其他 (QAccount)
      3. 新增流程，執行 QAccountLogin ()
   3. function login () 修改執行流程，當以下條件成立時 :
      1. 登入頁面的選項 Company = 協力廠商 (shop)
      2. 登入頁面的選項 Login Type = 其他 (QAccount)
      3. 新增流程，執行 QAccountLogin ()
2. app/ lib/ **CommonUtil.php**
   1. 修改 function getUserInfoByUUID () 新增回傳資料 :
      1. `qp\_user`.ad\_flag
      2. `qp\_user`.company
3. resources/ views/ **login.blade.php**
   1. UI 調整 :
      1. 樣式修改
      2. 新增 下拉選單 [ Login Type ]
         1. 使用桌機帳號 (AD)
         2. 其他 (QAccount)
4. public/ js/ lang/
   1. **login-en-us.js**
   2. **login-zh-cn.js**
   3. **login-zh-tw.js**
      1. Plugin QLogin 登入頁面的多國語言文字設定。
      2. 依據新的需求，增加字串。
5. app/ Repositoreies/ **UserRepository.php**
   1. 定義 QAccountLogin ()
      1. 依據 User 在 Login Page 輸入的 Account / Password 進行驗證。
      2. Account 比對 `qp\_user`.loginid。
      3. Password 比對 `qp\_user`.password。
         1. `qp\_user`.password 的資料預設使用 PHP password\_hash () 進行加密。
         2. Password 必須使用 PHP password\_verify () 進行驗證。
      4. 回傳結果。
6. app/ Services/ **UserService.php**
   1. 定義 QAccountLogin ()
      1. 透過 UserRepository.php 進行 QAccountLogin ()，取得回傳結果。
      2. 登入成功 :
         1. 不區分 AD (LDAP / API) 或是 QAccount，只要登入成功，回傳的資料必須新增 :
            1. `qp\_user`.ad\_flag
            2. `qp\_user`.company
         2. 承上，這兩個資料由 CommonUtil::getUserInfoByUUID () 提供。
      3. 登入失敗 :
         1. 當 QAccount 登入失敗的時候，回傳錯誤訊息 :
            1. 以下三個錯誤，個別對應 app/ lib/ ResultCode.php 的代碼

帳號不存在 : \_000901\_userNotExistError

密碼錯誤 : \_000902\_passwordError

帳號停權 : \_000914\_userWithoutRight

4.4.13 UI 示意圖

* Plugin QLogin 頁面支援 QAccount

