|  |  |  |  |
| --- | --- | --- | --- |
| **Subject :** | QPlayPESPIS EmpService & Status | **Latest Ver :** | 1.8 |
| **Latest Release Date :** | 2019-01-15 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Change History** | | | | |
| Date | Author | Version | Type | Comment |
| 2018-12-06 | Darren.K.Ti | 1.0 | Function  DB | 1. EmpService API + Service API PESPIS Ver 1.0 |
| 2018-12-10 | Darren.K.Ti | 1.1 | Function | 1. EmpServiceServiceIDRepository.php   * function getServiceRowID ()   + SQL 條件式 必須加上     - active = Y |
| 2018-12-11 | Darren.K.Ti | 1.2 | Function | 1. API - setEmpServiceTarget   * 修改處理流程   + 新增 Target 要檢查是否已經存在 DB，如果已經存在，改為更新 `target\_id`.active |
| 2018-12-14 | Darren.K.Ti | 1.3 |  | 1. Error.json 提供多國語言的內容 |
| 2018-12-24 | Darren.K.Ti | 1.4 | Function | 1. API - newReserve   * 修改執行流程 * EmpServiceServiceService 新增   function getServiceManagerByTarget () 用來取得此 Service 管理者的 login\_id / domain   * 依據上述資料，以及 API 帶入的使用者 login\_id / doamin 進行推播 |
| 2018-12-27 | Darren.K.Ti | 1.5 | Function | 1. API - setStatus   * Request Body 的下列資料，移到最上層   + login\_id   + doamin   + emp\_no |
| 2019-01-07 | Darren.K.Ti | 1.6 | Function | 1. API - setStatus   * 移除錯誤代碼 000938 的檢查流程，改為使用 999001 * 修改   + UIFlow   + 程式碼修改記錄 |
| 2019-01-08 | Darren.K.Ti | 1.7 | Function | 1. API - setStatus   * 新增選填參數 note : 備註文字 * 修改   + 程式碼修改記錄   2. API - getStatus   * 新增回傳資料 note : 備註文字 * 修改   + 程式碼修改記錄 |
| 2019-01-15 | Darren.K.Ti | 1.8 | Function | 1. 新增 API - deleteReserve  2. 新增 API - editReserve  3. 新增 API - setEmpServicePush  4. 修改 API - getEmpServiceList   * 新增回傳資料節點   + push\_list : 哪些人接收此 Service ID 的推播通知   5. 修改 API - newReserve   * 回應資料 - 移除   + info\_push\_title   + info\_push\_content * 回應資料 - 新增   + info\_push\_admin\_title   + info\_push\_admin\_content   + info\_push\_emp\_title   + info\_push\_emp\_content * 新增預約時，Server 端依據 Table `service\_push` 決定要發送推播給誰   6. 修改 API - getReserveRecord   * Request Body 資料 - 移除   + info\_push\_title   + info\_push\_content * Request Body 資料 - 新增   + info\_push\_admin\_title   + info\_push\_admin\_content   + info\_push\_emp\_title   + info\_push\_emp\_content   7. 修改 API - getTargetReserveData   * 回應資料 - 移除   + info\_push\_title   + info\_push\_content * 回應資料 - 新增   + info\_push\_admin\_title   + info\_push\_admin\_content   + info\_push\_emp\_title   + info\_push\_emp\_content   8. 修改 API - getMyReserve   * 回應資料 - 移除   + info\_push\_title   + info\_push\_content * 回應資料 - 新增   + info\_push\_admin\_title   + info\_push\_admin\_content   + info\_push\_emp\_title   + info\_push\_emp\_content   9. 修改 API - newEmpService   * 新增資料時，另外將 Ownser 寫入 Table `service\_push`   10. 修改 API - deleteEmpService   * 刪除資料時，另外刪除 Table `service\_push` 相對應的資料 |

# 目錄

[**目錄**](#_30j0zll) **2**

[**1.0 系統設計 (System Design)**](#_3znysh7) **3**

[**2.0 資料庫設計 (Database Design)**](#_tyjcwt) **3**

[**3.0 Implementation**](#_3dy6vkm) **4**

[**4.0 功能修改記錄**](#_1t3h5sf) **5**

[[ EmpService API & Status API - 應用於 ( 茶水預約 ) ]](#_4d34og8) 5

[4.1 需求說明](#_sh3w8xzthq6) 5

[4.2 時程評估 (依照開發順序)](#_u8h9nl3u7gm5) 7

[4.3 對應文件](#_n13lnemcda1u) 7

[4.4 修改項目](#_yppwycepxn3w) 7

[4.5 功能架構圖](#_fftjyhwykpuh) 8

[4.6 資料庫物件清單](#_lzmsqs8dyqwn) 10

[4.7 實體關聯圖](#_k9qelncd1e7h) 11

[4.8 資料表結構](#_egrqds71as4x) 12

[4.9 公用元件](#_q7h41ae7amg6) 12

[4.10 共用介面](#_9lwuatinxq6n) 12

[4.11 資料流程圖 Data Flow Diagram](#_bshrfzvgoch1) 12

[4.12 程式碼修改記錄](#_tbufotq7rlf3) 12

[4.14 Test Case](#_iluhyr3tobf) 13

# 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

## [ EmpService API & Status API - 應用於 ( 茶水預約 ) ]

### 4.1 需求說明

* **使用以下 API 達成 ( 茶水預約 ) 的功能 :**
  + **EmpService API**
  + **Status API**
  + **QForum API + QStorage API**
  + **Push API**
* **Phase 1 ( 2018 年 )**
  + **[ 茶水預約 - 總機 ]**
    - **Status API**
      * **總機服務狀態 (** 新增 / 設定 **)**
    - **EmpService API**
      * **取得目前所有被設定的會議室列表，設定可以預約茶水的會議室 (** 新增 / 刪除 **)**
        + **BQT/QTT，QTY，雙星，QTH，BMT(明基三豐)，Lily(怡安)**
        + **依據 RRS API - ListAllMeetingRoom 回傳的資料，兩者做比對，讓 APP 端可以判定，目前哪些會議室可以被選擇，哪些無法被選擇**
        + **承上，如果是自已設定的會議室，可以進行移除**
      * **查詢預約記錄 (** 本日 / 明日 **)**
      * **設定茶水預約服務處理完成**
    - **QForum API + QStorage API**
      * **服務公告 (** 新增 / 編輯 / 刪除 **)**
        + **比照 ENS 使用 QForum + QStorage 來實作功能**
  + **[ 茶水預約 - 員工 ]**
    - **Status API**
      * **總機服務狀態 (** 查詢 **)**
    - **EmpService API**
      * **取得支援茶水預約的會議室列表**
      * **查詢單一個會議室，前後一小時的預約狀況**
      * **我的預約記錄 ( 本日 ~ 往後 14 日內 )**
    - **EmpService API + Push API**
      * **新增預約 + 推播**
        + **即時預約 和 排成預約 共用同一個設定流程**
* **Phase 2 ( 2019 年 )**
  + **[ 茶水預約 - 總機 ]**
    - **QForum API + QStorage API**
      * **會議室問題反應 (** 查詢 **)**
        + **比照 ENS 使用 QForum + QStorage 來實作功能**
  + **[ 茶水預約 - 員工 ]**
    - **QForum API + QStorage API**
      * **會議室問題反應 (** 新增 **)**
        + **比照 ENS 使用 QForum + QStorage 來實作功能**
    - **排程預約 (** 新增 / 編輯 / 刪除 **) + 推播**
* **MySQL** 
  + **新增 EmpService 相關 DB 和 Table**
    - **DB `EmpService`**
      * **Table**
        + **`service\_id`**
        + **`target\_id`**
        + **`reserve\_record`**
        + **`data\_log`**
        + **`reserve\_limit\_target`**
  + **DB `qplay` 新增 Status 相關 Table**
    - **Table**
      * **`status\_id`**
      * **`status\_life\_crontab`**
      * **`status\_log`**
* **API 新增**
  + **EmpService 相關 API ( Phase 1 )**
    - **服務人員**
      * **API - newEmpService** ( 新增服務 )
        + Post : Service\_ID / Service\_Type。
        + Return : 設定結果。
      * **API - deleteEmpService** ( 刪除服務 )
        + Post : Service\_ID。
        + Return : 設定結果。
        + 備註 : 刪除的同時，與其關聯的負責目標，也會移除關聯。
      * **API - setEmpServiceTarget** ( 設定 服務 和 負責目標 的關聯資料 [新增 / 移除] )
        + Post : Service\_ID / Target\_Data / Action (add & remove) 。
        + Return : 設定結果。
      * **API - getReserveRecord** ( 取得指定服務，其負責目標，在指定時間範圍的預約記錄，以及處理狀態 )
        + Post : Service\_ID / Start\_Date / End\_Date。
        + Return : 預約記錄以及 Complete ( Y or N )，每個 Reserve 有 Unique ID。
    - **一般員工**
      * **API - newReserve** ( 新增預約服務 )
        + Post : Target\_ID / Start\_Date / End\_Date。
        + Return : 預約結果。
      * **API - getTargetReserveData** ( 取得指定目標，在指定時間範圍的預約記錄 )
        + Post : Target\_ID / Start\_Date / End\_Date。
        + Return : ex => 單一個會議室，前後一小時的預約狀況。
      * **API - getMyReserve** ( 取得指定服務，在指定時間範圍，我的預約紀錄)
        + Post : Service\_ID / Service\_Type / Start Date / End Date。
        + Return : ex => 取得我在 14 天之內的茶水預約記錄。
    - **服務人員 / 一般員工**
      * **API - getEmpServiceList** ( 取得服務列表 )
        + Post : Service\_Type。
        + Return : 服務列表。
      * **API - getEmpServiceTargetList** ( 取得 服務 和 負責目標 的關聯資料 )
        + Post : Service\_ID / Service\_Type。
        + Return : ex => 取得茶水預約所有服務人員 - 負責會議室的資料。
      * **API - setReserveComplete** ( 設定預約服務處理完成 )
        + Post : Reserve\_ID。
        + Return : 設定結果。
        + 備註 : 預約服務處理完成，可由服務人員 or 員工自己，個別設定。
  + **Status 相關 API ( Phase 1 )**
    - **服務人員**
      * **API - setStatus** ( 新增 / 修改 設定 [ 指定功能 ] 的狀態 )
        + Post : Status\_ID / Status\_Type / Life Setting / Period Setting / Status。
        + Return : 設定結果。
      * **API - deleteStatus** ( 刪除 [ 指定功能 ] 的狀態 )
        + Post : Status\_ID or Status\_Type。
        + Return : 設定結果。
    - **服務人員 / 一般員工**
      * **API - getStatus** ( 取得 [ 指定功能 ] 的狀態 )
        + Post : Status\_ID or Status\_Type。
        + Return : 資料列表。
  + **EmpService 相關 API ( Phase 2 )**
    - **一般員工**
      * **API - deleteReserve** ( 刪除預約服務 )
        + Post : Reserve\_ID。
        + Return : 處理結果。
      * **API - editReserve** ( 編輯預約服務 )
        + Post : Reserve\_ID / Target\_ID。
        + Return : 處理結果。
      * **API - setEmpServicePush** ( 設定推播管理)
        + Post : Service\_ID。
        + Return : 處理結果。

### 4.2 時程評估 (依照開發順序)

* Phase 1
* [ **API - newEmpService** ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
  + 備註 : 包含開發 [ 紀錄 Log Function ] 的時間
* [ **API - setEmpServiceTarget** ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ **API - getEmpServiceList** ]
  + 預估時程 : ? ~ ? (1 day)
  + 實際時程 :
* [ **API - getEmpServiceTargetList** ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ **API - newReserve** ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
  + 備註 :
    - 包含處理推播，先預設 Target 不會過期且沒有額度限制
    - 不包含判斷 Target 是否過期 / 額滿
* [ **API - getReserveRecord** ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ **API - getTargetReserveData** ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ **API - getMyReserve** ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ **API - setReserveComplete** ]
  + 預估時程 : ? ~ ? (1 day)
  + 實際時程 :
* [ **API - setStatus** ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ **API - getStatus** ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
* [ **API - deleteEmpservice** ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
  + 備註 : 刪除 Service，與其關聯的 Target 也需要關閉 ( 設定 active=N )，讓其它 Service 可以選擇此 Target 做設定
* [ **API - deleteStatus** ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
  + 備註 : 刪除 Status，與其關聯的 Life\_Period 也需要停用 ( 設定 active=N )
* Phase 2
* [ **API - deleteReserve** ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ **API - editReserve** ]
  + 預估時程 : ? ~ ? (2 day)
  + 實際時程 :
* [ **API - setEmpServicePush** ]
  + 預估時程 : ? ~ ? (3.5 day)
  + 實際時程 :
* [ **API - newEmpService** ]
  + 預估時程 : ? ~ ? (1 day)
  + 實際時程 :
  + 備註 : 新增 EmpService 的時候，將 Owner 寫入 Service Push 推播設定
* [ **API - deleteEmpService** ]
  + 預估時程 : ? ~ ? (1 day)
  + 實際時程 :
  + 備註 : 刪除 EmpService，與其關聯的 Service Push 推播設定，也要刪除
* [ **API - getEmpServiceList** ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
  + 備註 : 新增資料節點 push\_list，顯示此 Service 的推播設定有哪些成員
* [ **API - newReserve** ]
  + 預估時程 : ? ~ ? (1.5 day)
  + 實際時程 :
  + 備註 :
    - 新增預約，依據 Service Push 的推播設定，決定要發送推播給哪些成員
    - 修改推播的資料參數
* [ **API - getReserveRecord** ]
  + 預估時程 : ? ~ ? (0.5 day)
  + 實際時程 :
  + 備註 :
    - 修改回傳資料的推播內容
* [ **API - getTargetReserveData** ]
  + 預估時程 : ? ~ ? (0.5 day)
  + 實際時程 :
  + 備註 :
    - 修改回傳資料的推播內容
* [ **API - getMyReserve** ]
  + 預估時程 : ? ~ ? (0.5 day)
  + 實際時程 :
  + 備註 :
    - 修改回傳資料的推播內容

### 4.3 對應文件

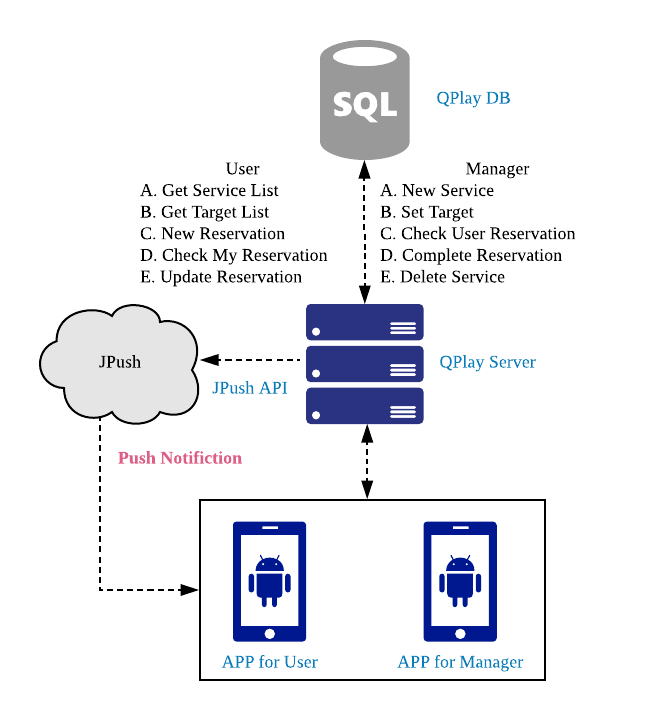
* EmpService\_API\_1.1\_20190115.docx
* Status\_API\_1.4\_20190117.docx
* QPlay\_DB\_Schema\_3.22\_20190108.docx
* EmpService\_DB\_Schema\_1.1\_20190115.docx

### 4.4 修改項目

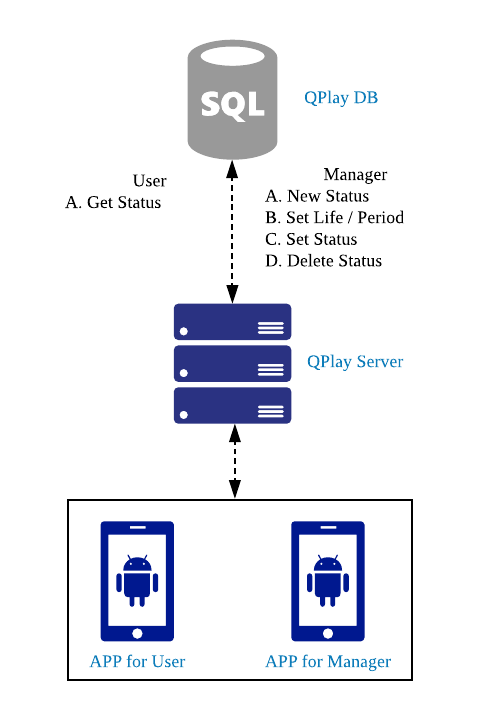
* DB `**qplay**`
  + 新增 Table
    - **`status\_id`**
    - **`status\_life\_crontab`**
    - **`status\_log`**
* 新增 DB `**EmpService**`
  + 新增 Table
    - **`service\_id`**
    - **`target\_id`**
    - **`reserve\_record`**
    - **`data\_log`**
    - **`reserve\_limit\_target`**
    - **`service\_push`**
* qplayApi/
  + 修改
    - **.env**
    - app/ Http/ **routes.php**
    - config/ **database.php**
    - app/ lib/
      1. **ResultCode.php**
      2. **Error.json ( 用於 QPlay 後台 > APP 管理 > 錯誤代碼 )**
  + 新增
    - app/ Http/ Controllers/
      1. Service/
         * **ServiceController.php**
         * **TargetController.php**
         * **ReserveController.php**
      2. Status/
         * **StatusController.php**
    - app/ Model/
      1. **EmpService\_Service\_ID.php**
      2. **EmpService\_Target\_ID.php**
      3. **EmpService\_Reserve\_Record.php**
      4. **EmpService\_Data\_Log.php**
      5. **EmpService\_Reserve\_Limit\_Target.php**
      6. **EmpService\_Service\_Push.php**
      7. **Status\_ID.php**
      8. **Status\_Life\_Crontab.php**
      9. **Status\_Log.php**
    - app/ Service/
      1. **EmpServiceServiceService.php**
      2. **EmpServiceTargetService.php**
      3. **EmpServiceReserveService.php**
      4. **EmpServiceLogService.php**
      5. **StatusService.php**
      6. **StatusLogService.php**
    - app/ Repositories/
      1. **EmpServiceServiceIDRepository.php**
      2. **EmpServiceTargetIDRepository.php**
      3. **EmpServiceReserveRecordRepository.php**
      4. **EmpServiceDataLogRepository.php**
      5. **EmpServiceReserveLimitTargetRepository.php**
      6. **EmpServiceServicePushRepository.php**
      7. **StatusIDRepository.php**
      8. **StatusLifeCrontabRepository.php**
      9. **StatusLogRepository.php**

### 4.5 功能架構圖

**Service API**



**Status API**

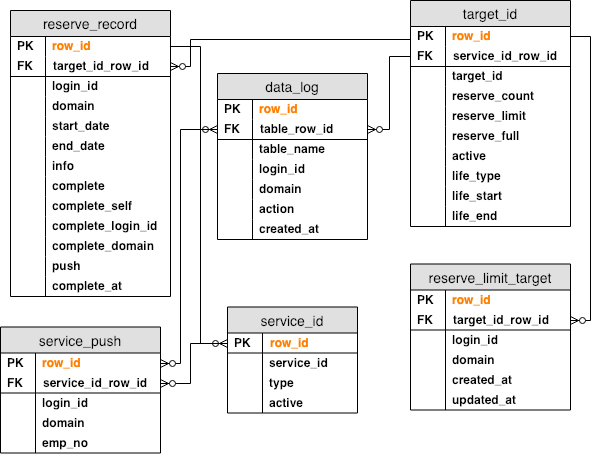


### 4.6 資料庫物件清單

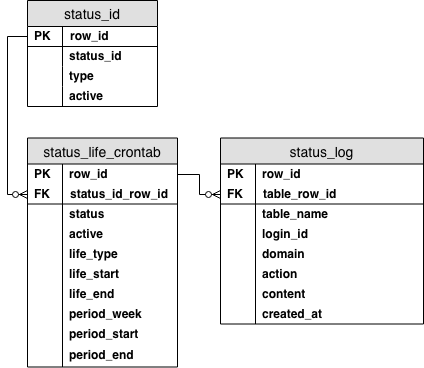
|  |  |  |  |
| --- | --- | --- | --- |
| **DB Type** | **Type** | **Object Name** | **Description** |
| MySQL | InnoDB | service\_id | 紀錄 service\_id 和 type |
| MySQL | InnoDB | target\_id | 紀錄 target\_id，以及生命週期和人數限制及統計。 |
| MySQL | InnoDB | reserve\_record | 員工的預約記錄 |
| MySQL | InnoDB | data\_log | service\_id / target\_id / reserve\_record 的操作記錄，紀錄 [ 新增 ] or [ 更新 ] or [ 刪除 ] 此資料的使用者的 login\_id 和 domain。 |
| MySQL | InnoDB | reserve\_limit\_target | 當 Target 有數量限制，使用此 Table 做紀錄 |
| MySQL | InnoDB | service\_push | Service 的推播管理 |
| MySQL | InnoDB | status\_id | 各種功能的 Status ID |
| MySQL | InnoDB | status\_life\_crontab | Status ID 的生命週期和期效 |
| MySQL | InnoDB | status\_log | Status ID 的操作記錄 |

### 4.7 實體關聯圖

**Service API**



**Status API**



### 4.8 資料表結構

* 參考文件 :
  + QPlay\_DB\_Schema\_3.22\_20190108.docx
  + EmpService\_DB\_Schema\_1.1\_20190115.docx

### 4.9 公用元件

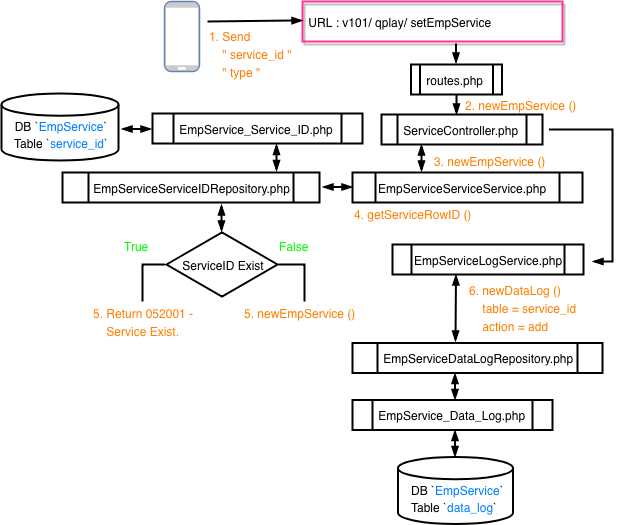
* 內部模組

### 4.10 共用介面

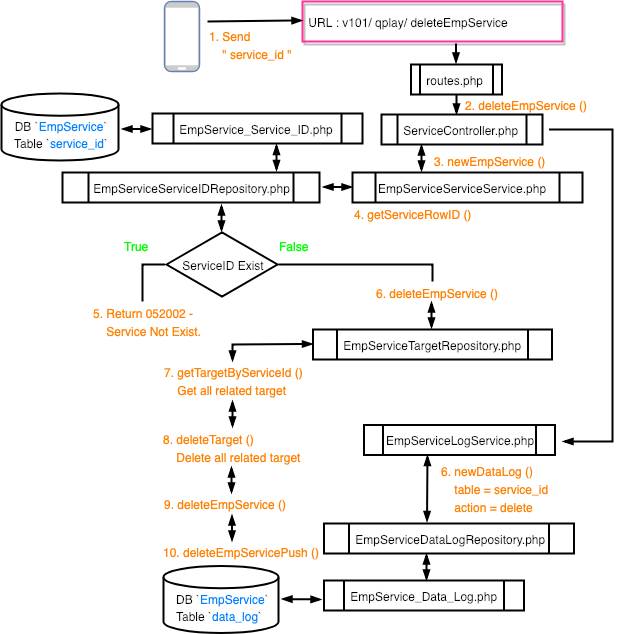
* EmpService\_API\_1.0\_20190115.docx
* Status\_API\_1.4\_20190117.docx

### 4.11 資料流程圖 Data Flow Diagram

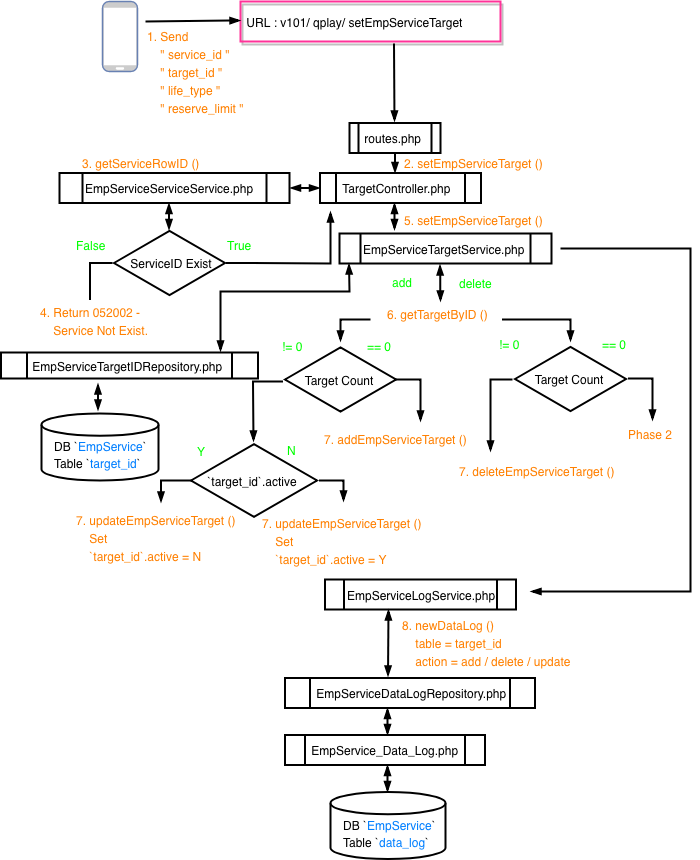
#### API - newEmpService



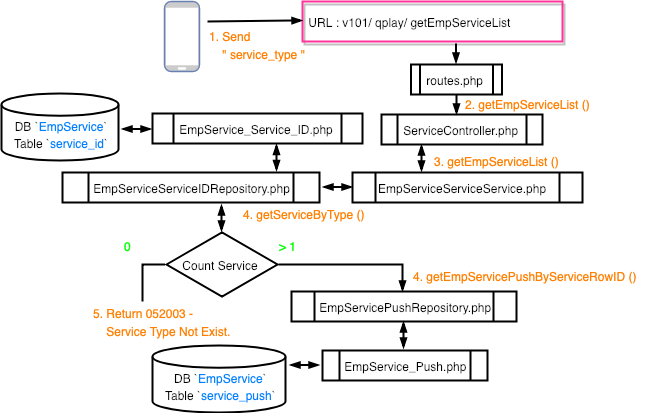
#### API - deleteEmpService



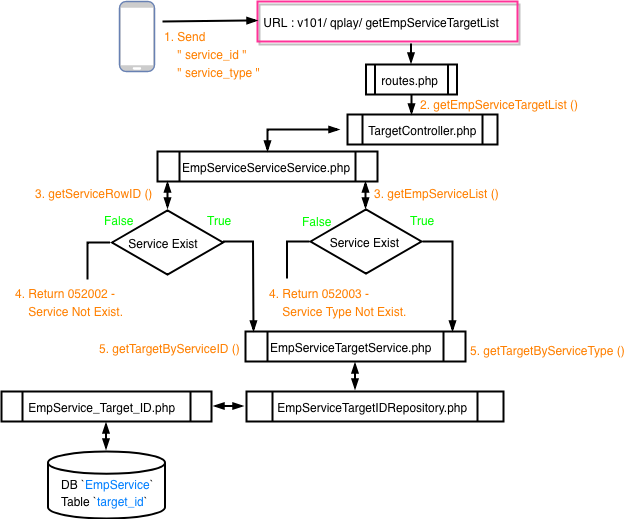
#### API - setEmpServiceTarget



#### API - getEmpServiceList



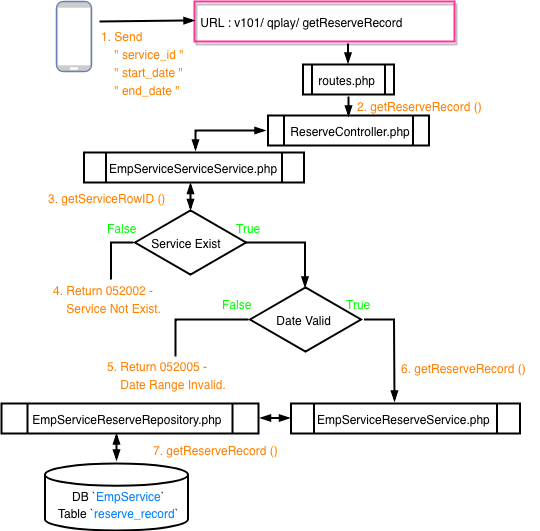
#### API - getEmpServiceTargetList



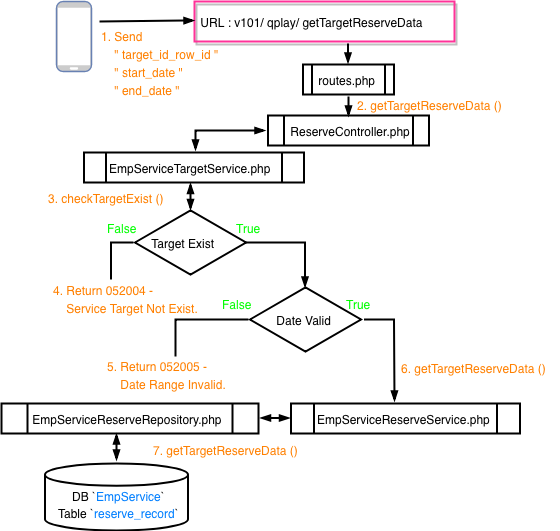
#### API - newReserve



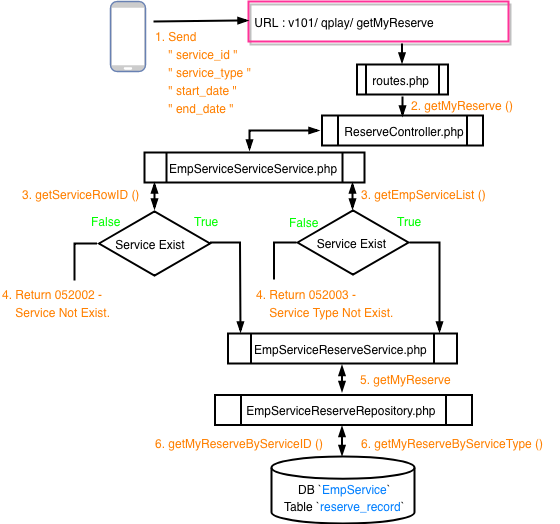
#### API - getReserveRecord



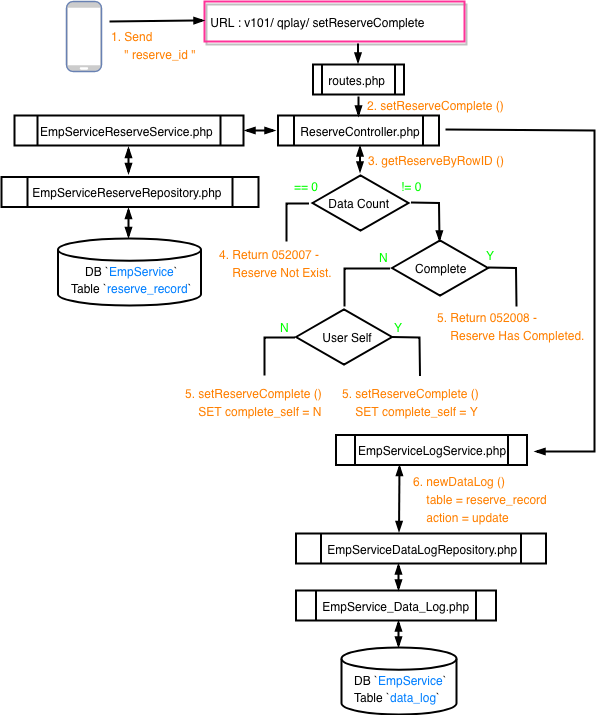
#### API - getTargetReserveData



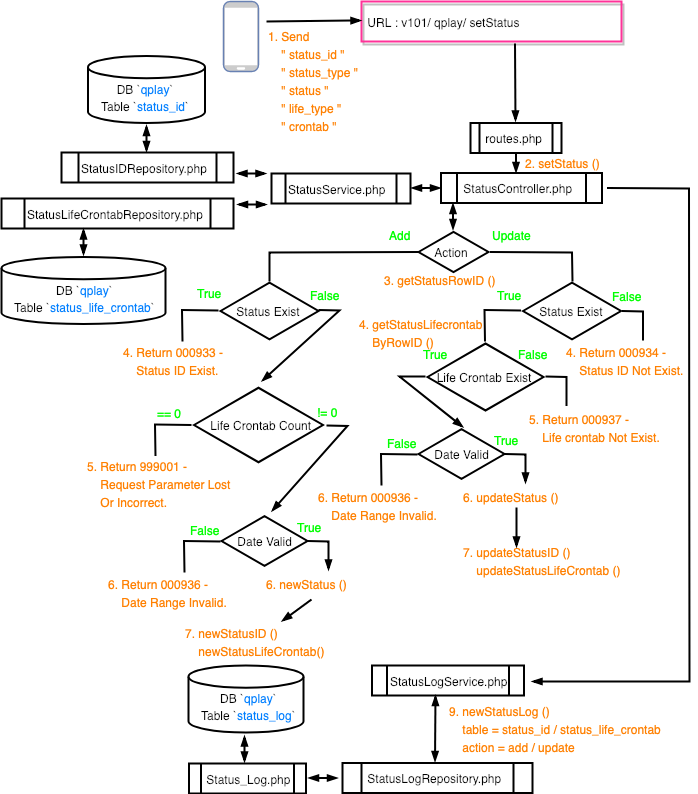
#### API - getMyReserve



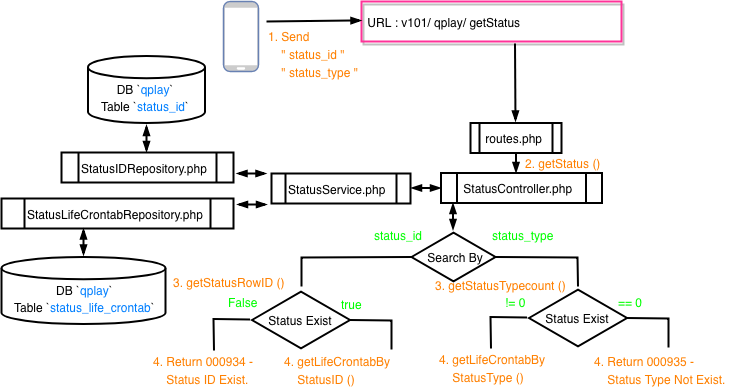
#### API - setReserveComplete



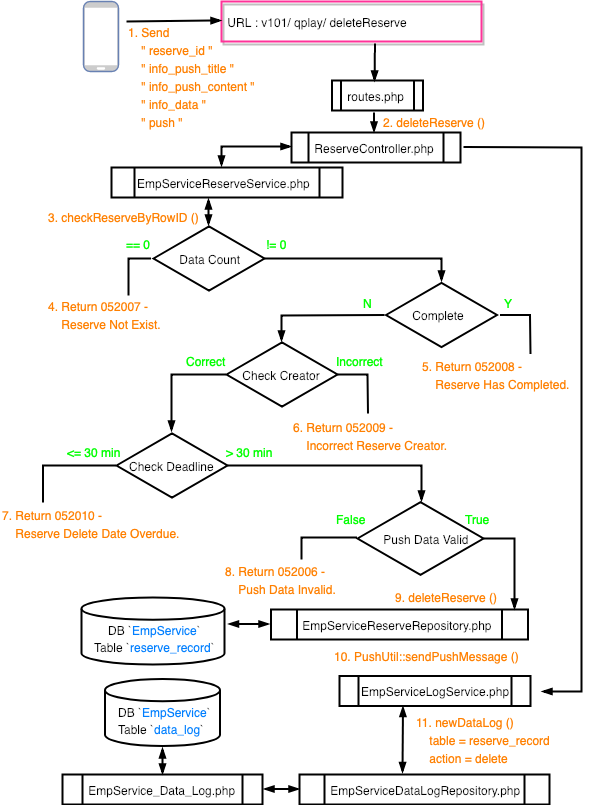
#### API - setStatus



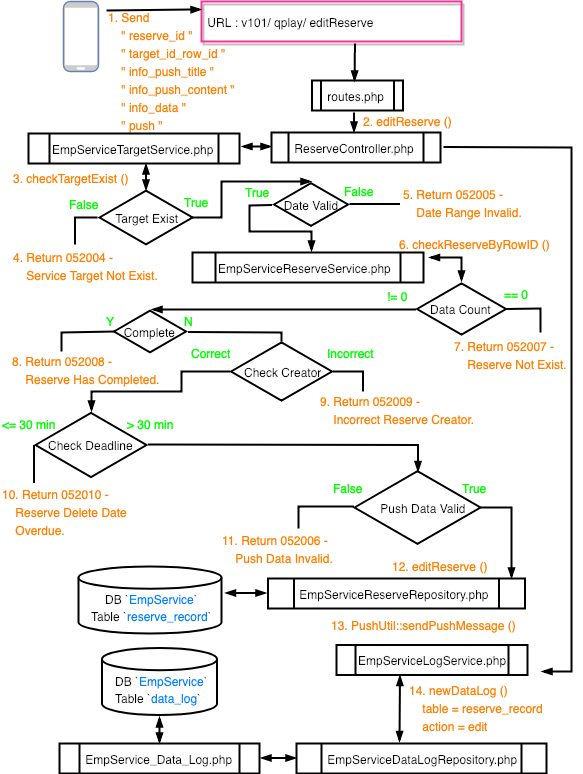
#### API - getStatus



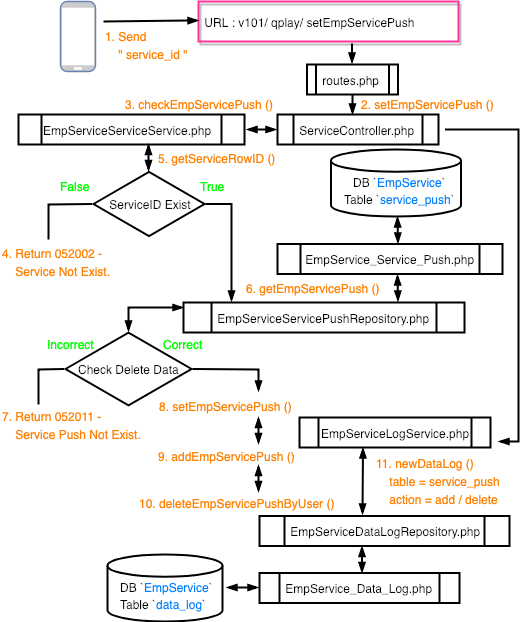
#### API - deleteReserve



#### API - editReserve



#### API - setEmpServicePush



### 4.12 程式碼修改記錄

* **.env** ( 實際參數內容，參考現有環境的 .env )
  + Develop 新增
    - DB\_CONNECTION = mysql\_emp\_service
    - DB\_HOST\_EMP\_SERVICE = \*.\*.\*.\*
    - DB\_PORT\_EMP\_SERVICE = 3306
    - DB\_DATABASE\_EMP\_SERVICE = emp\_service
    - DB\_USERNAME\_EMP\_SERVICE = \*\*\*\*
    - DB\_PASSWORD\_EMP\_SERVICE = \*\*\*\*\*\*\*\*
  + Staging 新增
    - DB\_CONNECTION = mysql\_emp\_service
    - DB\_HOST\_EMP\_SERVICE = \*.\*.\*.\*
    - DB\_PORT\_EMP\_SERVICE = 3306
    - DB\_DATABASE\_EMP\_SERVICE = emp\_service
    - DB\_USERNAME\_EMP\_SERVICE = \*\*\*\*
    - DB\_PASSWORD\_EMP\_SERVICE = \*\*\*\*\*\*\*\*
  + Production 新增
    - DB\_CONNECTION = mysql\_emp\_service
    - DB\_HOST\_EMP\_SERVICE = \*.\*.\*.\*
    - DB\_PORT\_EMP\_SERVICE = 3306
    - DB\_DATABASE\_EMP\_SERVICE = emp\_service
    - DB\_USERNAME\_EMP\_SERVICE = \*\*\*\*\*
    - DB\_PASSWORD\_EMP\_SERVICE = \*\*\*\*\*\*\*\*
* config/ **database.php**
  + 新增設定
    - 'mysql\_emp\_service' => [

'driver' => 'mysql',

'host' => env('DB\_HOST\_EMP\_SERVICE'),

'port' => env('DB\_PORT\_EMP\_SERVICE'),

'database' => env('DB\_DATABASE\_EMP\_SERVICE'),

'username' => env('DB\_USERNAME\_EMP\_SERVICE'),

'password' => env('DB\_PASSWORD\_EMP\_SERVICE'),

'charset' => 'utf8',

'collation' => 'utf8\_general\_ci',

'prefix' => '',

'strict' => false,

'engine' => null,

]

* app/ Http/ **routes.php**
  + URL Request Handle 新增下列 Route :
    - 使用 EmpService/ ServiceController.php
      * newEmpService，新增 EmpService。
      * getEmpServiceList，取得 EmpService 資料列表。
      * setEmpServicePush，設定 EmpService 的推播管理。
    - 使用 EmpService/ TargetController.php
      * setEmpServiceTarget，新增 EmpService 負責的 Target。
      * getEmpServiceTargetList，取得 Target 資料列表。
    - 使用 EmpService/ ReserveController.php
      * newReserve，新增預約。
      * getReserveRecord，取得所有預約記錄。
      * getTargetReserveData，取得指定 Target 的預約記錄。
      * getMyReserve，取得我自己的預約記錄。
      * setReserveComplete，將指定的預約設定為處理完成。
      * deleteReserve，刪除預約。
      * editReserve，編輯預約。
    - 使用 Status/ StatusController.php
      * setStatus，新增 / 更新狀態。
      * getStatus，取得狀態。
* app/ Http/ Controllers/ EmpService/

#### **ServiceController.php** 定義 function

* + - newEmpService ()
      * 帶入
        + service\_id : required
        + type : required
        + login\_id : required
        + domain : required
        + emp\_no : required
      * 使用EmpServiceServiceService.php 定義的 newEmpService ()。
        + Response Data Array

Array [0] : 回應至 APP 端

Array [1] : 使用EmpServiceLogService.php 定義的 newDataLog ()。

* + - deleteEmpService ()
      * 帶入
        + service\_id : required
        + login\_id : required
        + domain : required
        + emp\_no : required
      * 使用EmpServiceServiceService.php 定義的 getServiceRowID ()。
        + Service ID 不存在 ( IS NULL )

回傳 052002 Service Not Exist

* + - * + Service ID 已經存在 ( IS NOT NULL )

繼續往下執行

* + - * 使用EmpServiceServiceService.php 定義的 deleteEmpService ()。
        + Response Data Array

Array [0] : 回應至 APP 端

Array [1] : 使用EmpServiceLogService.php 定義的 newDataLog ()。

* + - getEmpServiceList ()
      * 帶入
        + service\_type : required
      * 使用EmpServiceServiceService.php 定義的 getEmpServiceList ()。
    - setEmpServicePush ()
      * 帶入
        + service\_id : required
        + add : optional
        + delete : optional
        + login\_id : required
        + domain : required
        + emp\_no : required
      * 使用 EmpServiceServiceService.php 定義的 checkEmpServicePush () 檢查
        + 如果驗證失敗

回傳 錯誤代碼

* + - * + 如果驗證成功

取得回傳的資料

add\_service\_row\_id : array

delete\_service\_row\_id : array

繼續往下執行

* + - * 使用EmpServiceServiceService.php 定義的 setEmpServicePush ()。
        + Response Data Array

Array [0] : 回應至 APP 端

Array [1] : 使用EmpServiceLogService.php 定義的 newDataLog ()。

#### **TargetController.php** 定義 function

* + - setEmpServiceTarget ()
      * 帶入
        + service\_id : required
        + login\_id : required
        + domain : required
        + emp\_no : required
        + add [

target\_id : required,

life\_type : required | numeric,

reserve\_limit : required | numeric

]

* + - * + delete [

target\_id : required

]

* + - * 使用EmpServiceServiceService.php 定義的 getServiceRowID ()。
        + Service ID 不存在 ( IS NULL )

回傳 052002 Service Not Exist

* + - * + Service ID 已經存在 ( IS NOT NULL )

繼續往下執行

* + - * 使用EmpServiceTargetService.php 定義的 setEmpServiceTarget ()。
        + Response Data Array

Array [0] : 回應至 APP 端

Array [1] : 使用EmpServiceLogService.php 定義的 newDataLog ()。

* + - getEmpServiceTargetList ()
      * 帶入
        + service\_id : required ( 與 service\_type 擇一)
        + service\_type : required ( 與 service\_id 擇一)
      * 依據查詢條件
        + service\_id

使用EmpServiceServiceService.php 定義的 getServiceRowID ()。

Service ID 不存在 ( IS NULL )

回傳 052002 Service Not Exist

Service ID 已經存在 ( IS NOT NULL )

取得 Service ID 的 Row ID，繼續往下執行

使用EmpServiceTargetService.php

定義的 getTargetByServiceID ()

* + - * + service\_type

使用EmpServiceServiceService.php 定義的 getEmpServiceList ()。

如果資料筆數 == 0

回傳 052003 Service Type Not Exist

如果資料筆數 != 0

繼續往下執行

使用EmpServiceTargetService.php

定義的 getTargetByServiceType ()

#### **ReserveController.php** 定義 function

* + - newReserve ()
      * 帶入
        + target\_id\_row\_id : required | numeric
        + login\_id : required
        + domain : required
        + emp\_no : required
        + start\_date : required | numeric | digits:10
        + end\_date : required | numeric | digits:10
        + info\_push\_admin\_title : required
        + info\_push\_admin\_content : required
        + info\_push\_emp\_title : required
        + info\_push\_emp\_content : required
        + info\_data : required
        + push : required
      * 使用 EmpServiceTargetService.php 定義的 getTargetByRowID ()
        + 資料筆數 == 0

回傳 052004 Service Target Not Exist

* + - * + 資料筆數 != 0

繼續往下執行

* + - * 檢查 start\_date 是否 <= end\_date
        + 否

回傳 052005 Date Range Invalid

* + - * + 是

繼續往下執行

* + - * 檢查 push 的資料格式是否在陣列之內
        + [ “00”, “01”, “10”, “11” ]
        + 否

回傳 052006 Push Data Invalid

* + - * + 是

繼續往下執行

* + - * 使用EmpServiceServiceService.php

定義的 getServiceManagerByTarget ()

* + - * + 帶入 target\_id\_row\_id
        + 回傳 Service Manager 的

login\_id

domain

* + - * 使用EmpServiceReserveService.php 定義的 newReserve ()。
        + Response Data Array

Array [0] : 回應至 APP 端

Array [1] : 使用EmpServiceLogService.php 定義的 newDataLog ()。

* + - getReserveRecord ()
      * 帶入
        + service\_id : required
        + start\_date : required | numeric | digits:10
        + end\_date : required | numeric | digits:10
      * 使用EmpServiceServiceService.php 定義的 getServiceRowID ()。
        + Service ID 不存在 ( IS NULL )

回傳 052002 Service Not Exist

* + - * + Service ID 已經存在 ( IS NOT NULL )

取得 Service ID 的 Row ID，繼續往下執行

* + - * 檢查 start\_date 是否 <= end\_date
        + 否

回傳 052005 Date Range Invalid

* + - * + 是

繼續往下執行

* + - * 使用EmpServiceReserveService.php 定義的 getReserveRecord ()。
    - getTargetReserveData ()
      * 帶入
        + target\_id\_row\_id : required | numeric
        + start\_date : required | numeric | digits:10
        + end\_date : required | numeric | digits:10
      * 使用 EmpServiceTargetService.php 定義的 getTargetByRowID ()
        + 資料筆數 == 0

回傳 052004 Service Target Not Exist

* + - * + 資料筆數 != 0

繼續往下執行

* + - * 檢查 start\_date 是否 <= end\_date
        + 否

回傳 052005 Date Range Invalid

* + - * + 是

繼續往下執行

* + - * 使用EmpServiceReserveService.php 定義的 getTargetReserveData ()。
    - getMyReserve ()
      * 帶入
        + service\_id : required ( 與 service\_type 擇一)
        + service\_type : required ( 與 service\_id 擇一)
        + login\_id : required
        + domain : required
        + emp\_no : required
        + start\_date : required | numeric | digits:10
        + end\_date : required | numeric | digits:10
      * 依據查詢條件
        + service\_id

使用EmpServiceServiceService.php 定義的 getServiceRowID ()。

Service ID 不存在 ( IS NULL )

回傳 052002 Service Not Exist

Service ID 已經存在 ( IS NOT NULL )

取得 Service ID 的 Row ID，繼續往下執行

使用EmpServiceReserveService.php 定義的 getMyReserveByServiceID ()。

* + - * + service\_type

使用EmpServiceServiceService.php 定義的 getEmpServiceList ()。

如果資料筆數 == 0

回傳 052003 Service Type Not Exist

如果資料筆數 != 0

繼續往下執行

使用EmpServiceReserveService.php 定義的 getMyReserveByServiceType ()。

* + - setReserveComplete ()
      * 帶入
        + reserve\_id : required | numeric
        + login\_id : required
        + domain : required
        + emp\_no : required
      * 使用EmpServiceReserveService.php 定義的 getReserveByRowID ()。
        + 如果資料筆數 == 0

回傳 052007 Reserve Not Exist

* + - * + 如果資料筆數 != 0

如果 complete == Y

回傳 052008 Reserve Has Completed

如果 complete == N

如果是員工本人

complete\_self = Y

如果不是員工本人

complete\_self = N

* + - * 使用EmpServiceReserveService.php 定義的 setReserveComplete () 帶入 complete\_self。
        + Response Data Array

Array [0] : 回應至 APP 端

Array [1] : 使用EmpServiceLogService.php 定義的 newDataLog ()。

* + - deleteReserve ()
      * 帶入
        + reserve\_id : required | numeric
        + login\_id : required
        + domain : required
        + emp\_no : required
        + info\_push\_admin\_title : required
        + info\_push\_admin\_content : required
        + info\_push\_emp\_title : required
        + info\_push\_emp\_content : required
        + push : required
      * 檢查 push 的資料格式是否在陣列之內
        + [ “00”, “01”, “10”, “11” ]
        + 否

回傳 052006 Push Data Invalid

* + - * + 是

繼續往下執行

* + - * 使用EmpServiceReserveService.php 定義的 checkReserveByRowID ()。
        + 如果驗證失敗

回傳 錯誤代碼

* + - * + 如果驗證成功

繼續往下執行

* + - * 使用EmpServiceReserveService.php 定義的 getReserveByRowID ()

取得 `reserve\_record`.target\_id\_row\_id。

* + - * 使用EmpServiceTragetService.php 定義的 getTargetByRowID ()

取得 `target\_id`.service\_id\_row\_id。

* + - * 使用EmpServiceReserveService.php 定義的 deleteReserve ()。
    - editReserve ()
      * 帶入
        + reserve\_id : required | numeric
        + target\_id\_row\_id : required | numeric
        + login\_id : required
        + domain : required
        + emp\_no : required
        + start\_date : required | numeric | digits:10
        + end\_date : required | numeric | digits:10
        + info\_push\_admin\_title : required
        + info\_push\_admin\_content : required
        + info\_push\_emp\_title : required
        + info\_push\_emp\_content : required
        + info\_data : required
        + push : required
      * 使用 EmpServiceTargetService.php 定義的 getTargetByRowID ()
        + 資料筆數 == 0

回傳 052004 Service Target Not Exist

* + - * + 資料筆數 != 0

繼續往下執行

* + - * 檢查 start\_date 是否 <= end\_date
        + 否

回傳 052005 Date Range Invalid

* + - * + 是

繼續往下執行

* + - * 檢查 push 的資料格式是否在陣列之內
        + [ “00”, “01”, “10”, “11” ]
        + 否

回傳 052006 Push Data Invalid

* + - * + 是

繼續往下執行

* + - * 使用EmpServiceReserveService.php 定義的 checkReserveByRowID ()。
        + 如果驗證失敗

回傳 錯誤代碼

* + - * + 如果資料成功

繼續往下執行

* + - * 使用EmpServiceTragetService.php 定義的 getTargetByRowID ()

取得 `target\_id`.service\_id\_row\_id。

* + - * 使用EmpServiceReserveService.php 定義的 editReserve ()。
* app/ Http/ Controllers/ Status/

#### **StatusController.php** 定義 function

* + - setStatus ()
      * 開啟 Transaction
      * 流程 1.
        + 最上層的節點 login\_id / domain / emp\_no，取得員工資料
      * 流程 2.
        + 最上層的節點 status\_new / status\_update，分開為兩個流程個別處理
      * 流程 3. - status\_new
        + 使用 foreach () 處理內層的資料

使用StatusService.php 定義的 getStatusRowID ()。

Status ID 已經存在 ( IS NOT NULL )

回傳 000933 Status ID Exist

Status ID 不存在 ( IS NULL )

取得 Status ID 的 Row ID，繼續往下執行

* + - * + 使用StatusService.php 定義的 newStatus()。

帶入

status\_id : required

status\_type : required

period\_list : required

* + - * + Response Data Array

Array [0] : 回應至 APP 端

Array [1] : 使用StatusLogService.php 定義的 newStatusLog ()。

* + - * 流程 4. - status\_update
        + 使用 foreach () 處理內層的資料

使用StatusService.php 定義的 getStatusRowID ()。

Status ID 不存在 ( IS NULL )

回傳 000934 Status ID Not Exist

Status ID 已經存在 ( IS NOT NULL )

取得 Status ID 的 Row ID，繼續往下執行

* + - * + 使用StatusService.php 定義的 updateStatus()。

帶入

period\_list : required

* + - * + Response Data Array

Array [0] : 回應至 APP 端

Array [1] : 使用StatusLogService.php 定義的 newStatusLog ()。

* + - * 備註 :
        + status\_new / status\_update 只要任何一個流程有 error，

就判定 Transaction rollBack()

* + - * + status\_new / status\_update 每一個流程都 OK，才回傳 Success 至 APP 端
    - getStatus ()
      * 帶入
        + status\_id : required ( 與 status\_type 擇一)
        + status\_type : required ( 與 status\_id 擇一)
      * 依據查詢條件
        + status\_id

使用StatusService.php 定義的 getStatusRowID ()。

Status ID 不存在 ( IS NULL )

回傳 000934 Status Not Exist

Status ID 已經存在 ( IS NOT NULL )

取得 Service ID 的 Row ID，繼續往下執行

使用StatusService.php

定義的 getLifeCrontabByStatusID ()

* + - * + status\_type

使用StatusService.php 定義的 getStatusTypeCount ()。

如果資料筆數 == 0

回傳 000935 Status Type Not Exist

如果資料筆數 != 0

繼續往下執行

使用StatusService.php

定義的 getLifeCrontabByStatusType ()

* app/ Model/
  + **EmpService\_Service\_ID.php** :
    - 定義 DB `empService` Table `service\_id`。
    - 需要設定 public $timestamp = false;
  + **EmpService\_Target\_ID.php** :
    - 定義 DB `empService` Table `target\_id`。
    - 需要設定 public $timestamp = false;
  + **EmpService\_Reserve\_Record.php** :
    - 定義 DB `empService` Table `reserve\_record`。
    - 需要設定 public $timestamp = false;
  + **EmpService\_Data\_Log.php** :
    - 定義 DB `empService` Table `data\_id`。
    - 需要設定 public $timestamp = false;
  + **EmpService\_Reserve\_Limit\_Target.php** :
    - 定義 DB `empService` Table `reserve\_limit\_target`。
  + **EmpService\_Service\_Push.php** :
    - 定義 DB `empService` Table `service\_push`。
    - 需要設定 public $timestamp = false;
  + **Status\_ID.php** :
    - 定義 DB `qplay` Table `status\_id`。
    - 需要設定 public $timestamp = false;
  + **Status\_Life\_Crontab.php** :
    - 定義 DB `qplay` Table `status\_life\_crontab`。
    - 需要設定 public $timestamp = false;
  + **Status\_Log.php** :
    - 定義 DB `qplay` Table `status\_log`。
* app/ Services/

#### **EmpServiceServiceService.php** 定義 Function

* + - getServiceRowID ()
      * 帶入
        + service\_id
      * 使用 EmpServiceServiceIDRepository.php定義的 getServiceRowID ()。
      * 回傳
        + `servie\_id`.row\_id
    - newEmpService ()
      * 帶入
        + service\_id
        + type
        + login\_id
        + domain
        + emp\_no
      * 使用 EmpServiceServiceService.php定義的 getServiceRowID ()。
        + Service ID 已經存在 ( IS NOT NULL )

回傳 052001 Service Exist

* + - * + Service ID 不存在 ( IS NULL )

繼續往下執行

* + - * 使用 EmpServiceServiceIDRepository.php定義的 newEmpService ()。
        + 取得 insertGetId
        + 將新增的資料整理為下列內容，填入變數 $logData

$logData = [

table\_name => service\_id,

table\_row\_id => insertGetId,

action => add,

content => “service\_id:=>service\_id,type:=>type”

]

* + - * 使用 EmpServicePushRepository.php定義的 addEmpServicePush ()。
      * 回傳至 Controller 的資料
        + Array [ $result, $logData ]

$result = [“result\_code” => ResultCode::\_1\_reponseSuccessful, “message” => CommonUtil::getMessageContentByCode(ResultCode::\_1\_reponseSuccessful)]

* + - deleteEmpService ()
      * 帶入
        + service\_id
        + type
      * 使用 EmpServiceTargetIDRepository.php定義的 getTargetByServiceId () 取得所有關聯的 Target。
      * 使用 EmpServiceTargetIDRepository.php定義的 deleteTarget () 刪除所有關聯的 Target。
      * 使用 EmpServiceServiceIDRepository.php定義的 deleteEmpService () 。
        + 取得剛剛更新的 `service\_id`.row\_id
        + 將更新的資料整理為下列內容，填入變數 $logData

$logData = [

table\_name =>service\_id,

table\_row\_id => `service\_id`.row\_id,

action => delete,

content => “active:Y=>N”

]

* + - * 使用 EmpServicePushRepository.php定義的 deleteEmpServicePush () 。
      * 回傳至 Controller 的資料
        + Array [ $result, $logData ]

$result = [“result\_code” => ResultCode::\_1\_reponseSuccessful, “message” => CommonUtil::getMessageContentByCode(ResultCode::\_1\_reponseSuccessful)]

* + - getEmpServiceList ()
      * 帶入
        + service\_type
      * 使用 EmpServiceServiceIDRepository.php定義的 getEmpServiceByType ()。
        + 如果 service\_type == All，參數帶入 NULL
        + 如果 service\_type != All，參數帶入 service\_type
      * 承上，依據回傳資料
        + 如果 getEmpServiceByType () 資料筆數 == 0

回傳 052003 Service Type Not Exist

* + - * + 如果 getEmpServiceByType () 資料筆數 != 0

繼續往下執行

* + - * 承上，依據回傳資料，取得 `service\_id`.row\_id，使用

EmpServicePushRepository.php定義的 getEmpServicePushByServiceRowID ()。

* + - * + 取得指定 Service 的推播設定成員
      * 整合 getEmpServiceByType () 和 getEmpServicePushByServiceRowID () 的資料。
      * 回傳至 Controller 的資料

回傳 Service 資料列表

{

"service\_type\_list" : [{

"service\_type" : "staff",

"service\_id\_list" : [{

"service\_id" : "staff\_BQT",

"owner\_login\_id" : "台北總機" ,

"owner\_domain" : "BenQ",

"owner\_emp\_no" : "1609009",

"push\_list" : [{

"login\_id" : "台北總機",

"domain" : "BenQ",

"emp\_no" : "1609009"

}, {

"login\_id" : "Darren.K.Ti",

"domain" : "Qisda",

"emp\_no" : "123456"

}]

}, {

"service\_id" : "staff\_QTY",

"owner\_login\_id" : "桃園總機" ,

"owner\_domain" : "QGroup",

"owner\_emp\_no" : "1609009",

"push\_list" : [{

"login\_id" : "桃園總機",

"domain" : "QGroup",

"emp\_no" : "1609009"

}, {

"login\_id" : "Cleo.W.Chan",

"domain" : "BenQ",

"emp\_no" : "123455"

}]

}]

}, {

"service\_type" : "movie",

"service\_id\_list" : [{

"service\_id" : "movie\_BQT",

"owner\_login\_id" : "台北電影" ,

"owner\_domain" : "BenQ",

"owner\_emp\_no" : "1609009",

"push\_list" : [{

"login\_id" : "台北電影",

"domain" : "BenQ",

"emp\_no" : "1609009"

}, {

"login\_id" : "Darren.K.Ti",

"domain" : "Qisda",

"emp\_no" : "123456"

}, {

"login\_id" : "Cleo.W.Chan",

"domain" : "BenQ",

"emp\_no" : "123455"

}]

}]

}]

}

* + - getServiceManagerByTarget ()
      * 帶入
        + target\_id\_row\_id
      * 使用 EmpServiceDataLogRepository.php定義的 getServiceManagerByTarget ()。
    - checkEmpServicePush ()
      * 帶入
        + login\_id
        + domain
        + emp\_no
        + add [

service\_id : required

]

* + - * + delete [

service\_id : required

]

* + - * 如果資料節點 add / delete 存在，使用 count() 檢查每個陣列，

是否至少有一筆資料

* + - * + 否

回傳 999001 傳入參數不足

* + - * + 是

繼續往下執行

* + - * 透過 Foreach 檢查 add / delete
        + 宣告暫存資料的參數

add\_service\_row\_id : array

delete\_service\_row\_id : array

* + - * + 使用 EmpServiceServiceService.php定義的 getServiceRowID ()，檢查並且取得每一個 Service ID 的 row\_id。

其中一個 Service ID 不存在 ( IS NULL )

回傳 052002 Service Not Exist

每一個 Service ID 皆存在 ( IS NOT NULL )

將 add / delete 的 `service\_id`.row\_id 分別暫存在

add\_service\_row\_id

delete\_service\_row\_id

繼續往下執行

* + - * 使用 EmpServicePushRepository.php定義的 getEmpServicePushByUser () 取得此 User 的推播設定資料
        + 比對 delete\_service\_row\_id 的資料，是否確實存在。

其中一個 Service ID 的 row\_id 不存在

回傳 052011 Service Push Not Exist

每一個 Service ID 的 row\_id 皆存在

繼續往下執行

* + - * + 比對 add\_service\_row\_id 的資料，是否已經存在 Table `service\_push`。

如果已經存在 Table `service\_push`

從 add\_service\_row\_id 移除重複的資料

* + - * 回傳
        + add\_service\_row\_id : array
        + delete\_service\_row\_id : array
    - setEmpServicePush ()
      * 帶入
        + login\_id
        + domain
        + emp\_no
        + add\_service\_row\_id
        + delete\_service\_row\_id
      * 依序
        + 處理 add\_service\_row\_id

透過 Foreach 使用 EmpServicePushRepository.php定義

的 addEmpServicePush ()

* + - * + 處理 delete\_service\_row\_id

使用 EmpServicePushRepository.php定義

的 deleteEmpServicePushByUser ()

* + - * 回傳
        + 處理結果

#### **EmpServiceLogService.php** 定義 Function

* + - newDataLog ()
      * 帶入
        + login\_id
        + domain
        + emp\_no
        + $logData
      * 流程
        + 透過 $logData 取得

table\_name

table\_row\_id

action

content

* + - * + 接著將所有資料，透過 foreach 處理，

使用 EmpServiceDataLogRepository.php定義的 newDataLog ()。

#### **EmpServiceTargetService.php** 定義 Function

* + - setEmpServiceTarget ()
      * 帶入
        + `service\_id`.row\_id
        + add [

target\_id : required,

life\_type : required | numeric,

reserve\_limit : required | numeric

]

* + - * + delete [

target\_id : required

]

* + - * 檢查 add / delete 陣列內的資料是否符合規格
      * 開啟 Transaction
      * 接著將所有資料，透過 foreach 處理
        + add :

使用 EmpServiceTargetService.php 定義的 getTargetByID ()

資料筆數 == 0

使用 EmpServiceTargetIDRepository.php定義的 addEmpServiceTarget ()

取得 insertGetId

資料筆數 != 0

當前資料 active = Y

使用 EmpServiceTargetIDRepository.php定義的 updateEmpServiceTarget ()

設定 active = N

當前資料 active = N

使用 EmpServiceTargetIDRepository.php定義的 updateEmpServiceTarget ()

設定 active = Y

取得 Updated Row ID

* + - * + delete :

使用 EmpServiceTargetService.php 定義的 getTargetByID ()

資料筆數 == 0

Phase 2 在處理

資料筆數 != 0

使用 EmpServiceTargetIDRepository.php

定義的 deleteTarget ()

取得 Updated Row ID

* + - * + 將新增的資料整理為下列內容，填入變數 $logData

$logData = [

[

table\_name => target\_id,

table\_row\_id => insertGetId,

action => add,

content => “service\_id\_row\_id:=>`service\_id`.row\_id,target\_id:=>target\_id,reserve\_limit:=>0,life\_type:=>0”

],

[

table\_name => target\_id,

table\_row\_id => Updated Row ID,

action => update,

content => “active:=>active”

],

[

table\_name => target\_id,

table\_row\_id => Updated Row ID,

action => delete,

content => “”

]

]

* + - * 回傳至 Controller 的資料
        + Array [ $result, $logData ]

$result = [“result\_code” => ResultCode::\_1\_reponseSuccessful, “message” => CommonUtil::getMessageContentByCode(ResultCode::\_1\_reponseSuccessful)]

* + - getTargetByServiceID ()
      * 帶入
        + `service\_id`.row\_id
      * 使用 EmpServiceTargetIDRepository.php定義的 getTargetByServiceID () 。
      * 回傳至 Controller 的資料
        + 回傳 Target 資料列表
        + {

"service\_type\_list" : [{

"service\_type" : "staff",

"service\_id\_list" : [{

"service\_id" : "staff\_BQT",

"target\_list" : [{

"target\_id" : "BQT/T01",

"target\_id\_row\_id" : 1,

"life\_type" : 0,

"reserve\_limit" : 0,

"owner\_login\_id" : "台北總機" ,

"owner\_domain" : "BenQ",

"owner\_emp\_no" : "1609009"

}, {

"target\_id" : "BQT/T02",

"target\_id\_row\_id" : 2,

"life\_type" : 0,

"reserve\_limit" : 0,

"owner\_login\_id" : "台北總機" ,

"owner\_domain" : "BenQ",

"owner\_emp\_no" : "1609009"

}]

}]

}

}

* + - getTargetByServiceType ()
      * 帶入
        + `service\_id`.type
      * 使用 EmpServiceTargetIDRepository.php定義的 getTargetByServiceType () 。
      * 回傳至 Controller 的資料
        + 回傳 Target 資料列表
        + {

"service\_type\_list" : [{

"service\_type" : "staff",

"service\_id\_list" : [{

"service\_id" : "staff\_BQT",

"target\_list" : [{

"target\_id" : "BQT/T01",

"target\_id\_row\_id" : 1,

"life\_type" : 0,

"reserve\_limit" : 0,

"owner\_login\_id" : "台北總機" ,

"owner\_domain" : "BenQ",

"owner\_emp\_no" : "1609009"

}, {

"target\_id" : "BQT/T02",

"target\_id\_row\_id" : 2,

"life\_type" : 0,

"reserve\_limit" : 0,

"owner\_login\_id" : "台北總機" ,

"owner\_domain" : "BenQ",

"owner\_emp\_no" : "1609009"

}]

}]

}

}

* + - getTargetByRowID ()
      * 帶入
        + `target\_id`.row\_id
      * 使用 EmpServiceTargetIDRepository.php定義的 getTargetByRowID ()。
    - getTargetByID ()
      * 帶入
        + `service\_id`.row\_id
        + `target\_id`.target\_id
      * 使用 EmpServiceTargetIDRepository.php定義的 getTargetByID ()。

#### **EmpServiceReserveService.php** 定義 Function

* + - newReserve ()
      * 帶入
        + ManagerInfo
        + target\_id\_row\_id
        + login\_id
        + domain
        + emp\_no
        + start\_date
        + end\_date
        + info\_push\_admin\_title
        + info\_push\_admin\_content
        + info\_push\_emp\_title
        + info\_push\_emp\_content
        + info\_data
        + push
      * 使用 EmpServiceReserveRepository.php定義的 newReserve ()。
        + 取得 insertGetId
        + 將新增的資料整理為下列內容，填入變數 $logData

$logData = [

table\_name =>reserve\_record,

table\_row\_id => insertGetId,

action => add,

content => “target\_id\_row\_id:=>target\_id\_row\_id,start\_date:=>start\_date,end\_date:=>end\_date,info\_push\_admin\_title:=>info\_push\_admin\_title,info\_push\_admin\_content:=>info\_push\_admin\_content,info\_push\_emp\_title:=>info\_push\_emp\_title,info\_push\_emp\_content:=>info\_push\_emp\_content,info\_data:=>info\_data,push:=>push”

]

* + - * 新增完成，使用 PushUtil::sendPushMessage() 發送推播
        + 依據 push 的設定，決定發送的對象
        + 如果要發送給 Owner

使用EmpServiceTragetService.php 定義的

getTargetByRowID () 取得 `target\_id`.service\_id\_row\_id。

使用 EmpServiceReserveRepository.php定義的

getEmpServicePushByServiceRowID () 取得要接收推播的人員

* + - * 回傳至 Controller 的資料
        + Array [ $result, $logData ]

$result = [“result\_code” => ResultCode::\_1\_reponseSuccessful, “message” => CommonUtil::getMessageContentByCode(ResultCode::\_1\_reponseSuccessful)]

* + - getReserveRecord ()
      * 帶入
        + `service\_id`.row\_id
        + start\_date
        + end\_date
      * 使用 EmpServiceReserveRepository.php定義的 getReserveRecord ()。
      * 回傳 Reserve Record 資料列表
      * {

"record\_list" : [{

"target\_id" : "BQT/T01",

"target\_id\_row\_id" : 1,

"reserve\_id" : 1,

"reserve\_login\_id" : "Darren.K.Ti",

"reserve\_domain" : "BenQ",

"reserve\_emp\_no" : "1609009",

"info\_push\_admin\_title" : "茶水預約",

"info\_push\_admin\_content" : "BQT/T01 08:30 預約茶杯 5 杯, 水杯 5 杯",

"info\_push\_emp\_title" : "茶水預約",

"info\_push\_emp\_content" : "BQT/T01 08:30 預約茶杯 5 杯, 水杯 5 杯",

"info\_data" : "{‘BQT/T01‘,‘08:30‘,‘5‘,‘5‘}",

"start\_date" : "1542956720",

"end\_date" : "1542956720",

"complete" : "N"

}, {

"target\_id" : "BQT/T02",

"target\_id\_row\_id" : 2,

"reserve\_id" : 2,

"reserve\_login\_id" : "Cleo.W.Chan",

"reserve\_domain" : "BenQ",

"reserve\_emp\_no" : "1609009",

"info\_push\_admin\_title" : "茶水預約",

"info\_push\_admin\_content" : "BQT/T02 10:00 預約茶杯 3 杯, 水杯 7 杯",

"info\_push\_emp\_title" : "茶水預約",

"info\_push\_emp\_content" : "BQT/T02 10:00 預約茶杯 3 杯, 水杯 7 杯",

"info\_data" : "{‘BQT/T02‘,‘10:00‘,‘3‘,‘7‘}",

"start\_date" : "1542956720",

"end\_date" : "1542956720",

"complete" : "Y",

"complete\_login\_id" : "台北總機",

"complete\_at" : "1542956720"

}]

}

* + - getTargetReserveData ()
      * 帶入
        + `target\_id`.row\_id
        + start\_date
        + end\_date
      * 使用 EmpServiceReserveRepository.php定義的 getTargetReserveData ()。
      * 回傳 Reserve Record 資料列表
      * {

"record\_list" : [{

"target\_id" : "BQT/T01",

"target\_id\_row\_id" : 1,

"reserve\_id" : 1,

"reserve\_login\_id" : "Darren.K.Ti",

"reserve\_domain" : "BenQ",

"reserve\_emp\_no" : "1609009",

"info\_push\_admin\_title" : "茶水預約",

"info\_push\_admin\_content" : "BQT/T01 08:30 預約茶杯 5 杯, 水杯 5 杯",

"info\_push\_emp\_title" : "茶水預約",

"info\_push\_emp\_content" : "BQT/T01 08:30 預約茶杯 5 杯, 水杯 5 杯",

"info\_data" : "{‘BQT/T01‘,‘08:30‘,‘5‘,‘5‘}",

"start\_date" : "1542956720",

"end\_date" : "1542956720",

"complete" : "N"

}, {

"target\_id" : "BQT/T02",

"target\_id\_row\_id" : 2,

"reserve\_id" : 2,

"reserve\_login\_id" : "Cleo.W.Chan",

"reserve\_domain" : "BenQ",

"reserve\_emp\_no" : "1609009",

"info\_push\_admin\_title" : "茶水預約",

"info\_push\_admin\_content" : "BQT/T02 10:00 預約茶杯 3 杯, 水杯 7 杯",

"info\_push\_title" : "茶水預約",

"info\_push\_content" : "BQT/T02 10:00 預約茶杯 3 杯, 水杯 7 杯",

"info\_data" : "{‘BQT/T02‘,‘10:00‘,‘3‘,‘7‘}",

"start\_date" : "1542956720",

"end\_date" : "1542956720",

"complete" : "Y",

"complete\_login\_id" : "台北總機",

"complete\_at" : "1542956720"

}]

}

* + - getMyReserveByServiceID ()
      * 帶入
        + `service`.row\_id
        + login\_id
        + domain
        + emp\_no
        + start\_date
        + end\_date
      * 使用 EmpServiceReserveRepository.php定義的 getMyReserveByServiceID ()。
      * 回傳 Reserve Record 資料列表
      * {

"service\_list" : [{

"service\_id" : "staff\_BQT",

"record\_list" : [{

"target\_id" : "BQT/T01",

"target\_id\_row\_id" : 1,

"reserve\_id" : 1,

"info\_push\_admin\_title" : "茶水預約",

"info\_push\_admin\_content" : "BQT/T01 08:30 預約茶杯 5 杯, 水杯 5 杯",

"info\_push\_emp\_title" : "茶水預約",

"info\_push\_emp\_content" : "BQT/T01 08:30 預約茶杯 5 杯, 水杯 5 杯",

"info\_data" : "{‘BQT/T01‘,‘08:30‘,‘5‘,‘5‘}",

"start\_date" : "1542956720",

"end\_date" : "1542956720",

"complete" : "N"

}, {

"target\_id" : "BQT/T02",

"target\_id\_row\_id" : 2,

"reserve\_id" : 2,

"info\_push\_admin\_title" : "茶水預約",

"info\_push\_admin\_content" : "BQT/T02 10:00 預約茶杯 3 杯, 水杯 7 杯",

"info\_push\_emp\_title" : "茶水預約",

"info\_push\_emp\_content" : "BQT/T02 10:00 預約茶杯 3 杯, 水杯 7 杯",

"info\_data" : "{‘BQT/T02‘,‘10:00‘,‘3‘,‘7‘}",

"start\_date" : "1542956720",

"end\_date" : "1542956720",

"complete" : "Y",

"complete\_login\_id" : "台北總機",

"complete\_at" : "1542956720”

}]

}, {

"service\_id" : "staff\_QTY",

"record\_list" : [{

"target\_id" : "QTY/T01",

"target\_id\_row\_id" : 3,

"reserve\_id" : 3,

"info\_push\_admin\_title" : "茶水預約",

"info\_push\_admin\_content" : "QTY/T01 14:30 預約茶杯 8 杯, 水杯 8 杯",

"info\_push\_emp\_title" : "茶水預約",

"info\_push\_emp\_content" : "QTY/T01 14:30 預約茶杯 8 杯, 水杯 8 杯",

"info\_data" : "{‘QTY/T01‘,‘14:30‘,‘8‘,‘8‘}",

"start\_date" : "1542956720",

"end\_date" : "1542956720",

"complete" : "N"

}]

}]

}

* + - getMyReserveByServiceType ()
      * 帶入
        + `service\_id`.type
        + login\_id
        + domain
        + emp\_no
        + start\_date
        + end\_date
      * 使用 EmpServiceReserveRepository.php定義的 getMyReserveByServiceType ()。
      * 回傳 Reserve Record 資料列表
      * {

"service\_list" : [{

"service\_id" : "staff\_BQT",

"record\_list" : [{

"target\_id" : "BQT/T01",

"target\_id\_row\_id" : 1,

"reserve\_id" : 1,

"info\_push\_admin\_title" : "茶水預約",

"info\_push\_admin\_content" : "BQT/T01 08:30 預約茶杯 5 杯, 水杯 5 杯",

"info\_push\_emp\_title" : "茶水預約",

"info\_push\_emp\_content" : "BQT/T01 08:30 預約茶杯 5 杯, 水杯 5 杯",

"info\_data" : "{‘BQT/T01‘,‘08:30‘,‘5‘,‘5‘}",

"start\_date" : "1542956720",

"end\_date" : "1542956720",

"complete" : "N"

}, {

"target\_id" : "BQT/T02",

"target\_id\_row\_id" : 2,

"reserve\_id" : 2,

"info\_push\_admin\_title" : "茶水預約",

"info\_push\_admin\_content" : "BQT/T02 10:00 預約茶杯 3 杯, 水杯 7 杯",

"info\_push\_emp\_title" : "茶水預約",

"info\_push\_emp\_content" : "BQT/T02 10:00 預約茶杯 3 杯, 水杯 7 杯",

"info\_data" : "{‘BQT/T02‘,‘10:00‘,‘3‘,‘7‘}",

"start\_date" : "1542956720",

"end\_date" : "1542956720",

"complete" : "Y",

"complete\_login\_id" : "台北總機",

"complete\_at" : "1542956720”

}]

}, {

"service\_id" : "staff\_QTY",

"record\_list" : [{

"target\_id" : "QTY/T01",

"target\_id\_row\_id" : 3,

"reserve\_id" : 3,

"info\_push\_admin\_title" : "茶水預約",

"info\_push\_admin\_content" : "QTY/T01 14:30 預約茶杯 8 杯, 水杯 8 杯",

"info\_push\_emp\_title" : "茶水預約",

"info\_push\_emp\_content" : "QTY/T01 14:30 預約茶杯 8 杯, 水杯 8 杯",

"info\_data" : "{‘QTY/T01‘,‘14:30‘,‘8‘,‘8‘}",

"start\_date" : "1542956720",

"end\_date" : "1542956720",

"complete" : "N"

}]

}]

}

* + - getReserveByRowID ()
      * 帶入
        + `reserve\_record`.row\_id
      * 使用 EmpServiceReserveRepository.php定義的 getReserveByRowID ()。
    - setReserveComplete ()
      * 帶入
        + `reserve\_record`.row\_id
        + login\_id
        + domain
        + emp\_no
        + complete\_self
      * 使用 EmpServiceReserveRepository.php定義的 setReserveComplete ()。
      * 取得更新結果
      * 將更新的資料整理為下列內容，填入變數 $logData
        + $logData = [

table\_name =>reserve\_record,

table\_row\_id => `reserve\_record`.row\_id,

action => update,

content => “complete:N=>Y,complete\_self:=>complete\_self,complete\_login\_id:=>login\_id,complete\_domain:=>domain,complete\_emp\_no:=>emp\_no]

* + - * 回傳至 Controller 的資料
        + Array [ $result, $logData ]

$result = [“result\_code” => ResultCode::\_1\_reponseSuccessful, “message” => CommonUtil::getMessageContentByCode(ResultCode::\_1\_reponseSuccessful)]

* + - checkReserveByRowID ()
      * 帶入
        + `reserve\_record`.row\_id
        + login\_id
        + domain
        + emp\_no
      * 使用 EmpServiceReserveRepository.php定義的 getReserveByRowID () 取得 Reserve 資料。
        + 檢查 Step 1

如果資料筆數 == 0

回傳 052007 Reserve Not Exist

如果資料筆數 != 0

繼續往下執行

* + - * + 檢查 Step 2

如果 complete == Y

回傳 052008 Reserve Has Completed

如果 complete == N

繼續往下執行

* + - * + 檢查 Step 3

如果 Creator (login\_id / domain / emp\_no) 資料不符合

回傳 052009 Incorrect Reserve Creator

如果 Creator (login\_id / domain / emp\_no) 資料相符

繼續往下執行

* + - * + 檢查 Step 4

如果 `reserve\_record`.start\_date 跟 Now 小於 30 分鐘

回傳 052010 Reserve Delete Date Overdue

如果 `reserve\_record`.start\_date 跟 Now 大於 30 分鐘

回傳

True

* + - deleteReserve ()
      * 帶入
        + `target\_id`.service\_id\_row\_id
        + reserve\_id
        + login\_id
        + domain
        + emp\_no
        + info\_push\_admin\_title
        + info\_push\_admin\_content
        + info\_push\_emp\_title
        + info\_push\_emp\_content
        + push
      * 使用 EmpServiceReserveRepository.php定義的 deleteReserve ()。
        + 將更新的資料整理為下列內容，填入變數 $logData

$logData = [

table\_name =>reserve\_record,

table\_row\_id => reserve\_id,

action => delete,

content => “info\_push\_admin\_title:=>info\_push\_admin\_title,info\_push\_admin\_content:=>info\_push\_admin\_content,info\_push\_emp\_title:=>info\_push\_emp\_title,info\_push\_emp\_content:=>info\_push\_emp\_content,active:=>N,push:=>push”

]

* + - * 新增完成，使用 PushUtil::sendPushMessage() 發送推播
        + 依據 push 的設定，決定發送的對象
        + 如果要發送給 Owner

使用 EmpServiceReserveRepository.php定義的

getEmpServicePushByServiceRowID () 取得要接收推播的人員

* + - * 回傳至 Controller 的資料
        + Array [ $result, $logData ]

$result = [“result\_code” => ResultCode::\_1\_reponseSuccessful, “message” => CommonUtil::getMessageContentByCode(ResultCode::\_1\_reponseSuccessful)]

* + - deitReserve ()
      * 帶入
        + `target\_id`.service\_id\_row\_id
        + reserve\_id
        + target\_id\_row\_id
        + login\_id
        + domain
        + emp\_no
        + start\_date
        + end\_date
        + info\_push\_admin\_title
        + info\_push\_admin\_content
        + info\_push\_emp\_title
        + info\_push\_emp\_content
        + info\_data
        + push
      * 使用 EmpServiceReserveRepository.php定義的 editReserve ()。
        + 將更新的資料整理為下列內容，填入變數 $logData

$logData = [

table\_name =>reserve\_record,

table\_row\_id => reserve\_id,

action => edit,

content => “target\_id\_row\_id:=>target\_id\_row\_id,start\_date:=>start\_date,end\_date:=>end\_date,info\_push\_admin\_title:=>info\_push\_admin\_title,info\_push\_admin\_content:=>info\_push\_admin\_content,info\_push\_emp\_title:=>info\_push\_emp\_title,info\_push\_emp\_content:=>info\_push\_emp\_content,info\_data:=>info\_data,push:=>push”

]

* + - * 更新完成，使用 PushUtil::sendPushMessage() 發送推播
        + 依據 push 的設定，決定發送的對象
        + 如果要發送給 Owner

使用 EmpServiceReserveRepository.php定義的

getEmpServicePushByServiceRowID () 取得要接收推播的人員

* + - * 回傳至 Controller 的資料
        + Array [ $result, $logData ]

$result = [“result\_code” => ResultCode::\_1\_reponseSuccessful, “message” => CommonUtil::getMessageContentByCode(ResultCode::\_1\_reponseSuccessful)]

#### **StatusService.php** 定義 Function

* + - getStatusRowID ()
      * 帶入
        + status\_id
      * 使用 StatusIDRepository.php定義的 getStatusRowID ()。
      * 回傳
        + `status\_id`.row\_id
    - newStatus ()
      * 帶入
        + status\_id : required
        + status\_type : required
        + period\_list : required
      * 使用 StatusIDRepository.php定義的 newStatusID ()。
        + 帶入

status\_id : required

status\_type : required

* + - * + 取得 insertGetId，繼續往下執行
      * 檢查 period\_list 的資料筆數
        + 資料筆數 == 0

回傳 999001 Request Parameter Lost Or Incorrect

* + - * + 資料筆數 != 0

繼續往下處理

* + - * 使用 foreach () 處理 period\_list 內層的資料
        + 如果 life\_type = 1，檢查 start\_date 是否 <= end\_date

否

回傳 000936 Date Range Invalid

是

繼續往下執行

* + - * + 使用 StatusLifeCrontabRepository.php定義的 newStatusLifeCrontab ()。

帶入

`status\_id`.row\_id : required

status : required

note : optional

life\_type : required

crontab : required

* + - * 依據新增的結果，將資料整理為下列內容，填入變數 $logData

$logData = [

[

table\_name =>status\_id,

table\_row\_id => `status\_id`.row\_id,

action => add,

content => “status\_id:=>status\_id,type:=>type

],

[

table\_name =>status\_life\_crontab,

table\_row\_id => `status\_life\_crontab`.row\_id,

action => add,

content => “status\_id\_row\_id:=>status\_id\_row\_id,status:=>status,life\_type:=>life\_type,crontab:=>crontab,note:=>note

]

]

* + - * 回傳至 Controller 的資料
        + Array [ $result, $logData ]

$result = [“result\_code” => ResultCode::\_1\_reponseSuccessful, “message” => CommonUtil::getMessageContentByCode(ResultCode::\_1\_reponseSuccessful)]

* + - getStatusLifeCrontabByRowID ()
      * 帶入
        + `status\_life\_crontab`.row\_id
      * 使用 StatusLifeCrontabRepository.php定義的 getStatusLifeCrontabByRowID ()。
    - updateStatus ()
      * 帶入
        + period\_list : required
      * 使用 foreach () 處理 period\_list 內層的資料
        + 建立一個暫存 [ 比對資料 ] 的陣列
        + 使用StatusService.php 定義的 getStatusLifeCrontabByRowID ()。

如果資料筆數 == 0

回傳 000937 Life Crontab Not Exist

如果資料筆數 != 0

將資料寫入 [ 比對資料 ]，繼續往下執行

* + - * + 如果 life\_type = 1，檢查 start\_date 是否 <= end\_date

否

回傳 000936 Date Range Invalid

是

繼續往下執行

* + - * + 使用 StatusLifeCrontabRepository.php定義的 updateStatusLifeCrontab ()。

帶入

`status\_life\_crontab`.row\_id : required

status : required

note : optional

life\_type : required

crontab : required

依據更新的結果，將新的資料寫入 [ 比對資料 ]

* + - * 依據更新的結果，將 [ 比對資料 ] 整理為下列內容，填入變數 $logData
        + $logData = [

[

table\_name =>status\_life\_crontab,

table\_row\_id => `status\_life\_crontab`.row\_id,

action => update,

content => “status:old\_status=>new\_status,life\_type:old\_life\_type=>new\_life\_type,crontab:old\_crontab=>new\_crontab,note:old\_note=>new\_note

]

]

* + - * 回傳至 Controller 的資料
        + Array [ $result, $logData ]

$result = [“result\_code” => ResultCode::\_1\_reponseSuccessful, “message” => CommonUtil::getMessageContentByCode(ResultCode::\_1\_reponseSuccessful)]

* + - getLifeCrontabByStatusID
      * 帶入
        + `status\_id`.row\_id
      * 使用 StatusLifeCrontabRepository.php定義的 getLifeCrontabByStatusID ()。
      * 回傳 Life Crontab 資料列表
      * {

"status\_list" : [{

"status\_id" : "staff\_BQT",

"status\_type" : "staff",

"period\_list" : [{

"status" : 1,

"note" : "正常服務",

"life\_crontab\_row\_id" : 2,

"life\_type" : 0,

"crontab" : "\* 10-16 \* \* 135",

"owner\_login\_id" : "台北總機" ,

"owner\_domain" : "BenQ”,

"owner\_emp\_no" : "1609009”

}, {

"status" : 1,

"note" : "正常服務",

"life\_crontab\_row\_id" : 2,

"life\_type" : 0,

"crontab" : "\* 10-16 \* \* 135",

"owner\_login\_id" : "台北總機" ,

"owner\_domain" : "BenQ”,

"owner\_emp\_no" : "1609009”

}]

}]

}

* + - getStatusTypeCount ()
      * 帶入
        + `status\_id`.type
      * 使用 StatusIDRepository.php定義的 getStatusTypeCount ()。
    - getLifeCrontabByStatusType
      * 帶入
        + `status\_id`.type
      * 使用 StatusLifeCrontabRepository.php定義的 getLifeCrontabByStatusType ()。
      * 回傳 Life Crontab 資料列表
      * {

"status\_list" : [{

"status\_id" : "staff\_BQT",

"status\_type" : "staff",

"period\_list" : [{

"status" : 1,

"note" : "正常服務",

"life\_crontab\_row\_id" : 2,

"life\_type" : 0,

"crontab" : "\* 10-16 \* \* 135",

"owner\_login\_id" : "台北總機" ,

"owner\_domain" : "BenQ”,

"owner\_emp\_no" : "1609009”

}, {

"status" : 1,

"note" : "正常服務",

"life\_crontab\_row\_id" : 2,

"life\_type" : 0,

"crontab" : "\* 10-16 \* \* 135",

"owner\_login\_id" : "台北總機" ,

"owner\_domain" : "BenQ”,

"owner\_emp\_no" : "1609009”

}]

}]

}

#### **StatusLogService.php** 定義 Function

* + - * newStatusLog ()
        + 帶入

login\_id

domain

emp\_no

$logData

* + - * + 流程

透過 $logData 取得

table\_name

table\_row\_id

action

content

接著將所有資料，透過 foreach 處理，

使用 StatusLogRepository.php定義的 newStatusLog ()。

* app/ Repositories/

#### **EmpServiceServiceIDRepository.php** 定義 Function

* + - getServiceRowID ()
      * 檢查 Service ID 是否已經存在
      * 帶入
        + `service\_id`.service\_id
      * 條件
        + `service\_id`.active = Y
      * 回傳
        + `service\_id`.row\_id
    - newEmpService ()
      * 新增 Emp Service
      * 帶入
        + `service\_id`.service\_id
        + `service\_id`.type
      * 回傳
        + Insert Row ID
    - deleteEmpService ()
      * 刪除 Emp Service
      * 帶入
        + service\_id
      * 更新
        + `service\_id`.action = N
      * 條件
        + `service\_id`.action = Y
        + `service\_id`.service\_id = service\_id
      * 回傳
        + 更新完成的 `service\_id`.row\_id
    - getEmpServiceByType ()
      * 依據 Type 查詢 Service ID
      * 帶入
        + `service\_id`.type
      * 條件
        + 如果帶入的參數 `service\_id`.type == NULL，不指定 type
        + 如果帶入的參數 `service\_id`.type != NULL，指定 type
        + `service\_id`.active = Y
      * JOIN 條件
        + `data\_log`

table\_name = “service\_id”

table\_row\_id = `service\_id`.row\_id

* + - * 回傳
        + 資料列表

#### **EmpServiceDataLogRepository.php** 定義 Function

* + - newDataLog ()
      * 紀錄操作資料
      * 帶入
        + login\_id
        + domain
        + emp\_no
        + table\_name
        + table\_row\_id
        + action
        + content
    - getServiceManagerByTarget ()
      * 依據 Target 的 Row ID 找到其所屬 Service 的管理者資料
      * 帶入
        + `target\_id`.row\_id
      * 條件
        + `data\_log`.table\_row\_id = (

SELECT row\_id FROM `service\_id` WHERE row\_id = (

SELECT service\_id\_row\_id FROM `target\_id` WHERE row\_id =

`target\_id`.row\_id

)

)

* + - * + `data\_log`.table\_name = “service\_id”
        + `data\_log`.action = “add”
        + ORDER BY created\_at DESC LIMIT 1
      * 回傳
        + `data\_log`.login\_id
        + `data\_log`.domain

#### **EmpServiceTargetIDRepository.php** 定義 Function

* + - addEmpServiceTarget ()
      * 新增資料
      * 帶入
        + `service\_id`.row\_id
        + target\_id
        + reserve\_limit
        + life\_type
      * 回傳
        + Insert Row ID
    - updateEmpServiceTarget ()
      * 更新資料
      * 帶入
        + `target\_id`.row\_id
        + active
      * 回傳
        + Updated Row ID
    - deleteTarget ()
      * 設定資料的 Active = N
      * 帶入
        + `service\_id`.row\_id
        + target\_id
      * 條件
        + `target\_id`.active = Y
      * 回傳
        + Updated Row ID
    - getTargetByServiceID ()
      * 依據 Service ID 查詢 Target
      * 帶入
        + `service\_id`.row\_id
      * 條件
        + `target\_id`.active = Y
        + `target\_id`.service\_id\_row\_id = `service\_id`.row\_id
      * JOIN 條件
        + `data\_log`

table\_name = “target\_id”

table\_row\_id = `target\_id`.row\_id

* + - * 回傳
        + 資料列表
    - getTargetByServiceType ()
      * 依據 Service Type 查詢 Target
      * 帶入
        + `service\_id`.type
      * 條件
        + `target\_id`.active = Y
        + `target\_id`.service\_id\_row\_id IN ( SELECT

`service\_id`.row\_id FROM `service\_id` WHERE `service\_id`.type = `service\_id`.type )

* + - * JOIN 條件
        + `data\_log`

table\_name = “target\_id”

table\_row\_id = `target\_id`.row\_id

* + - * 回傳
        + 資料列表
    - getTargetByRowID
      * 依據 Row ID 查詢 Target
      * 帶入
        + `target\_id`.row\_id
      * 條件
        + `target\_id`.active = Y
        + `target\_id`.row\_id = `target\_id`.row\_id
      * 回傳
        + 資料列表
    - getTargetByID
      * 依據 ID 查詢 Target
      * 帶入
        + `service\_id`.row\_id
        + `target\_id`.target\_id
      * 條件
        + `target\_id`.target\_id = `target\_id`.target\_id
        + `target\_id`.service\_id\_row\_id = `service\_id`.row\_id
      * 回傳
        + 資料列表

#### **EmpServiceReserveRepository.php** 定義 Function

* + - newReserve()
      * 新增預約資料
      * 帶入
        + target\_id\_row\_id
        + login\_id
        + domain
        + emp\_no
        + start\_date
        + end\_date
        + info\_push\_admin\_title
        + info\_push\_admin\_content
        + info\_push\_emp\_title
        + info\_push\_emp\_content
        + info\_data
        + push
      * 回傳
        + Insert Row ID
    - getReserveRecord()
      * 取得指定 Service 底下 Target 的預約資料
      * 帶入
        + `service\_id`.row\_id
        + start\_date
        + end\_date
      * 條件
        + `reserve\_record`.target\_id\_row\_id IN ( SELECT `target\_id`.row\_id FROM `target\_id` WHERE `target\_id`.service\_id\_row\_id = `service\_id`.row\_id )
        + `reserve\_record`.start\_date >= start\_date
        + `reserve\_record`.end\_date >= end\_date
        + `reserve\_record`.active == Y
      * 回傳
        + 資料列表
    - getTargetReserveData()
      * 取得指定 Target 的預約資料
      * 帶入
        + `target\_id`.row\_id
        + start\_date
        + end\_date
      * 條件
        + `reserve\_record`.target\_id\_row\_id == `target\_id`.row\_id
        + `reserve\_record`.start\_date >= start\_date
        + `reserve\_record`.end\_date >= end\_date
        + `reserve\_record`.active == Y
      * 回傳
        + 資料列表
    - getMyReserveByServiceID()
      * 取得指定 Service，我的預約資料
      * 帶入
        + `service\_id`.row\_id
        + login\_id
        + domain
        + emp\_no
        + start\_date
        + end\_date
      * 條件
        + `reserve\_record`.target\_id\_row\_id IN ( SELECT `target\_id`.row\_id FROM `target\_id` WHERE `target\_id`.service\_id\_row\_id = `service\_id`.row\_id )
        + `reserve\_record`.login\_id == login\_id
        + `reserve\_record`.domain == domain
        + `reserve\_record`.emp\_no == emp\_no
        + `reserve\_record`.start\_date >= start\_date
        + `reserve\_record`.end\_date >= end\_date
      * 回傳
        + 資料列表
    - getMyReserveByServiceType()
      * 取得指定 Service Type，我的預約資料
      * 帶入
        + `service\_id`.type
        + login\_id
        + domain
        + emp\_no
        + start\_date
        + end\_date
      * 條件
        + `reserve\_record`.target\_id\_row\_id IN ( SELECT `target\_id`.row\_id FROM `target\_id` WHERE `target\_id`.service\_id\_row\_id IN ( SELECT `service\_id`.row\_id FROM `service\_id` WHERE `service\_id`.type = `service\_id`.type ) )
        + `reserve\_record`.login\_id == login\_id
        + `reserve\_record`.domain == domain
        + `reserve\_record`.emp\_no == emp\_no
        + `reserve\_record`.start\_date >= start\_date
        + `reserve\_record`.end\_date >= end\_date
      * JOIN 條件
        + `target\_id`

`target\_id`.row\_id = `reserve\_record`.target\_id\_row\_id

table\_row\_id = `target\_id`.row\_id

* + - * 回傳
        + 資料列表
    - getReserveByRowID()
      * 取得指定 Row ID 的預約資料
      * 帶入
        + `reserve\_record`.row\_id
      * 條件
        + `reserve\_record`.row\_id == `reserve\_record`.row\_id
      * 回傳
        + 單一筆資料
    - setReserveComplete()
      * 設定指定 Row ID 的預約為處理完成
      * 帶入
        + `reserve\_record`.row\_id
        + login\_id
        + domain
        + emp\_no
        + complete\_self
      * 更新
        + `reserve\_record`.complete = Y
        + `reserve\_record`.complete\_self = complete\_self
        + `reserve\_record`.complete\_login\_id = login\_id
        + `reserve\_record`.complete\_domain = domain
        + `reserve\_record`.complete\_emp\_no = emp\_no
      * 回傳
        + 更新結果
    - deleteReserve()
      * 刪除預約資料 (更新 active = N)
      * 帶入
        + `reserve\_record`.row\_id
        + info\_push\_admin\_title
        + info\_push\_admin\_content
        + info\_push\_emp\_title
        + info\_push\_emp\_content
        + push
        + active = N
      * 回傳
        + 更新資料的結果
    - editReserve()
      * 編輯預約資料
      * 帶入
        + `reserve\_record`.row\_id
        + target\_id\_row\_id
        + start\_date
        + end\_date
        + info\_push\_admin\_title
        + info\_push\_admin\_content
        + info\_push\_emp\_title
        + info\_push\_emp\_content
        + info\_data
        + push
      * 回傳
        + 更新資料的結果

#### **EmpServicePushRepository.php** 定義 Function

* + - getEmpServicePushByUser ()
      * 取得員工的 Service 推播設定資料
      * 帶入
        + login\_id
        + domain
        + emp\_no
      * 回傳
        + `service\_push`.service\_id\_row\_id
    - getEmpServicePushByServiceRowID ()
      * 取得 Service 推播設定有哪些成員
      * 帶入
        + `service\_id`.row\_id
      * 回傳
        + `service\_push`.login\_id
        + `service\_push`.domain
        + `service\_push`.emp\_no
    - addEmpServicePush ()
      * 新增員工的 Service 推播設定資料
      * 帶入
        + login\_id
        + domain
        + emp\_no
        + `service\_id`.row\_id
      * 回傳
        + `service\_push`.row\_id
    - deleteEmpServicePushByUser ()
      * 刪除員工的 Service 推播設定資料
      * 帶入
        + login\_id
        + domain
        + emp\_no
        + delete\_service\_row\_id
      * 執行
        + DELETE FROM `service\_push` WHERE

login\_id = login\_id

domain = domain

emp\_no = emp\_no

servie\_id\_row \_id IN (delete\_service\_row\_id)

* + - * 回傳
        + 處理結果
    - deleteEmpServicePush ()
      * 刪除指定 Service 的所有推播設定資料
      * 帶入
        + service\_row\_id
      * 執行
        + DELETE FROM `service\_push` WHERE

servie\_id\_row \_id = service\_row\_id

* + - * 回傳
        + 處理結果

#### **StatusIDRepository.php** 定義 Function

* + - getStatusRowID ()
      * 檢查 Status ID 是否已經存在
      * 帶入
        + `status\_id`.status\_id
      * 條件
        + `status\_id`.active == Y
      * 回傳
        + `status\_id`.row\_id
    - newStatusID ()
      * 新增 Status ID
      * 帶入
        + `status\_id`.status\_id
        + `status\_id`.type
      * 回傳
        + `status\_id`.row\_id
    - getStatusTypeCount ()
      * 取得指定 Status Type 的資料筆數
      * 帶入
        + `status\_id`.type
      * 條件
        + `status\_id`.active == Y
      * 回傳
        + 資料筆數

#### **StatusLifeCrontabRepository.php** 定義 Function

* + - newStatusLifeCrontab ()
      * 新增 Status Life Crontab
      * 帶入
        + `status\_id`.row\_id
        + `status\_life\_crontab`.status
        + `status\_life\_crontab`.life\_type
        + `status\_life\_crontab`.crontab
      * 回傳
        + `status\_life\_crontab`.row\_id
    - getStatusLifeCrontabByRowID ()
      * 取得指定 Row ID 的 Life Crontab 資料
      * 帶入
        + `status\_life\_crontab`.row\_id
      * 條件
        + `status\_life\_crontab`.active == Y
      * 回傳
        + 單一筆資料
    - updateStatusLifeCrontab ()
      * 更新 Status Life Crontab
      * 帶入
        + `status\_life\_crontab`.row\_id
        + `status\_life\_crontab`.status
        + `status\_life\_crontab`.note
        + `status\_life\_crontab`.life\_type
        + `status\_life\_crontab`.crontab
      * 回傳
        + 更新結果
    - getLifeCrontabByStatusID ()
      * 取得指定 Status ID Row ID 的 Life Crontab 資料
      * 帶入
        + `status\_id`.row\_id
      * 條件
        + `status\_life\_crontab`.status\_id\_row\_id == `status\_id`.row\_id
        + `status\_life\_crontab`.active == Y
      * JOIN 條件
        + `status\_log`

table\_name = “status\_life\_crontab”

table\_row\_id = `status\_life\_crontab`.row\_id

* + - * 回傳
        + 資料列表
    - getLifeCrontabByStatusType ()
      * 取得指定 Status Type 的 Life Crontab 資料
      * 帶入
        + `status\_id`.type
      * 條件
        + `status\_life\_crontab`.status\_id\_row\_id IN ( SELECT `status\_id`.row\_id FROM `status\_id` WHERE `status\_id`.type = `status\_id`.type AND `status\_id`.active = Y)
        + `status\_life\_crontab`.active == Y
      * JOIN 條件
        + `status\_log`

table\_name = “status\_life\_crontab”

table\_row\_id = `status\_life\_crontab`.row\_id

* + - * 回傳
        + 資料列表

#### **StatusLogRepository.php** 定義 Function

* + - newStatusLog ()
      * 紀錄操作資料
      * 帶入
        + login\_id
        + domain
        + emp\_no
        + table\_name
        + table\_row\_id
        + action
        + content

#### app/ lib/ **ResultCode.php / Error.json**

* + 新增
    - \_052001\_empServiceExist
      * en-us : this service\_id already exist.
      * zh-tw : service\_id已存在
      * zh-cn : service\_id已存在
    - \_052002\_empServiceNotExist
      * en-us : this service not exist.
      * zh-tw : service\_id不存在
      * zh-cn : service\_id不存在
    - \_052003\_empServiceTypeNotExist
      * en-us : this service type not exist
      * zh-tw : service type 不存在
      * zh-cn : service type 不存在
    - \_052004\_empServiceTargetNotExist
      * en-us : service - target not exist
      * zh-tw : service - target 不存在
      * zh-cn : service - target 不存在
    - \_052005\_empServiceDateRangeInvalid
      * en-us : service - target life cycle date range invalid
      * zh-tw : service - target 生命週期的日期範圍錯誤
      * zh-cn : service - target 生命周期的日期范围错误
    - \_052006\_empServicePushDataInvalid
      * en-us : push receiver setting data invalid
      * zh-tw : 推播對象的設定資料格式錯誤
      * zh-cn : 推播对象的设定资料格式错误
    - \_052007\_empServiceReserveNotExist
      * en-us : reservation not exist
      * zh-tw : 預約不存在
      * zh-cn : 预约不存在
    - \_052008\_empServiceReserveHasCompleted
      * en-us : reservation has completed
      * zh-tw : 預約已經完成
      * zh-cn : 预约已经完成
    - \_052009\_empServiceReserveEmpDataIncorrct
      * en-us : Incorrect emp data of reservation
      * zh-tw : 建立預約服務的員工資料不正確
      * zh-cn : 建立预约服务的员工资料不正确
    - \_052010\_empServiceReserveDateOverdueCannotCancel
      * en-us : Reservation cannot be canceled due to it will start in 30 mins.
      * zh-tw : 預約服務即將在30分鐘內執行, 無法編輯
      * zh-cn : 预约服务即将在30分钟内执行, 无法编辑
    - \_052011\_empServicePushNotExist
      * en-us : Delete Service push fail, the emp didn’t set it.
      * zh-tw : 移除 Service 推播失敗, 此員工沒有新增此 Service 推播
      * zh-cn : 移除 Service 推播失败, 此员工没有新增此 Service 推播
    - \_000933\_statusStatusIDExist
      * en-us : this status\_id already exist.
      * zh-tw : status\_id 已存在
      * zh-cn : status\_id 已存在
    - \_000934\_statusStatusIDNotExist
      * en-us : this status\_id not exist.
      * zh-tw : status\_id 不存在
      * zh-cn : status\_id 不存在
    - \_000935\_statusStatusTypeNotExist
      * en-us : this status type not exist
      * zh-tw : status type 不存在
      * zh-cn : status type 不存在
    - \_000936\_statusStatusDateRangeInvalid
      * en-us : status life cycle date range invalid
      * zh-tw : status 生命週期的日期範圍錯誤
      * zh-cn : status 生命周期的日期范围错误
    - \_000937\_statusStatusLifeCrontabNotExist
      * en-us : status life cycle not exist
      * zh-tw : status 生命週期不存在
      * zh-cn : status 生命周期不存在
    - \_000938\_statusStatusLifeCrontabCanNotNull
      * en-us : status life cycle can not be null
      * zh-tw : status 生命週期不可以為空值
      * zh-cn : status 生命周期不可以为空值

### 4.13 UI 示意圖

* 無

### 4.14 Test Case

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |
|  |  |