第9章　程式

9-1 元件清單及其規格描述

▼表9-1-2 元件清單及其規格描述表(後端)

|  |  |  |
| --- | --- | --- |
| Python檔案(後端)－Flask API | | |
| 編號 | 檔案名稱 | 功能 |
| 2-4-1 | FlaskAPI.py | 1. 將資料庫與APP串接。  2. 驗證使用者資料，並修改資料庫中的資料。  3. APP使用HTTP請求向API索取符合搜尋條件之房屋資料，API搜尋資料庫中的房屋資訊並回傳。  4. APP使用HTTP請求向API傳送刊登或修改的房屋資料，API讀取後將使用者填寫的資料庫儲存進資料庫。  5. APP使用HTTP請求向API傳送訂閱條件資料，API讀取後將使用者訂閱的資料庫儲存進資料庫，在之後若有符合條件之房屋回傳  6. APP使用HTTP請求向API獲取收藏及瀏覽紀錄，API搜尋資料庫後回傳。 |

9-2 其他附屬之各種元件

▼表9-2-1 部分程式碼－subscription.py

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 2-4-1 | 程式名稱 | subscription.py |
| 目的 | 在會員忘記密碼時，寄送驗證碼供會員可以變更資料庫中的密碼 | | |
| 部分程式碼 | | | |
| 1. **Add Subscription API 流程**    * **接收並解析請求資料**:      + 使用 request.json 從請求主體中提取 criteria。      + 從 criteria 中提取 member\_id，如果沒有提供 member\_id，返回錯誤訊息和 HTTP 400 狀態碼。    * **設置訂閱時間**:      + 使用 datetime.now() 設置訂閱時間 (subscription\_time) 和最後檢查時間 (last\_check\_time)。    * **插入訂閱資料至資料庫**:      + 使用 SQL 語句 "INSERT INTO subscriptions (member\_id, criteria, subscription\_time, last\_check\_time)" 將 member\_id、訂閱條件、訂閱時間、最後檢查時間插入 subscriptions 資料表。      + 如果插入成功，返回訂閱成功訊息，並附帶 subscription\_id 和訂閱時間。      + 如果插入失敗，捕捉錯誤並返回錯誤訊息與 HTTP 500 狀態碼。 2. **Delete Subscription API 流程**    * **接收並解析請求資料**:      + 使用 request.json 從請求主體中提取 subscription\_id。      + 如果 subscription\_id 沒有提供，返回錯誤訊息和 HTTP 400 狀態碼。    * **刪除訂閱資料**:      + 使用 SQL 語句 "DELETE FROM subscriptions WHERE subscription\_id = %s" 刪除對應的 subscription\_id 資料。      + 如果刪除成功，返回訂閱刪除成功訊息。      + 如果刪除失敗，捕捉錯誤並返回錯誤訊息與 HTTP 500 狀態碼。 3. **Get Subscriptions API 流程**    * **接收並解析請求資料**:      + 使用 request.json 從請求主體中提取 member\_id。      + 如果 member\_id 沒有提供，返回錯誤訊息和 HTTP 400 狀態碼。    * **查詢訂閱資料**:      + 使用 SQL 語句 "SELECT subscription\_id, criteria FROM subscriptions WHERE member\_id = %s" 查詢對應 member\_id 的訂閱資料。      + 如果沒有找到訂閱資料，返回錯誤訊息和 HTTP 404 狀態碼。    * **處理訂閱資料**:      + 從查詢結果中解析每筆訂閱資料的 criteria，並將 JSON 字符串轉換為字典格式。      + 組織訂閱資料並將其加入 subscriptions 列表。    * **返回訂閱資料**:      + 返回訂閱資料列表，包含 subscription\_id、city、district、pattern、rentalrange、roomcount、size 和 type 等資訊，並回應 HTTP 200 狀態碼。 4. **Update Last Check Time API 流程**    * **接收並解析請求資料**:      + 使用 request.json 從請求主體中提取 subscription\_id 和 member\_id。      + 如果沒有提供 subscription\_id 或 member\_id，返回錯誤訊息和 HTTP 400 狀態碼。    * **更新資料庫中的最後檢查時間**:      + 使用 datetime.now() 設置當前時間作為 last\_check\_time。      + 使用 SQL 語句 "UPDATE subscriptions SET last\_check\_time = %s WHERE subscription\_id = %s AND member\_id = %s" 將 last\_check\_time 更新到對應的 subscription\_id 和 member\_id 訂閱資料。      + 如果更新成功，返回更新成功訊息。      + 如果更新失敗，捕捉錯誤並返回錯誤訊息與 HTTP 500 狀態碼。 | | | |

▼表9-2-2 部分程式碼－xxx.py

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 對應9-1編號 | 程式名稱 | xxx.py |
| 目的 |  | | |
| 部分程式碼 | | | |
| 1. **Add Subscription API 流程**    * **接收並解析請求資料**:      + 使用 request.json 從請求主體中提取 criteria。      + 從 criteria 中提取 member\_id，如果沒有提供 member\_id，返回錯誤訊息和 HTTP 400 狀態碼。    * **設置訂閱時間**:      + 使用 datetime.now() 設置訂閱時間 (subscription\_time) 和最後檢查時間 (last\_check\_time)。    * **插入訂閱資料至資料庫**:      + 使用 SQL 語句 "INSERT INTO subscriptions (member\_id, criteria, subscription\_time, last\_check\_time)" 將 member\_id、訂閱條件、訂閱時間、最後檢查時間插入 subscriptions 資料表。      + 如果插入成功，返回訂閱成功訊息，並附帶 subscription\_id 和訂閱時間。      + 如果插入失敗，捕捉錯誤並返回錯誤訊息與 HTTP 500 狀態碼。 2. **Delete Subscription API 流程**    * **接收並解析請求資料**:      + 使用 request.json 從請求主體中提取 subscription\_id。      + 如果 subscription\_id 沒有提供，返回錯誤訊息和 HTTP 400 狀態碼。    * **刪除訂閱資料**:      + 使用 SQL 語句 "DELETE FROM subscriptions WHERE subscription\_id = %s" 刪除對應的 subscription\_id 資料。      + 如果刪除成功，返回訂閱刪除成功訊息。      + 如果刪除失敗，捕捉錯誤並返回錯誤訊息與 HTTP 500 狀態碼。 3. **Get Subscriptions API 流程**    * **接收並解析請求資料**:      + 使用 request.json 從請求主體中提取 member\_id。      + 如果 member\_id 沒有提供，返回錯誤訊息和 HTTP 400 狀態碼。    * **查詢訂閱資料**:      + 使用 SQL 語句 "SELECT subscription\_id, criteria FROM subscriptions WHERE member\_id = %s" 查詢對應 member\_id 的訂閱資料。      + 如果沒有找到訂閱資料，返回錯誤訊息和 HTTP 404 狀態碼。    * **處理訂閱資料**:      + 從查詢結果中解析每筆訂閱資料的 criteria，並將 JSON 字符串轉換為字典格式。      + 組織訂閱資料並將其加入 subscriptions 列表。    * **返回訂閱資料**:      + 返回訂閱資料列表，包含 subscription\_id、city、district、pattern、rentalrange、roomcount、size 和 type 等資訊，並回應 HTTP 200 狀態碼。 | | | |

▼表9-2-3 部分程式碼－subscriptiom.py

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 對應9-1編號 | 程式名稱 | .py |
| 目的 | 將會員執行新增或刪除的訂閱條件存入資料庫 | | |
| 部分程式碼 | | | |
| 1. **Add Subscription API 流程**    * **接收並解析請求資料**:      + 使用 request.json 從請求主體中提取 criteria。      + 從 criteria 中提取 member\_id，如果沒有提供 member\_id，返回錯誤訊息和 HTTP 400 狀態碼。    * **設置訂閱時間**:      + 使用 datetime.now() 設置訂閱時間 (subscription\_time) 和最後檢查時間 (last\_check\_time)。    * **插入訂閱資料至資料庫**:      + 使用 SQL 語句 "INSERT INTO subscriptions (member\_id, criteria, subscription\_time, last\_check\_time)" 將 member\_id、訂閱條件、訂閱時間、最後檢查時間插入 subscriptions 資料表。      + 如果插入成功，返回訂閱成功訊息，並附帶 subscription\_id 和訂閱時間。      + 如果插入失敗，捕捉錯誤並返回錯誤訊息與 HTTP 500 狀態碼。 2. **Delete Subscription API 流程**    * **接收並解析請求資料**:      + 使用 request.json 從請求主體中提取 subscription\_id。      + 如果 subscription\_id 沒有提供，返回錯誤訊息和 HTTP 400 狀態碼。    * **刪除訂閱資料**:      + 使用 SQL 語句 "DELETE FROM subscriptions WHERE subscription\_id = %s" 刪除對應的 subscription\_id 資料。      + 如果刪除成功，返回訂閱刪除成功訊息。      + 如果刪除失敗，捕捉錯誤並返回錯誤訊息與 HTTP 500 狀態碼。 | | | |

▼表9-2-4 部分程式碼－xxx.py

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 對應9-1編號 | 程式名稱 | xxx.py |
| 目的 |  | | |
| 部分程式碼 | | | |
|  | | | |

▼表9-2-3 部分程式碼－xxx.py

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 對應9-1編號 | 程式名稱 | xxx.py |
| 目的 |  | | |
| 部分程式碼 | | | |
|  | | | |