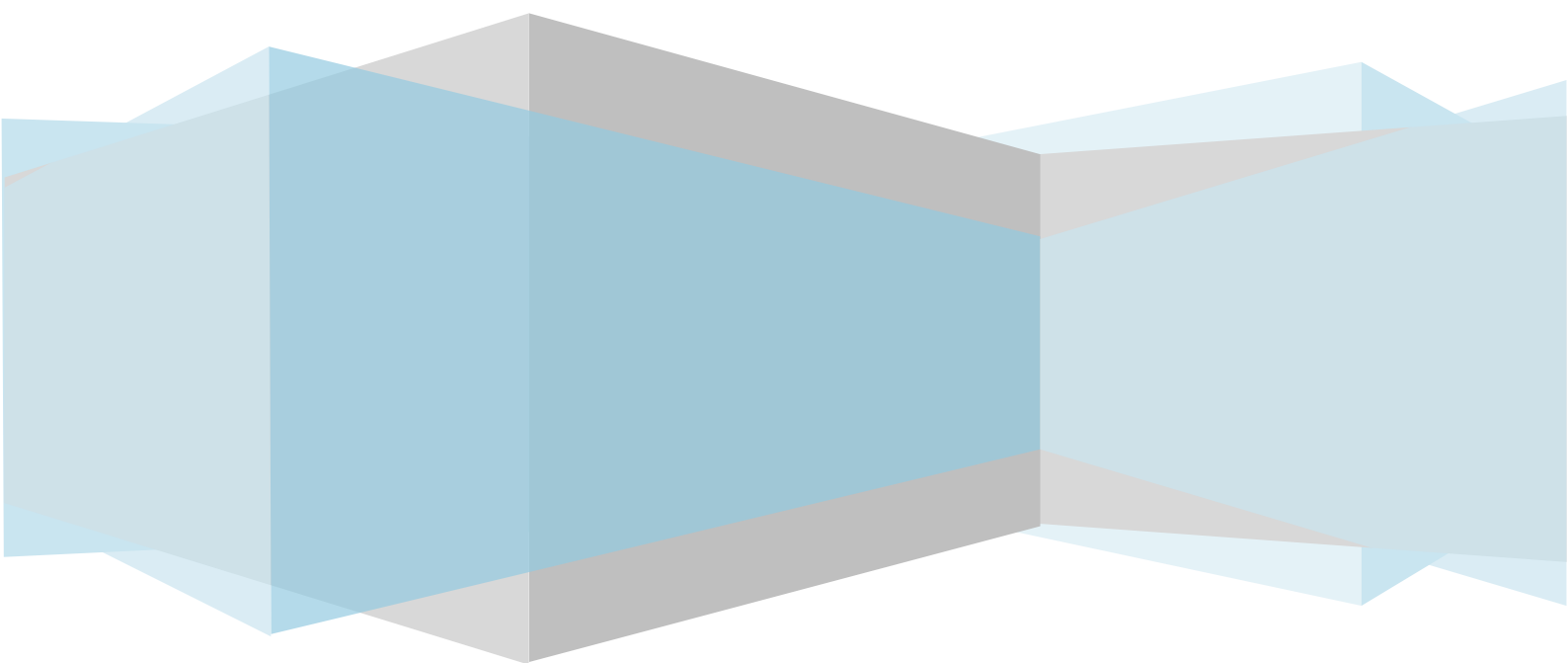


Mini 每日一問



目錄

壹、 前言	2
一、 設計理念	2
二、 適用範圍	2
貳、 流程圖	3
一、 流程圖說明	3
參、 程式介紹	4
一、 開始畫面	4
二、 首頁畫面	5
三、 Google 登入	13
四、 Github	15
肆、 資料庫結構	16
一、 留言板	16
二、 每日一問	17
伍、 台灣農產 B&Q(團隊)	18
一、 簡介	18
二、 功能介面	20
陸、 補充作品(個人)	27
一、 個人網站	27
二、 SPA 網站	28
三、 會員管理系統網站	29
四、 串接 Open Data 網站	30
柒、 實作心得	31

壹、前言

「Mini 每日一問」（以下簡稱 APP）是利用 **主從式架構**，讓 Client 端向 Server 端抓取資料，配合執行續讓 Client 端持續更新。而留言板更是需要即時更新資料，除了定時更新外，每次的新增留言、刪除留言也會影響資料庫的完整性，以下為 Class 使用到的功能：

✚ 讓 Client 端 連線 Server 端 ，抓取、新增、刪除資料。	✚ 申請 Google OAuth2.0 用戶端 ID 憑證、在 Google Play Console 上架 APP。
✚ 使用 SQLite 作為 暫存資料庫 ，無網路時，也可讀取暫存資料。	✚ Gradle 加入 play-services-auth:19.2.0 套件，來使用 Google Sign In。
✚ 使用 OKHttpClient 套件，連線 MySQL 資料庫。	✚ 使用 生命週期 ，onCreate、onStart、onResume、onPause、onStop。
✚ 使用 PHP 語法 ，SELECT、insert into、JOIN 篩選資料。	✚ 使用 for 迴圈，渲染多筆資料到畫面上，並判斷使用者資訊。
✚ 使用 Handler 執行續，持續更新資料、延後程式執行。	✚ 使用 AlertDialog 作為提示說明、顯示訊息。
✚ 使用 Menu，切換功能及頁面。	✚ 使用 Intent，進入、切換頁面。

一、設計理念

APP 以留言版功能為設計介面，並以每日一問作為聊天主題，每天都會有一個問題作為探討。

問題的內容採多元主題，有日常生活的知識、猜謎、腦筋急轉彎、台灣各種知識主題等，讓使用者每天都有使用 APP 的契機，使用者可以一起討論、並且延伸探討、抒發心情等，讓留言板達到最佳的使用效果。

二、適用範圍

客群以青少年、成年人為主要族群，這個族群擅長使用網路作為聊天的工具，而留言版的功能，正是網路發達時代常用的功能之一，APP 給予使用者每日一個主題去探討，來達到使用者互動的效果。

本應用程式介面適用於教學討論、知識探討、各類主題內容討論等。



圖 1-1 APP QR Code

貳、流程圖

一、 流程圖說明

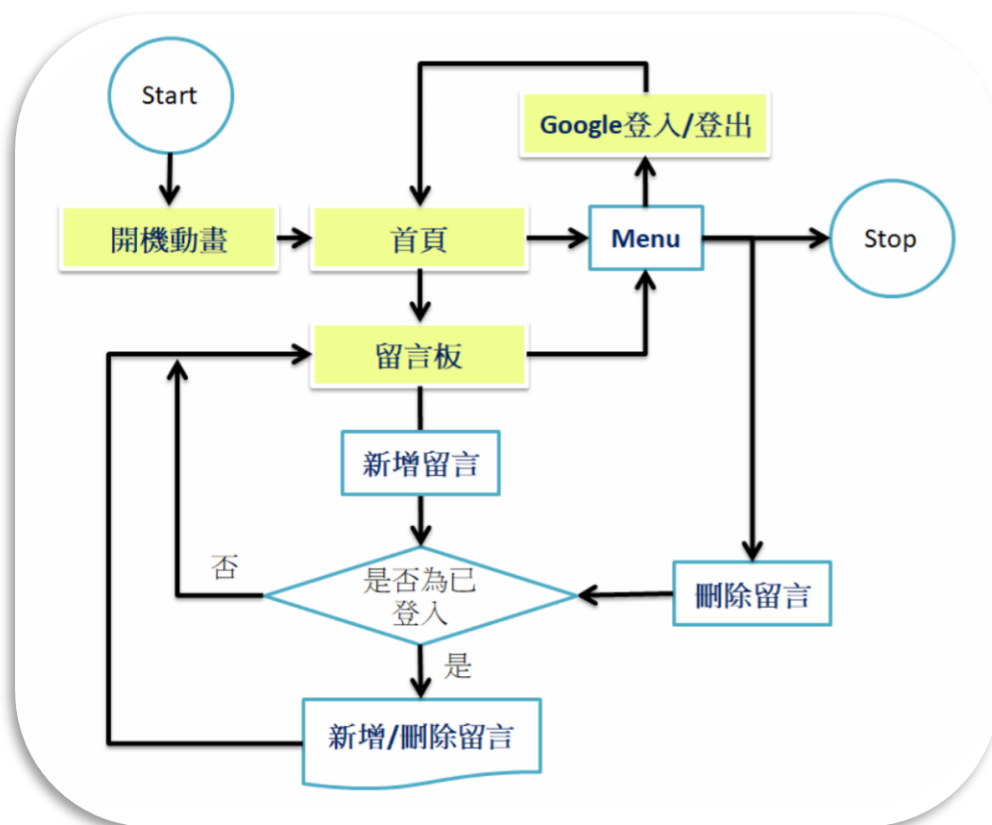


圖 2-1 APP 流程圖

進入 APP 會先看到開機動畫，動畫使用 **alpha 淡出** 效果，讓 APP 的 logo 出現，之後使用執行續延遲 3 秒，執行 Intent 進入首頁。

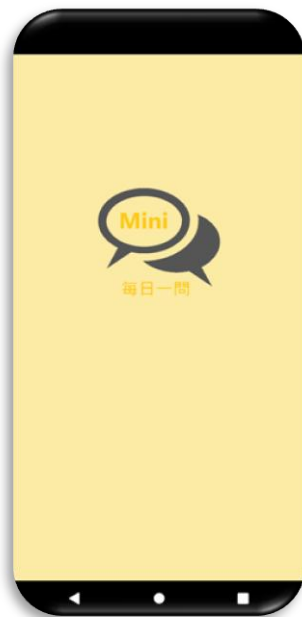
「首頁」主要功能為「留言板」，使用 for 迴圈再 Layout 渲染畫面，用 MySQL、SQLite 作為資料庫，可以執行讀取、新增、刪除留言等功能，因此，在進行新增/刪除前，會先**判斷是否為已登入**，才能使用新增/刪除功能。

「Google 登入」使用了 Google 的登入系統，藉由將會員資料、留言訊息寫入資料庫，來判斷該筆訊息的使用者，才可執行個人的新增、刪除訊息。

參、程式介紹

一、開始畫面

當開啟 APP 時，會先出現的畫面，以 logo 作為構圖中心，為了不讓畫面變靜態，增加了 alpha 淡出效果，並且使用執行續，延遲 3 秒後執行進入首頁畫面。



問題

✚ 因為 Style 樣式，導致圖片無法滿版覆蓋。

程式碼

設定 img01 (logo 圖) 執行動畫，alpha 淡出效果延遲 0.5 秒、執行 2 秒，並設定執行續 3 秒後執行 Intent，設定傳遞標題的參數為「class_title」，從 F0102.Class Intent 到 F0100.Class。

在 onCreate 加入 `getSupportActionBar().hide()` 語法，讓圖片可以有滿版的效果。

```
private void setupViewComponent() {
    lay01 = (RelativeLayout)findViewById(R.id.f0102_lay01);
    img01 = (ImageView)findViewById(R.id.f0102_img001);

    lay01.setOnClickListener(this);
    //--- 開機動畫---
    img01.setAnimation(AnimationUtils.LoadAnimation(context, R.anim.q0101_anim_alpha_in_01));

    //=====設定執行緒=====
    handler.postDelayed(updateTimer, delayMillis: 3000); // 設定DeLay的時間
    //-----

}

//=====設定執行續=====
private Runnable updateTimer = new Runnable() {
    @Override
    public void run() {
        intent.putExtra( name: "class_title", "Mini每日一問");
        intent.setClass( packageContext: F0102.this, F0100.class);
        startActivity(intent);
    }
};
```

```
// 設定隱藏標題
getSupportActionBar().hide();
// 設定隱藏狀態
getWindow().getDecorView().setSystemUiVisibility(View.SYSTEM_UI_FLAG_FULLSCREEN);
```

二、 首頁畫面

上方為每日一問的 TextView，下方則是留言板，右上角有 Menu 選單，最下面為輸入欄位。

將主要功能放在首頁，讓使用者可以一眼看出有哪些功能，並且快速上手。



問題

- 當用 For 迴圈更新留言板時，物件會一直重複疊加。
- 資料庫主機時區為「GMT」，欄位屬性為 timestamp，留言顯示會有時差。
- 在刪除模式時（只渲染使用者留言），點選留言 Button 會造成全部（含其他使用者）刪除鍵被渲染出來。

程式碼

此段為檢查使用者登入狀態，將 check_SignIn 放在 onStart，每次啟動都會檢查，當使用者為已登入狀態會傳遞 account，未登入則會傳遞空值，供渲染畫面使用。

```
//-----檢查登入狀態-----
private void check_SignIn() {
    // Check if the user is already signed in and all required scopes are granted
    GoogleSignInAccount account = GoogleSignIn.getLastSignedInAccount(context);
    if (account != null && GoogleSignIn.hasPermissions(account, new Scope(Scopes.DRIVE_APPFOLDER))) {
        ACC = account;
    } else {
        ACC = null;
    }
}
```

```
//-----SQLite-----
private void initDB() {
    if (dbHper == null) {
        dbHper = new DbHelper(context, DB_FILE, factory: null, DB_version);
    }
    dbmysql_F0100();
    dbmysql_F0101();
}
```

將 initBD 放在 onCreate，用來判斷 SQLite (dbHper) 是否存在(是否為第一次安裝)，如果為空則新增資料庫 (new DbHelper)，並執行 dbmysql，將 MySQL 的資料寫入 SQLite，讓 SQLite 的資料為最新。

setupViewComponent 放在 onCreate 為設定 Layout 元件，將需要監聽 Button、ImageButton 宣告在這裡。

抓取螢幕尺寸，將需要彈性調整的元件，設定寬度 (width) 等於螢幕寬度/30*22，即將螢幕寬度 30 等分並取 22 等分作為其實際渲染畫面的寬度。

showTopic 為渲染每日一句的 TextView，之後設定執行續 updateTimer，讓 APP 定時更新資料庫，最後 last_scroll 為 ScrollView 顯示最下面的畫面。

```
//-----設定Layout元件-----
private void setupViewComponent() {
    b001 = (Button)findViewById(R.id.f0100_b001);
    e001 = (EditText)findViewById(R.id.f0100_e001);
    t004 = (TextView)findViewById(R.id.f0100_t004);
    t005 = (TextView)findViewById(R.id.f0100_t005);
    img004 = (ImageButton)findViewById(R.id.f0100_img004);

    b001.setOnClickListener(this);
    img004.setOnClickListener(this);
    //-----抓取螢幕尺寸-----
    DisplayMetrics displayMetrics = new DisplayMetrics();
    this.getWindowManager().getDefaultDisplay().getMetrics(displayMetrics);

    //-----動態設定長度-----
    e001.getLayoutParams().width = displayMetrics.widthPixels / 30 * 22;
    b001.getLayoutParams().width = displayMetrics.widthPixels / 30 * 6;

    showTopic();

    //=====設定執行緒=====
    if (runAuto_flag == false){
        handler.postDelayed(updateTimer, delayMillis: 100); // 設定DeLay的時間
        runAuto_flag = true;
    }

    //----顯示最後面----
    last_scroll();
}
```

```
//=====設定執行緒=====
private Runnable updateTimer = new Runnable() {
    @Override
    public void run() {
        handler.postDelayed(this, delayMillis: autotime * 1000); // 真正延遲的時間
        //-----更新時間
        java.sql.Date curDate = new java.sql.Date(System.currentTimeMillis()); // 獲取當前時間
        str = formatter.format(curDate);
        t005.setText(str+" (10秒更新1次)");
        //-----執行進入MySQL
        dbmysql_F0100();
        recSet_F0100 = dbHper.getRecSet_F0100();
        showRec();
        //-----
        ++update_time;
    }
};
```

updateTimer 為設定定時更新，autotime 變數宣告為整數，初始值為 10，即為延後 10 秒執行。

str 變數為抓取系統時間並轉成字串，讓 TextView 顯示，之後執行

dbmysql_F0100 更新留言板資料庫，抓取資料並渲染畫面，++update_time 為持續執行。


```

private void dbmysql_F0100() {
    String sqlctl = "SELECT * FROM F0100"; //---SELECT 資料庫的 F0100 Table 的字串
    ArrayList<String> nameValuePairs = new ArrayList<>();
    nameValuePairs.add(sqlctl);
    try {
        String result = DBConnector.executeQuery_F0100(nameValuePairs); //---連線 MySQL
        JSONArray jsonArray = new JSONArray(result);
        // -----
        if (jsonArray.length() > 0) { //---MySQL 連結成功有資料
            int rowsAffected = dbHper.clearRec_F0100(); //---匯入前，刪除所有SQLite資料

            for (int i = 0; i < jsonArray.length(); i++) { //---處理JSON 傳回來的每筆資料
                JSONObject jsonData = jsonArray.getJSONObject(i);
                ContentValues newRow = new ContentValues();
                // --(1) 自動取的欄位 --取出 jsonObject 每個欄位("key", "value")
                Iterator itt = jsonData.keys();
                while (itt.hasNext()) {
                    String key = itt.next().toString();
                    String value = jsonData.getString(key); // 取出欄位的值
                    if (value == null) {
                        continue;
                    } else if (!"".equals(value.trim())) {
                        continue;
                    } else {
                        jsonData.put(key, value.trim());
                    }
                    // -----
                    newRow.put(key, value.toString()); // 動態找出有幾個欄位
                    // -----
                }
            }
            long rowID = dbHper.insertRec_m_F0100(newRow); //---MySQL 資料寫入SQLite資料
        }
    }
}

```

1

此段為取得 MySQL 資料庫資料，將 sqlctl 變數帶入 SQL 語法，並執行 DBConnector 外部 Class。

如果連線並取得資料，則先清空 SQLite，避免資料重複堆疊，用 for 迴圈取出每一筆資料，最後將資料寫入 SQLite 暫存。

此段為 DBConnector 的外部 Class，connect_ip 為連線資料庫的 PHP 語法網址，使用 OKHttpClient 連線，將 postUrl、body 帶入 request。

連線成功並取得資料，將回傳 result 字串資料。

```

public static String executeQuery_F0100(ArrayList<String> query_string) {
    // OkHttpClient client = new OkHttpClient();
    postUrl=connect_ip ;
    //-----建立一個封包-----
    String query_0 = query_string.get(0);
    FormBody body = new FormBody.Builder()
        .add( name: "selefunc_string", value: "query")
        .add( name: "query_string", query_0)
        .build();
    //-----執行-----
    Request request = new Request.Builder()
        .url(postUrl)
        .post(body)
        .build();
    try (Response response = client.newCall(request).execute()) {
        return response.body().string();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return result;
}

```

4

此段將取得的 newRow 資料，寫入 SQLite 的 F0100 Table，最後關閉與資料庫的連線。

```

/*****F0100寫入SQLite*****/
public long insertRec_m_F0100(ContentValues rec) {
    SQLiteDatabase db = getWritableDatabase();
    long rowID = db.insert(DB_TABLE_F0100, nullColumnHack: null, rec);
    db.close();
    return rowID;
}

```


此段為 PHP 語法，在一開始設定 hostname、username、password、dbname 的連線資料。

連線成功執行 switch，當 sefunc 變數取得的值為 query，則執行讀取資料，sql 變數為 PHP SELECT 語法，用 while 迴圈讀取資料，並回傳列資料。

最後要關閉 mysqli_con 的連線，避免佔用資源。

```
// Create mysqli_connection
$mysqli_con = new mysqli($hostname, $username, $password, $dbname);
// Check mysqli_connection
if ($mysqli_con->mysqli_connect_error) {
    die("Connection failed: " . $mysqli_con->mysqli_connect_error);
}

//-----
$mysqli_con->query("SET NAMES utf8");
$mysqli_con->query("SET CHARACTER SET 'UTF8'");
$mysqli_con->query('SET CHARACTER_SET_CLIENT=UTF8;');
$mysqli_con->query('SET CHARACTER_SET_RESULTS=UTF8;');
//-----
$sefunc=$_REQUEST['sefunc_string'];
switch($sefunc){
    //-----選擇瀏覽-----
    case "query":
        $sql = $_REQUEST['query_string'];
        $sql = stripslashes($sql);
        $result = $mysqli_con->query($sql);

        if ($result->num_rows > 0) {
            // output data of each row
            while($row = $result->fetch_assoc()) {
                $output[] = $row;
            }
            print(json_encode($output));
        } else {
            // $output[] = $row;
            echo "0 results";
        }
        //print(json_encode($output));
        $mysqli_con->close();
        break;
}
```

4

```
*****F0100抓資料*****
public ArrayList<String> getRecSet_F0100() {
    SQLiteDatabase db = getReadableDatabase();
    String sql = "SELECT * FROM " + DB_TABLE_F0100 + " ORDER BY id ASC";
    Cursor recSet = db.rawQuery(sql, selectionArgs: null);
    ArrayList<String> recAry = new ArrayList<>();
    //-----
    //Log.d(TAG, "recSet=" + recSet);
    int columnCount = recSet.getColumnCount();
    recSet.moveToFirst();
    String fldSet = "";
    if (recSet.getCount() != 0){//-----判斷資料如果 不是0比 才執行抓資料
        for (int i = 0; i < columnCount; i++)
            fldSet += recSet.getString(i) + "#"; // 欄位跟欄位 用 # 做區隔
        recAry.add(fldSet);
    }
    while (recSet.moveToNext()) {
        fldSet = "";
        for (int i = 0; i < columnCount; i++) {
            fldSet += recSet.getString(i) + "#"; // 欄位跟欄位 用 # 做區隔
        }
        recAry.add(fldSet);
    }
    //-----
    recSet.close();
    db.close();
    //Log.d(TAG, "recAry=" + recAry);
    return recAry;
}
```

2

此段為取得 SQLite 資料，sql 為 SELECT F0100 Table 的全部資料，並用 id 做遞增排序。

判斷 recSet 的資料數不等於 0，才執行 for 迴圈，並用 “#” 做欄位之間的區隔，最後關閉 db、recSet 連線並回傳 recAry。

```

private void showRec() {
    //---有資料才執行---
    if (recSet_F0100.size() != 0){
        //---宣告---
        TextView objt001 = (TextView) findViewById(R.id.f0100_t001);
        TextView objt002 = (TextView) findViewById(R.id.f0100_t002);
        TextView objt003 = (TextView) findViewById(R.id.f0100_t003);
        TableRow objt004 = (TableRow) findViewById(R.id.f0100_tab001);
        TableRow objt005 = (TableRow) findViewById(R.id.f0100_tab002);
        CircleImageView objt006 = (CircleImageView) findViewById(R.id.f0100_img002);
        LinearLayout objt007 = (LinearLayout) findViewById(R.id.f0100_Lay002);
        ImageButton objt008 = (ImageButton) findViewById(R.id.f0100_img003);

        //---清空Layout---
        LinearLayout lay01 = (LinearLayout) findViewById(R.id.f0100_Lay001);
        lay01.removeAllViews();

        //=====巨集=====
        try {
            for (int i = 1; i <= recSet_F0100.size(); i++){
                // ---產生新的TextView---
                TextView tv01 = new TextView( context: this);
                TextView tv02 = new TextView( context: this);
                TextView tv03 = new TextView( context: this);
                TableRow tab01 = new TableRow( context: this);
                TableRow tab02 = new TableRow( context: this);
                CircleImageView img01 = new CircleImageView( context: this);
            }
        }
    }
}

```

3

showRec 為渲染留言板畫面，因為程式碼冗長，以下只擷取重要的部分做說明。

開始執行渲染前，先判斷 SQLite 資料庫是否有資料，即 size() != 0 才執行，第一步宣告取用的元件，渲染前執行清空 Layout，避免資料一直疊加，最後執行 for 迴圈。

此段為擷取 SQLite 的資料，將取得的資料儲存在 f0100 陣列，並用 # 字符合號區隔。

時間是資料庫主機自動寫入，而主機時區為 GMT，因此需將時間資料轉換為時間格式，轉換時區後再轉為字串，以供使用。

```

//---擷取SQLite資料---
String[] f0100 = recSet_F0100.get(i-1).split( regex: "#");
String id00 = f0100[0];//---ID
String id01 = f0100[1];//---Email
String id02 = f0100[2];//---FirstName
String id03 = f0100[3];//---LastName
String id04 = f0100[4];//---UserImage
//-----
String id05 = f0100[5];//---Message
//-----時區轉換-----
String timeStr = f0100[6]; // 主機時間
SimpleDateFormat timeZone = new SimpleDateFormat( pattern: "yyyy-MM-dd HH:mm:ss");
timeZone.setTimeZone(TimeZone.getTimeZone("GMT"));
Date date = timeZone.parse(timeStr); // 將字符串時間按時間解析成Date對象
SimpleDateFormat timeZone_TW = new SimpleDateFormat( pattern: "yyyy-MM-dd HH:mm");
timeZone_TW.setTimeZone(TimeZone.getTimeZone("Asia/Taipei"));
String date_TW = timeZone_TW.format(date);
//-----

```

```

switch (view01){
    case 0://-----取消編輯、預設值
        img02.setVisibility(View.GONE);
        img004.setVisibility(View.VISIBLE);
        b001.setVisibility(View.VISIBLE);
        e001.setVisibility(View.VISIBLE);
        break;
    case 1://-----使用編輯
        img02.setVisibility(View.VISIBLE);
        img004.setVisibility(View.GONE);
        b001.setVisibility(View.GONE);
        e001.setVisibility(View.GONE);
        break;
}

```

此段在 showRec 後半段為判斷是否使用刪除功能。當點選刪除留言鍵時，會將 int view01 設為 1 並執行 showRec，顯示刪除的圖示(img02)，並隱藏輸入欄位。

此段為新增留言，當 b001 被觸動時，先判斷是否為已登入狀態，ACC 為儲存會員資料的變數，當 ACC 等於 null，會顯示請登入會員的對話框。

```
@Override
public void onClick(View v) {
    switch (v.getId()) {
        case R.id.f0100_b001:
            if (ACC != null){/--是為已登入
                e_message = e001.getText().toString().trim();/--取得留言
                if (e_message.equals("")){/--檢查輸入欄位是否空白
                    Toast.makeText(getApplicationContext(), text: "留言不可空白 !", Toast.LENGTH_SHORT).show();
                    return;
                }
                /-----取出會員資料-----
                e_email = ACC.getEmail();/--Email
                e_firstname = ACC.getGivenName();/--Firstname
                e_lastname = ACC.getFamilyName();/--Last name
                e_userimage = ACC.getPhotoUrl().toString().trim();/--頭像
                /-----執行SQL-----
                long rowID = dbHper.insertRec_F0100(e_email, e_firstname, e_lastname, e_userimage, e_message);
                /-----執行MySQL-----
                mysql_insert_F0100();

                if (rowID != -1){/--寫入是否成功
                    msg = "新增成功 !";
                    e001.setText("");
                } else {
                    msg = "新增失敗 !";
                }
                Toast.makeText(getApplicationContext(), msg, Toast.LENGTH_SHORT).show();/--顯示msg
                /-----
                dbmysql_F0100();/--抓取MySQL，寫入SQLite
                recSet_F0100 = dbHper.getRecSet_F0100();/--讀取SQLite
                showRec();/--渲染畫面
                last_scroll();/--ScrollView顯示最下面
            }
```

如果 ACC 不等於 null，則會判斷輸入欄位的訊息是否為空，避免空白訊息的問題，如欄位有訊息，則取出會員資料，執行 mysql_insert_F0100 與 insertRec_F0100，當 SQLite 寫入回傳不等於-1，顯示 Toast「新增成功」，否則為「新增失敗」。

最後執行 dbmysql_F0100 取得資料並寫入 SQLite，讓 SQLite 的資料為最新，getRecSet 重新讀取 SQLite 的資料後，showRec 渲染畫面。

```
private void mysql_insert_F0100() {
    //      sqlCtl = "SELECT * FROM member ORDER BY id ASC";
    ArrayList<String> nameValuePairs = new ArrayList<>();
    //      nameValuePairs.add(sqlCtl);
    nameValuePairs.add(e_email);
    nameValuePairs.add(e_firstname);
    nameValuePairs.add(e_lastname);
    nameValuePairs.add(e_userimage);
    nameValuePairs.add(e_message);

    try {
        Thread.sleep( millis: 500); // 延遲Thread 睡眠0.5秒
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
    //-----
    String result = DBConnector.executeInsert_F0100(nameValuePairs); //真正執行新增
    //-----
}
```

5

此段為將使用者資料寫入資料庫的過程，把 E-mail 等資訊帶入 ArrayList，並帶到外部 Class DBConnector，執行 SQL 語法連線、寫入資料庫。

此段為 DBConnector 的外部 Class，**connect_ip** 為連線資料庫的 PHP 語法網址，使用 **OkHttpClient** 連線，將 postUrl、body 帶入 request。

連線成功將資料寫入資料庫，並回傳 result 字串資料。

下方為 PHP 語法，將使用者資料帶入宣告變數，用 **insert into** 寫入 key、values 到資料庫，最後關閉 mysql_con 連線。

```
public static String executeInsert_F0100(ArrayList<String> query_string) {
    //      OkHttpClient client = new OkHttpClient();
    postUrl=connect_ip ;
    //-----
    String query_0 = query_string.get(0);
    String query_1 = query_string.get(1);
    String query_2 = query_string.get(2);
    String query_3 = query_string.get(3);
    String query_4 = query_string.get(4);

    FormBody body = new FormBody.Builder()
        .add( name: "selefunc_string", value: "insert")
        .add( name: "Email", query_0)
        .add( name: "FirstName", query_1)
        .add( name: "LastName", query_2)
        .add( name: "UserImage", query_3)
        .add( name: "Message", query_4)
        .build();
    //-----
    Request request = new Request.Builder()
        .url(postUrl)
        .post(body)
        .build();
    try (Response response = client.newCall(request).execute()) {
        return response.body().string();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return result;
}
```

```
//-----選擇新增-----
case "insert":
$cr_01=$REQUEST['Email'];
$cr_02=$REQUEST['FirstName'];
$cr_03=$REQUEST['LastName'];
$cr_04=$REQUEST['UserImage'];
$cr_05=$REQUEST['Message'];

$sql = "insert into F0100(Email, FirstName, LastName, UserImage, Message) values('$cr_01','$cr_02','$cr_03','$cr_04','$cr_05')";
$flag['code']=0;
if($row=mysqli_con->query($sql))
{
    $flag['code']=1;
    echo"hi ok";
}
print(json_encode($flag));
mysqli_con->close();
break;
```

此段為刪除留言，當刪除鍵被觸動時，會先詢問是否刪除的對話盒，避免使用者誤觸而刪錯留言。

iii 為被點擊 Button 的 id，f0100 取得該筆留言資料，如果為「確定刪除」，將把資料傳遞給 del_choose 並執行刪除。

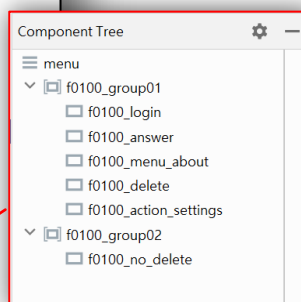
```
//-----刪除對話盒
private View.OnClickListener delOn = new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        int iii = (v.getId());
        f0100 = recSet_F0100.get(iii-1).split( regex: "#");

        MyAlertDialog myAltDlg = new MyAlertDialog( context: F0100.this);
        myAltDlg.getWindow().setBackgroundDrawableResource(R.color.Yellow);
        myAltDlg.setTitle("刪除留言");
        myAltDlg.setMessage("留言刪除無法復原\n確定要刪除嗎?");
        myAltDlg.setCancelable(false);
        myAltDlg.setIcon(android.R.drawable.ic_delete);
        myAltDlg.setButton(BUTTON_POSITIVE, text: "確定刪除", del_choose);
        myAltDlg.setButton(BUTTON_NEGATIVE, text: "取消刪除", del_choose);
        myAltDlg.show();
    }
};
```

當使用者選擇「確定刪除」，tid 變數帶入該筆資料的 id，並執行 MySQL、SQLite 刪除資料。

刪除完畢後，下載 MySQL 資料更新 SQLite 資料庫，讀取 SQLite、渲染畫面，將 view01 設為 0，回到預設畫面、開啟執行續，最後把 Menu 群組 group01 設為開啟。

```
//-----是否刪除
private DialogInterface.OnClickListener del_choose = new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        switch (which) {
            case BUTTON_POSITIVE: //--- 確定刪除
                tid = f0100[0]; //-----取得該筆資料id
                // -----執行刪除-----
                int rowsAffected = (int) dbHper.deleteRec_F0100(tid); //---執行SQLite刪除
                mysql_del_F0100(); //---執行MySQL刪除
                dbmysql_F0100(); //---更新SQLite
                //-----是否刪除成功-----
                if (rowsAffected == -1) {
                    msg = "留言不存在，無法刪除 !";
                } else if (rowsAffected == 0) {
                    msg = "留言不存在，無法刪除 !";
                } else {
                    msg = "留言已刪除 !";
                }
                //-----渲染畫面-----
                recSet_F0100 = dbHper.getRecSet_F0100(); //---讀取SQLite
                view01 = 0; //---設預設值
                showRec(); //---渲染畫面
                handler.post(updateTimer); //---開啟執行續
                runAuto_flag = true;
                //-----Menu群組-----
                menu.setGroupVisible(R.id.f0100_group01, visible: true);
                menu.setGroupVisible(R.id.f0100_group02, visible: false);
            }
        }
    }
};
```



三、Google 登入

登入畫面使用 Google Sign In，申請 Google OAuth2.0 用戶端 ID 憑證，取得使用者 E-mail、姓名、大頭照資料，來使用會員權限功能。

未登入時顯示登入 Button 及 Google logo，已登入會顯示會員的大頭照、登出 Button，及歡迎字樣。

問題

上架後的 Google Sign In 無法使用，
因為 SHA-1 憑證不一樣了，需改用上架
的 SHA-1 憑證。

程式碼

setupViewComponent 放在 onCreate 為
設定 Layout 元件，將需要監聽 Button、
ImageButton 宣告在這裡。

onClick 為 Button 的監聽，分別有
signIn（登入）、signOut（登出）、
revokeAccess（中斷連結）。



```
private void setupViewComponent() {  
    //-----  
    Intent intent=this.getIntent();  
    String mode_title = intent.getStringExtra( name: "class_title");  
    this.setTitle(mode_title);  
    //-----宣告  
    mStatusTextView = findViewById(R.id.status);  
  
    findViewById(R.id.sign_in_button).setOnClickListener(this);  
    findViewById(R.id.sign_out_button).setOnClickListener(this);  
    findViewById(R.id.disconnect_button).setOnClickListener(this);  
  
    GoogleSignInOptions gso = new GoogleSignInOptions  
        .Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)  
        .requestScopes(new Scope(Scopes.DRIVE_APPFOLDER))  
        .requestEmail()  
        .build();  
  
    mGoogleSignInClient = GoogleSignIn.getClient( activity: this, gso);  
  
    SignInButton signInButton = findViewById(R.id.sign_in_button);  
    signInButton.setSize(SignInButton.SIZE_STANDARD);  
    signInButton.setColorScheme(SignInButton.COLOR_LIGHT);  
}
```

在 onStart 加入檢查使用者登入狀態，
每次開啟頁面都會檢查，並會渲染登入或未登入畫面。

```
@Override  
public void onStart() {  
    super.onStart();  
    // Check if the user is already signed in and all required scopes are granted  
    GoogleSignInAccount account = GoogleSignIn.getLastSignedInAccount( context: this);  
    if (account != null && GoogleSignIn.hasPermissions(account, new Scope(Scopes.DRIVE_APPFOLDER))) {  
        updateUI(account);  
    } else {  
        updateUI( account: null);  
    }  
}
```

```
@Override  
public void onClick(View v) {  
    switch (v.getId()) {  
        case R.id.sign_in_button:  
            signIn();  
            break;  
        case R.id.sign_out_button:  
            signOut();  
            break;  
        case R.id.disconnect_button:  
            revokeAccess();  
            break;  
    }  
}
```



```

private void signIn() {
    Intent signInIntent = mGoogleSignInClient.getSignInIntent();
    startActivityForResult(signInIntent, RC_SIGN_IN);
}
private void signOut() {
    mGoogleSignInClient.signOut()
        .addOnCompleteListener( activity: this, new OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void> task) {
                // --START_EXCLUDE--
                updateUI( account: null);
                // [END_EXCLUDE]
                img.setImageResource(R.drawable.google_color); //還原圖示
            }
        });
}
private void revokeAccess() {
    mGoogleSignInClient.revokeAccess()
        .addOnCompleteListener( activity: this, new OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void> task) {
                // --START_EXCLUDE--
                updateUI( account: null);
                // --END_EXCLUDE--
                img.setImageResource(R.drawable.google_color); //還原圖示
            }
        });
}
}

```

此段為 signIn、signOut、revokeAccess 的語法，當被觸動時執行並重新渲染畫面。

```

private void updateUI(GoogleSignInAccount account) {
    GoogleSignInAccount aa = account;
    int aaa=1;
    if (account != null) {
        mStatusTextView.setText("歡迎 {account.getDisplayName()}");
    }
    //-----改變圖像-----
    User_IMAGE = account.getPhotoUrl();
    if(User_IMAGE==null){
        return;
    }
    img = (CircleImageView) findViewById(R.id.google_icon);
}

```

```

new AsyncTask<String,Void,Bitmap>() {
    @Override
    protected Bitmap doInBackground(String... params) {
        String url = params[0];
        return getBitmapFromURL(url);
    }
    @Override
    protected void onPostExecute(Bitmap result) {
        img.setImageBitmap(result);
        super.onPostExecute(result);
    }
}.execute(User_IMAGE.toString().trim());
//-----
//String g_id=account.getId();
findViewById(R.id.sign_in_button).setVisibility(View.GONE);
findViewById(R.id.sign_out_and_disconnect).setVisibility(View.VISIBLE);
} else {
    mStatusTextView.setText("未登入");

    findViewById(R.id.sign_in_button).setVisibility(View.VISIBLE);
    findViewById(R.id.sign_out_and_disconnect).setVisibility(View.GONE);
}
}
}

```

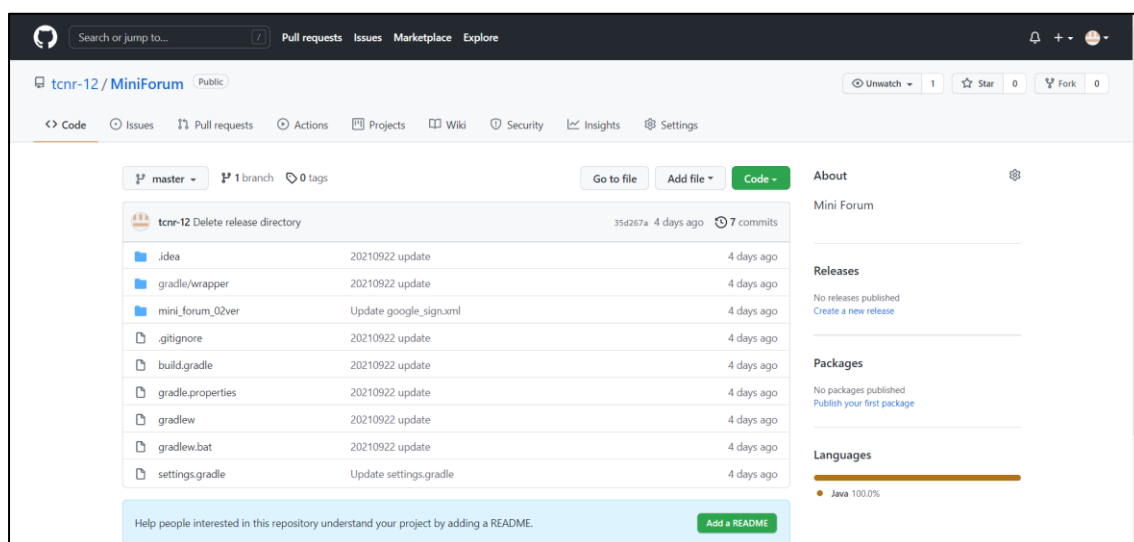

四、Github

Github 是紀錄程式碼的儲存工具，在團隊工作上，可以有效率確認每一次的更新內容，比較不會因為個人的失誤，造成團隊的損失。

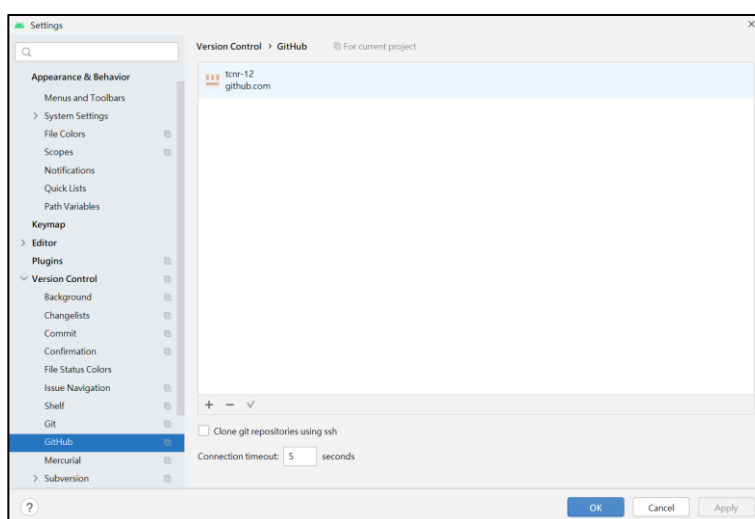
右圖為 Github 存放 APP 的 QR Code，下圖為 Github 網頁介面。



圖 3-1 GitHub QR Code



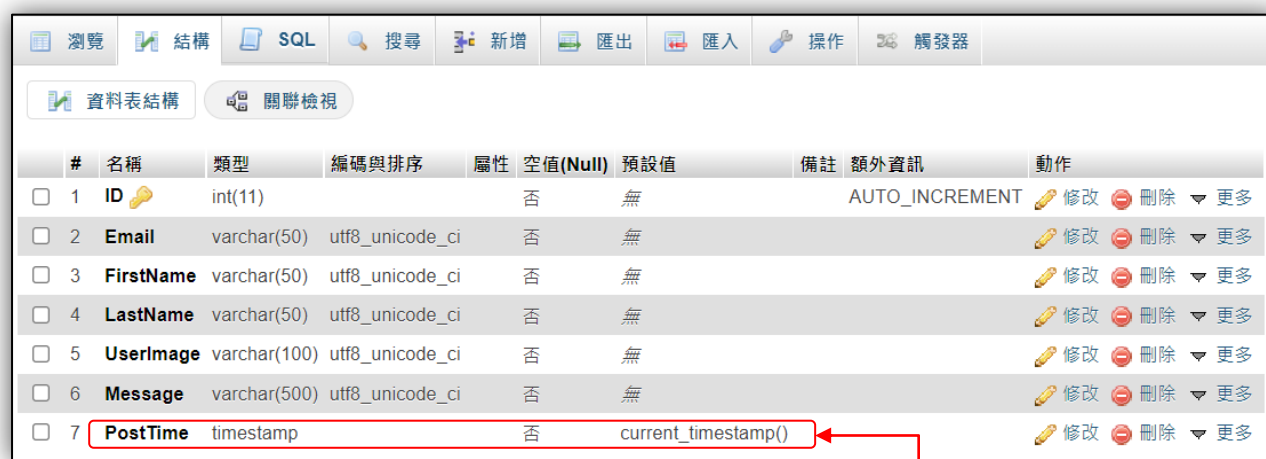
右圖為使用 Android studio 綁定 Github 介面，是以 Android studio 上傳更新至 Github。



肆、資料庫結構

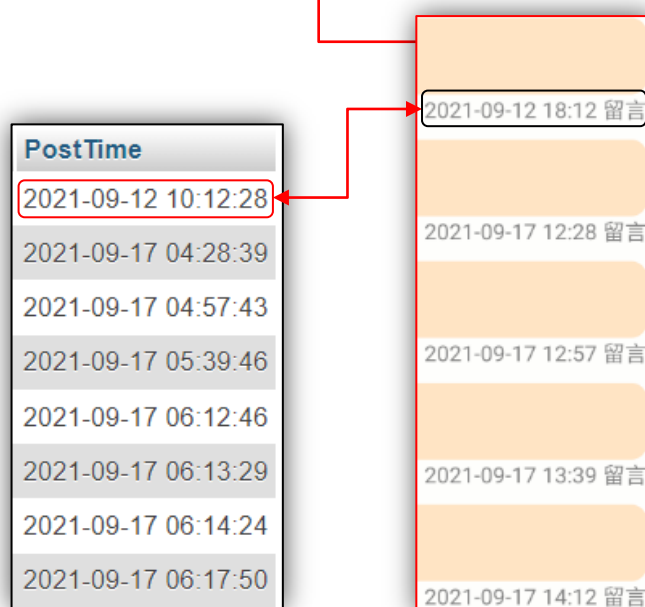
一、留言板

ID 是 primarykey 為不可重複的值，主要欄位用來記錄使用者資訊皆為 varchar，且不可為 null。



#	名稱	類型	編碼與排序	屬性	空值(Null)	預設值	備註	額外資訊	動作
1	ID	int(11)			否	無		AUTO_INCREMENT	修改 刪除 更多
2	Email	varchar(50)	utf8_unicode_ci		否	無			修改 刪除 更多
3	FirstName	varchar(50)	utf8_unicode_ci		否	無			修改 刪除 更多
4	LastName	varchar(50)	utf8_unicode_ci		否	無			修改 刪除 更多
5	UserImage	varchar(100)	utf8_unicode_ci		否	無			修改 刪除 更多
6	Message	varchar(500)	utf8_unicode_ci		否	無			修改 刪除 更多
7	PostTime	timestamp			否	current_timestamp()			修改 刪除 更多

PostTime 類型為 timestamp，是伺服器主機自動寫入主機時間的欄位，因此在使用時會有 8 小時的時差，需做時區轉換。



PostTime
2021-09-12 10:12:28
2021-09-17 04:28:39
2021-09-17 04:57:43
2021-09-17 05:39:46
2021-09-17 06:12:46
2021-09-17 06:13:29
2021-09-17 06:14:24
2021-09-17 06:17:50

2021-09-12 18:12 留言

2021-09-17 12:28 留言

2021-09-17 12:57 留言

2021-09-17 13:39 留言

2021-09-17 14:12 留言

二、每日一問

ID 是 primarykey 為不可重複的值，Today、Yesterday 為存放字串格式的日期，Topic、Answer 為存放題目與答案。

#	名稱	類型	編碼與排序	屬性	空值(Null)	預設值	備註	額外資訊	動作
1	ID	int(11)			否	無		AUTO_INCREMENT	修改 刪除 更多
2	Today	varchar(20)	utf8mb4_unicode_ci		否	無			修改 刪除 更多
3	TodayTopic	varchar(100)	utf8mb4_unicode_ci		否	無			修改 刪除 更多
4	Yesterday	varchar(20)	utf8mb4_unicode_ci		否	無			修改 刪除 更多
5	YesterdayTopic	varchar(100)	utf8mb4_unicode_ci		否	無			修改 刪除 更多
6	YesterdayAnswer	varchar(500)	utf8mb4_unicode_ci		否	無			修改 刪除 更多

showTopic 為渲染當日題目，取得系統時間並擷取年/月/日，代入 SQLite 搜尋 Today 欄位，Today 欄位避免重複的日期。

判斷是否有資料，取得第 0 筆資料，擷取題目欄位並渲染。

```
private void showTopic() {
    java.sql.Date curDate = new java.sql.Date(System.currentTimeMillis()); // 獲取當前時間
    str = formatter.format(curDate);

    if (str != null){
        String day = str.substring(0, 10); // 系統時間，擷取年/月/日
        recSet_F0101 = dbHper.getRecSet_query_F0101(day); // 代入SQLite搜尋

        ///---有資料才執行---
        if (recSet_F0101.size() != 0){
            ///---擷取SQLite資料---
            String[] f0101 = recSet_F0101.get(0).split( regex: "#"); // 取得第0筆資料
            t004.setText("每日一問："+f0101[2]); // 渲染當日題目
        }
    }
}
```

伍、台灣農產 B&Q（團隊）

一、簡介

「台灣農產 B&Q」是為了讓大家更加了解台灣各式各樣的農產品，透過結合遊玩問答遊戲提升使用者的興趣的同時，搭配農產知識庫達到學習效果。

台灣農產 B&Q 可以是一本電子書，同時也是一款益智遊戲！在認識台灣農產的同時，也可以輕鬆的學習。



圖 5-1 Google Play QR Code

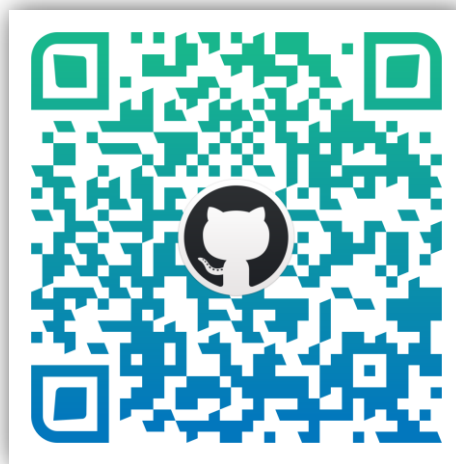


圖 5-2 GitHub QR Code

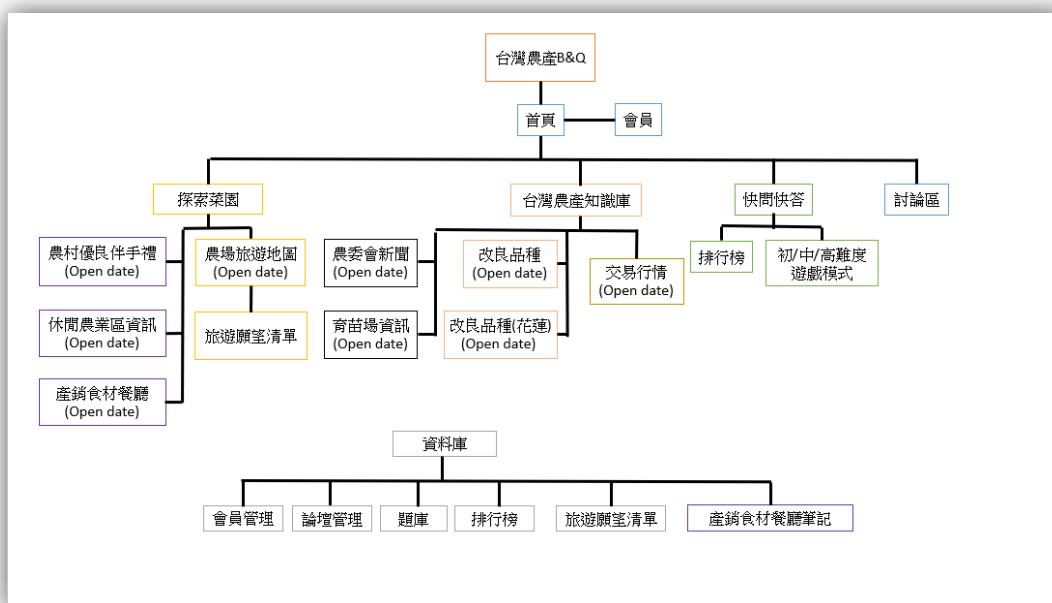
1. 適用範圍

我們目標使用客群為國中以上的年齡層，除了欲以遊戲的方式提升少年學習農產知識的興致以外，於農產知識庫設置農產品價格查詢、農業新聞、育苗資訊來供較為年長的用戶使用。

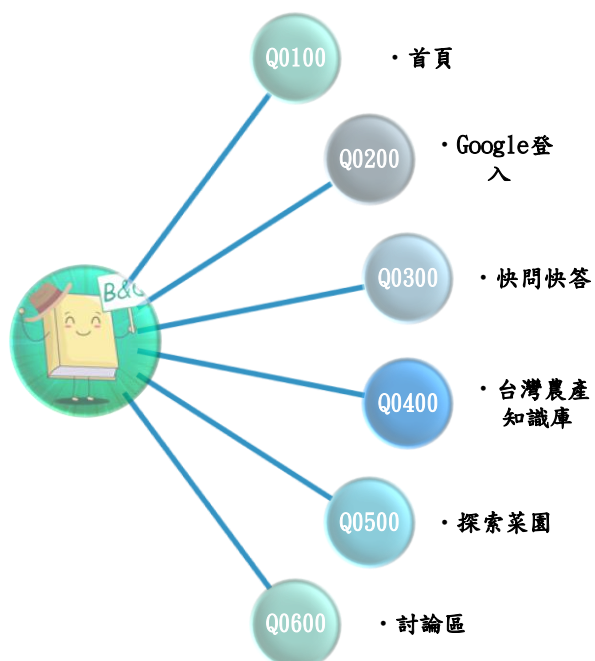
探索菜園可以提供使用者休閒農業區、農場旅遊、農產伴手禮等等的資訊，可以幫助使用者進行旅遊規劃。

可經過適當調整後用於各類教育知識方面，比如漁業、林業、花卉、飲食知識與各類市場交易資訊皆可適用。

2. 流程圖



3. 命名規則



二、 功能介面

1. 首頁 (Q0100)



開機動畫、開機音樂：

用動畫、輕快的音樂，讓使用者可以期待使用 APP。

滑動圖片：

利用 **WebView** 搭配 html，打造主題式的預覽圖，若要新增圖片，則在主機端新增即可。

公告欄：

在討論區發表的公告，使用 **SQL 語法** “SELECT * FROM Q0600 WHERE Category LIKE ‘公告’”，只擷取公告貼文，將標題文字渲染 TextView。

連結圖片：

用四張圖片，將主要功能顯示在首頁，可以讓使用者快速前往。

Menu：

將次要功能放在此處，有會員頁面、關於我們、結束應用程式。

2. 快問快答 (Q0300)



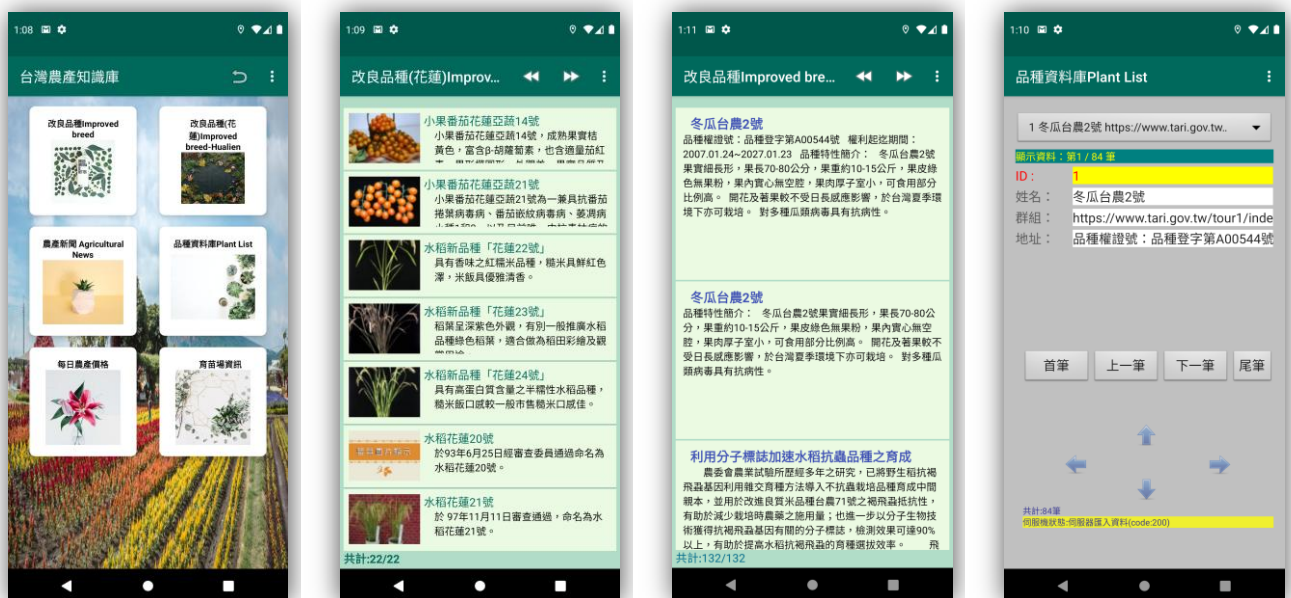
- 點選初級、中級、高級選項選擇難度，按下開始遊戲按鍵，開啟遊戲介面。
- 按下開始鍵，顯示題目，開始倒數計時，限時內作答。
- 在遊戲答題時會判斷目前的答題狀態，連續答對題目時，分數會有獎勵加成。
- 從中級難度開始，會根據連續答對題目的狀態縮短答題時間。
- 在高級難度中，只要連續答錯三題題目遊戲結束。
- 在遊戲結束時，可以選擇結束遊戲或是重新遊玩遊戲，並會根據是否有登入 google 會員，得以行使儲存分數的功能，將分數儲存於排行榜上。

3. 台灣農產知識庫 (Q0400)

利用政府資料公開平台結合相關資訊整合在此資料庫，包含改良品種、每日農產價格、農新聞、種苗資訊。

✚ 使用『CardView』介面，直覺點擊以進入主題。

✚ Menu: 可停止、播放背景音樂。



台灣改良品種

- ✚ 使用 OpenData 資料庫，藉由 OKHttp 第三方套件來串接。
- ✚ 使用 RecyclerView 設計介面。
- ✚ Menu 選單可選擇第一筆、往上一百筆、往後一百筆、最後一筆。

台灣農作物的改良品種介紹，此區使用政府提供的 OpenData，有簡介及圖片，可以了解及學習。



每日農產價格、農新聞、種苗資訊

- 使用 OpenData 資料庫，藉由 OKHttp 第三方套件來串接。
- 使用 RecyclerView 設計界面。
- Menu 選單可選擇第一筆、往前一百筆、往後一百筆、最後一筆。
- 電話聯絡：如有需要諮詢，點擊電話連絡可執行電話功能。
- 地圖搜尋：使用 google map (APP) 搜尋農場位置，並提供路線、交通時間。

每日農產價格，按下按鈕選擇日期，點擊地圖上的地名或是輸入搜尋欄，即可顯示當日交易行情。

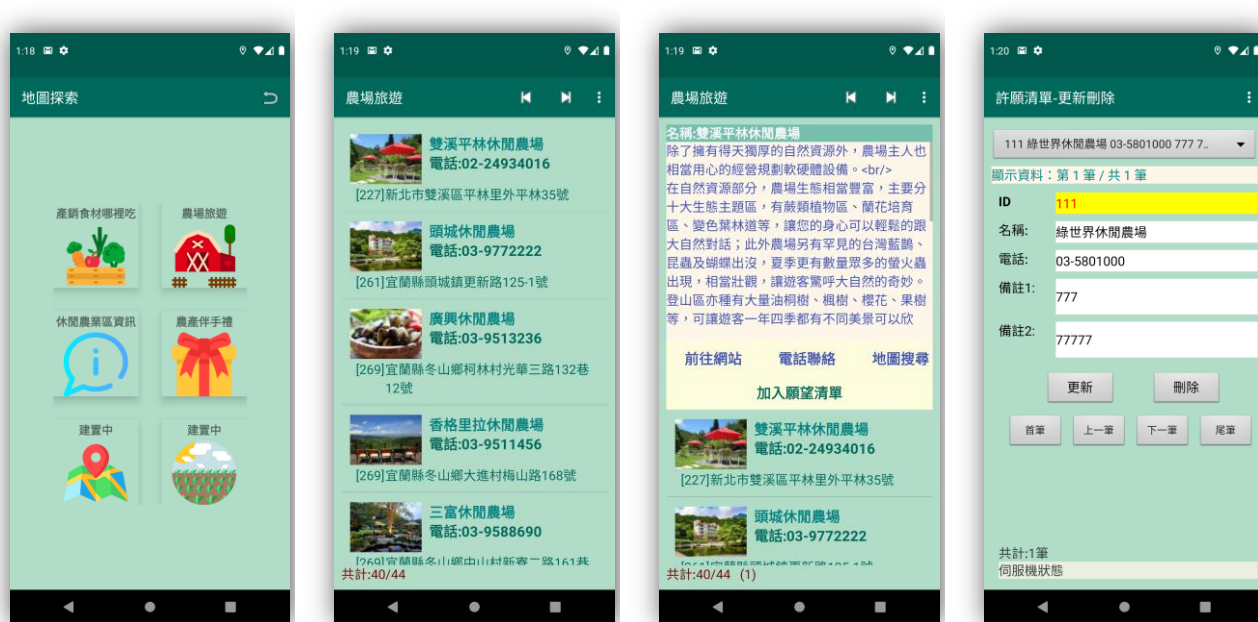
農新聞，一進入便會馬上詢問使用者是否下載最近的農業新聞時事，點擊有興趣的新聞報導標題，便會於上方展開顯示新聞內容。

種苗資訊，在這裡提供想要購買幼苗、種子管道資訊讓消費者了解，點取店家資訊，下方有附上電話與地圖導航。

4. 地圖探索 (Q0500)

利用政府資料公開平台結合相關資訊整合在此資料庫，包含農場旅遊、休閒農業區、農產伴手禮。

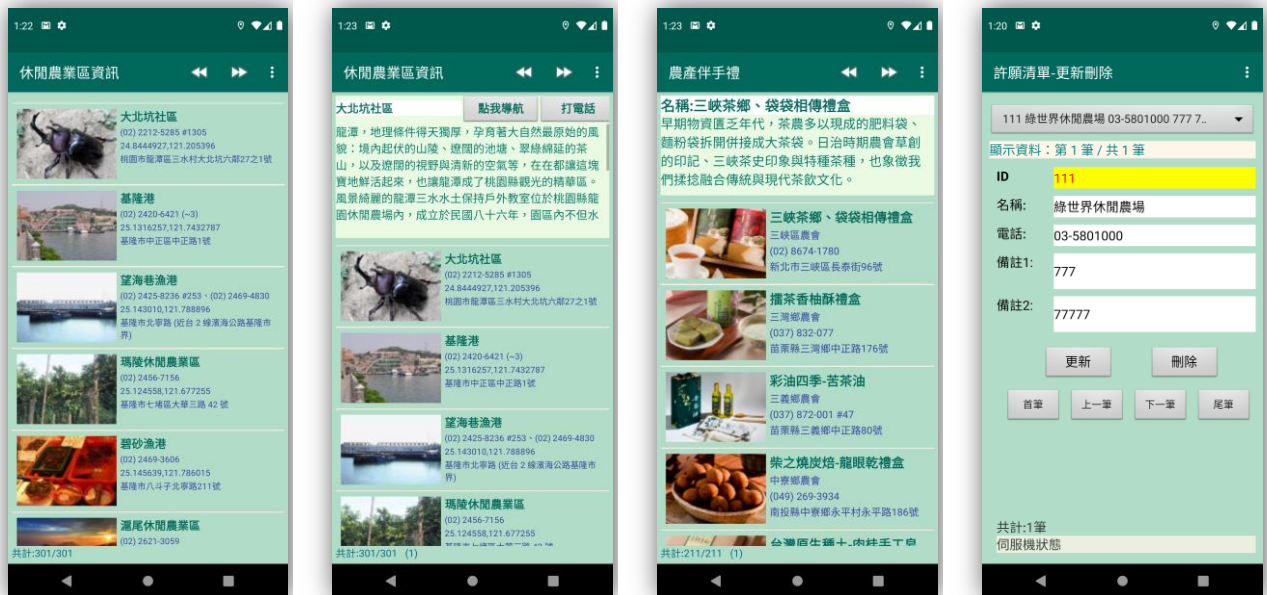
✚ 使用『CardView』介面，直覺點擊以進入主題。



農場旅遊

農場旅遊介紹各種休閒農場，使用者可透過下拉功能導入資料，點擊閱讀簡介。

- ✚ 使用 OpenData 資料庫，藉由 OKHttp 第三方套件來串接。
- ✚ 使用 RecyclerView 設計界面。
- ✚ Menu 選單可選擇第一筆、往前一百筆、往後一百筆、最後一筆。
- ✚ 前往網站：點擊後，可前往該農場提供的網站。
- ✚ 電話聯絡：如有需要諮詢，點擊電話連絡可執行電話功能。
- ✚ 地圖搜尋：使用 google map (APP) 搜尋農場位置，並提供路線、交通時間。
- ✚ 加入願望清單：使用者可用來記錄有興趣的農場，並添加備註。



產銷食材餐廳筆記、休閒農業資訊、農產伴手禮

- 使用 OpenData 資料庫，藉由 OKHttp 第三方套件來串接。
- 使用 RecyclerView 設計界面。
- Menu 選單可選擇第一筆、往前一百筆、往後一百筆、最後一筆。
- 電話聯絡：如有需要諮詢，點擊電話連絡可執行電話功能。
- 地圖搜尋：使用 google map (APP) 搜尋農場位置，並提供路線、交通時間。

看見食材「從產地到餐桌」美味之旅，外食族可以輕鬆追溯料理的食材來源，增添消費者享用料理時的安心感，落實「產地地消」的理念。

「產銷食材餐廳筆記」蒐集台灣優良的溯源餐廳，連結 MySQL 資料庫，可以新增、刪除、修改、查詢。

5. 討論區 (Q0600)



提供使用者發表文章，可以提問題，也可以發表小知識。資料庫是藉由 **OKHttp** 第三方套件串接，讓 Client 端向 Server 端抓取資料，配合執行續讓 Client 端持續更新。

🚩 搜尋：

上方的下拉選單 spinner 存放文章分類，內容有最新、公告、問題、知識、趣味等，點擊搜尋 Button，執行 SQL 語法篩選分類文章，並執行渲染。

🚩 新增：

可以新增文章，當點擊新增 Button，執行 SQL **insert 語法**，將使用者訊息及文章內容寫到 MySQL，並重新抓取 MySQL 資料，更新、渲染頁面。

🚩 編輯：

可以開啟編輯模式，修改、刪除文章或回應，使用編輯執行 SQL 語法，搜尋文章 ID，渲染編輯的頁面，將修改過的內容 **Update 到 MySQL**，並重新抓取 MySQL 資料，更新、渲染頁面。

🚩 Menu：

次要功能，可以開啟、取消編輯模式以及前往會員登入頁面，**Menu 使用群組**，當點擊編輯時，顯示取消編輯，隱藏編輯、會員頁面。

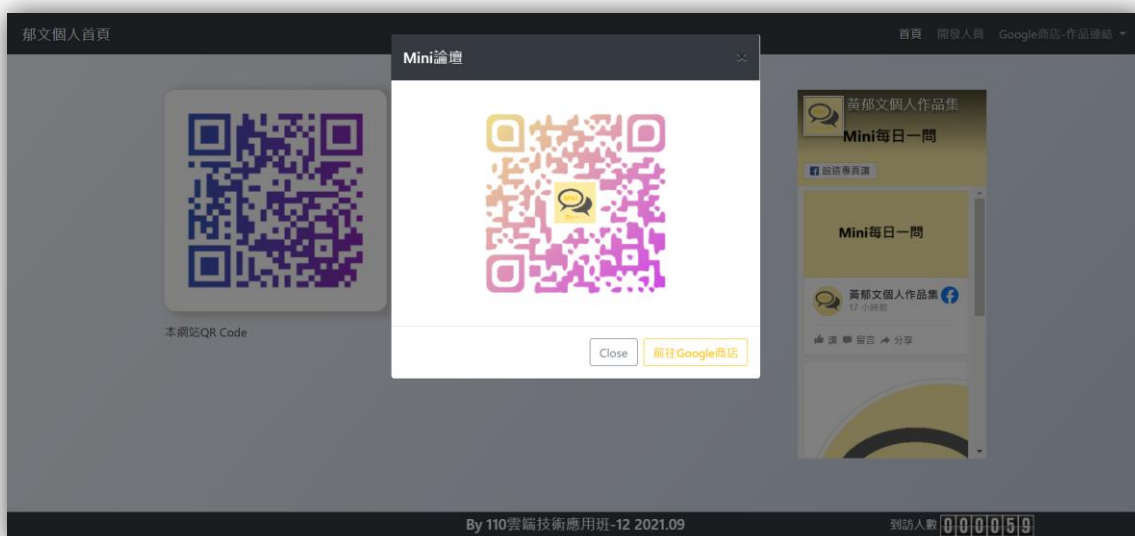
陸、補充作品(個人)

一、 個人網站

以上課所學的 Bootstrap 前端套件為設計框架，使用 Navbar 作為上方功能表，Grid System 作為介面排版，Button 作為連結網站的按鍵，fixed-bottom 作為下方功能表。再從網路資源，找到 QR Code、計數器、FaceBook 掛件搭配使用。



下方為使用 Modal 彈出對話窗，顯示 APP 的 QR Code 與前往連結的 Button，未來還會再增加功能，以豐富網站的多樣性。



二、SPA 網站 (Single Page Application)

一樣是使用 Bootstrap 作為前端套件，上方功能表是使用 Navbar，往下使用 Carousel 功能做圖片輪播，以及使用 Grid System 作為介面排版。

滑鼠滾輪下滑，還搭配 Animate.css 掛件，設計了淡出、閃爍、滑動進入、彈跳等動畫，讓網站增加可看性。



下圖為使用 Google 地圖的嵌入網址功能，再搭配 Forms 表單，可以做為問卷、意見調查等介面。



三、會員管理系統網站

使用 Table 來設計排版，並且使用 Ajax 串接資料庫，可以讀取、新增、刪除、更新資料，使用 Cookie 暫存使用者登入狀態，並可搭配其他管理系統，讓會員功能更加完整。

會員管理控制台(a123456已登入)														登出
編號	帳號	姓名	性別	電話	信箱	出生日期	平均預算	平均花費	付款方式	消息來源	寄送廣告	註冊日期	功能	
31	admin	測試人員	男生	09788787878	78787878@787878787878	2021-09-14	20000	5001元以上	信用卡	電視廣告,報章雜誌,廣播,網路,親友	同意	2021-09-14 12:02:11.452404	更新 刪除	
30	test0911	測試帳號	男生	0915151515	15155@222222	2021-09-11	20000	5001元以上	信用卡	電視廣告,報章雜誌	同意	2021-09-11 06:22:41.043878	更新 刪除	
29	h123456	大G	男生	0915151555	7878@26262626	2008-02-02	3500	1000~5000元	現金	電視廣告,網路,親友	同意	2021-08-02 01:59:24.918819	更新 刪除	
28	g123456	小G	女生	0989898989	4545@54512126	2009-02-11	3000	501~1000元	其他電子錢包	電視廣告,報章雜誌,網路	不同意	2021-07-29 05:49:35.378784	更新 刪除	
27	f123456	小F	女生	093232323	3323@1414141	2012-02-26	1000	500元以下	悠遊卡	電視廣告,報章雜誌,網路	不同意	2021-07-26 02:30:38.098468	更新 刪除	
26	e123456	小E	男生	0989898989	89898@8989898	2006-05-03	2500	501~1000元	悠遊卡	電視廣告,網路,親友	不同意	2021-07-26 02:28:13.556445	更新 刪除	
25	d123456	小D	男生	096262626	62626@2323232	2000-10-23	4000	1000~5000元	信用卡	電視廣告,報章雜誌,網路	同意	2021-07-26 02:23:19.425576	更新 刪除	
24	c123456	小C	女生	0925252525	8888@23232323	2021-07-18	1500	501~1000元	悠遊卡	電視廣告,報章雜誌	不同意	2021-07-26 02:22:13.661665	更新 刪除	

註冊頁面使用 input、label 屬性，將會員註冊時，常用到的資料作配置，如帳號、密碼、姓名、性別、電話、E-mail、生日等，當點擊送出鍵，將串接資料庫並寫入，讓會員使用相關功能。

購物中心-會員註冊

帳號

帳號介於5到10個字

密碼

密碼要大於8個字

確認密碼

密碼同上

基本資料

姓名

姓名不可空白

性別

☒ 男生

連絡電話

電話不可空白

E-mail

信箱不可空白

出生日期

年 / 月 / 日

購物資訊

購物平均預算

1000

四、串接 Open Data 網站

使用「政府資料開放平台」的 json 格式資料，使用 Bootstrap 的 Card 設計頁面，將 json 用 Ajax 做串接，並且渲染畫面，下圖為地方美食介紹。



右圖為使用 media screen 及 Bootstrap 的 Grid System，來做 RWD 的網站介面，依照不同螢幕的寬度，讓系統判斷該使用哪種介面配置，達到電腦、平板、手機共用使用同一個網頁。



柒、實作心得

在設計 APP 的過程中，總會遇到許多問題，如跑版、效果出不來、閃退等，到目前為止，還是一一克服了各種不同的狀況。

從一開始決定要設計哪一種 APP 開始，開始有了困惑，APP 使用上有很多分類、功能，那我該選哪一種？在思考一陣子後，在自己玩的手遊看到一點想法，那就是聊天室，不管大型或小型的遊戲都需要團隊來完成，那我就以聊天室為基礎，發想一個小型的主題式社交 APP，就是「Mini 每日一問」。

初次設計版面時，跑版、設定沒有效果是最常發生的問題，因為物件的寬度屬性，或是物件擺放的位置，會造成重疊、跑出版面外等問題。在 Java 語法上，也有不少問題產生，例如在抓取 MySQL 資料庫時，會有抓不到資料的狀況發生，此時就要判斷哪個部分發生問題，Java？PHP？資料庫？往往卻是一個小問題，讓人花費時間 debug，但當資料抓到了，總會有一種，辛苦努力 debug 後，成功了的成就感！

剛接觸程式設計時，就有一種感覺，程式軟體每過一天，都會有不同的變化，尤其是程式更新的時候，換言之，就是沒有學完的一天，那我就應該對程式抱好奇心，每天都期待接觸到不一樣的東西，當程式軟體因為功能不完整，而有了跳躍性的進階，我們更應該持續精進，這也是對我自己的期許。