低價拍賣網 我們拍賣吧!

後端工程師企業訓練班 期末成果發表

組員: 丁彥云 吳宜謙 栗永欣 吳紀瑄

後端工程師企業訓練班 第四組

講師 簡志璁 老師

目錄

1. 摘要.....3
2. 動機與前言.....3
3. 系統位置.....3
4. 系統功能與架構.....4
a. 系統架構.....4
b. 操作流程.....4
c. 資料庫.....5
5. 系統流程與網站成果.....5
a. 使用者帳戶.....5
b. 會員系統.....13
c. 商品資訊.....28
d. 拍賣系統.....30
e. 管理系統.....39
6. 實作困難點.....39
7. 未來展望.....40
8. 甘特圖.....40

1. 摘要

這是一個提供使用者有趣的低價拍賣系統，為了使提供商品及參與拍賣的會員，能夠更加便利的參與競拍，我們提供了有趣的概念，以及相對完善的功能，會員都可以在會員訂單看到自己的上架商品以及下標訂單。此外使用者可以在此網站平台，競拍到自己心儀的商品，並且藉由全新或二手商品的互相拍賣，提供一個資源循環平台的概念。

2. 動機與開發規劃及環境

一、動機

1. 隨著購物型態的改變，網路買賣已經成為現代嶄新的經營型態，現今許多企業會以網路做為商品行銷的新通路。
2. 年輕的消費族群頗為喜愛網路消費的模式，因為網路的便利性，可以同時滿足業者與消費者雙方的需求。
3. 有鑒於此，使用不同以往的價高者得的方式進行拍賣行為，可以提升參與競標人的好奇心與買到便宜價格心態，用以獲得更好之商業拍賣行為。

二、開發規劃及環境

1. ASP.NET 設計模型為主要架構
2. 使用 C#、SQL 撰寫後端程式與資料庫建立
3. 使用 HTML、CSS、JavaScript、套用 bootstrap 樣板
4. 利用 Azure 雲端建立網站的實行
5. 使用 GitHub 專案管理進行團隊的協作

3. 系統位置

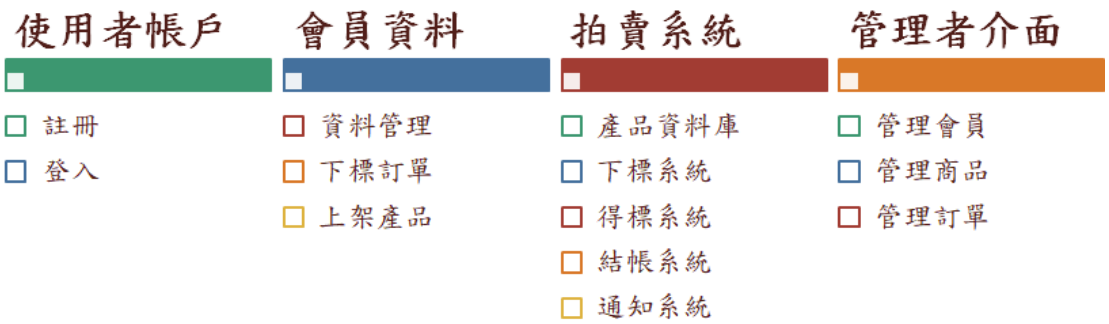
系統網址　https://letsbiddingweb.azurewebsites.net/Home

系統 QRCODE

4. 系統功能與架構

一、系統架構

此專案系統規劃為主要四個功能系統，使用者帳戶、會員資料、拍賣系統、管理者介面，分別對應功能為會員的註冊與登入、會員相關資料的查詢(會員資料與會員訂單)、商品資訊與下標(商品上架與得標邏輯)、網站管理者(新增修改刪除商品與會員)，詳細功能於下圖一所示。



圖一、系統架構

二、操作流程

網站操作流程為分為四條路徑:

1. 會員註冊 ：

訪客非會員，可進行瀏覽商品，但是需要進行下標動作時，將會導向會員登入系統，才能進行商品下標，因此新訪客則需進行會員註冊後，方可登入後下標商品。

2. 會員登入 ：

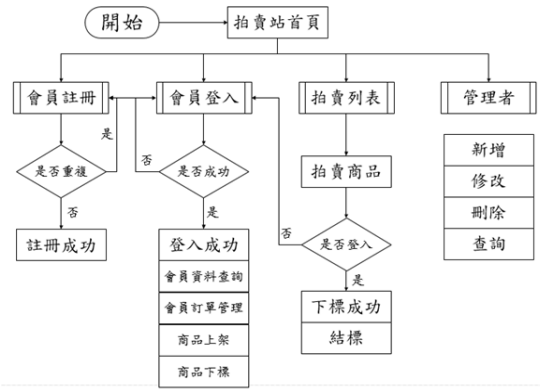
訪客為會員，則可進行登入會員後，進行會員資料查詢與修改，包括之前下標過以及上架過之商品資料，以及參與商品的上架、下標與結標等等功能。

3. 拍賣下標 ：

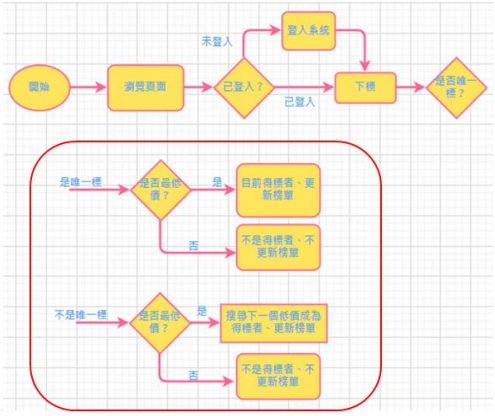
訪客為會員登入後，下標為進行判定，先判定標價錢為唯一價錢標，再行判定是否為最低價錢。若此標為唯一標，並且為最低價，會員為該商品暫時得標者，將由此會員名稱展示於商品頁，待下名下標者替換得標榜單，直至該商品上架時間結束。非唯一標及最低價，則扣除會員帳戶手續費後，不更新商品得標榜單。如下圖三所示。

4. 管理介面 ：

若訪客為管理者，則可直接登入管理者系統，可進行管理會員、商品以及結標資訊。如下圖二所示。



圖二、操作流程



圖三、商品拍賣邏輯

三、資料庫

資料庫主要分為三大資料：分別為管理者、會員資料、商品資料。其中管理者包含管理者詳細帳戶密碼等等紀錄資訊，會員資料包含會員帳號密碼以及錢包等等與拍賣系統相關功能，商品資料則包含商品介紹、下標紀錄、得標者以及收件者資訊。相關詳細項目如下圖四所示。

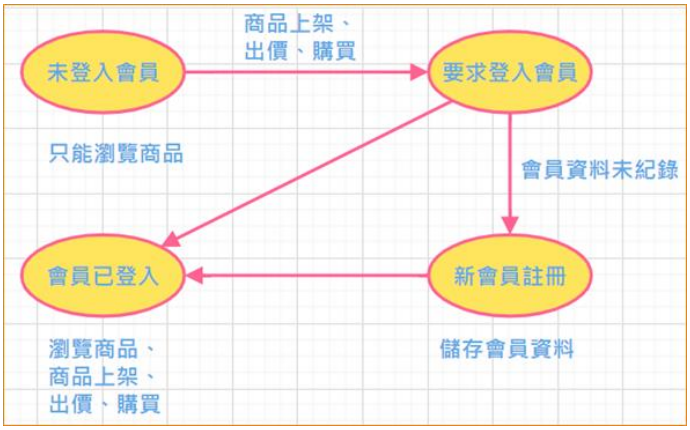
圖四、資料庫項目

5. 系統流程與網站成果

一、使用者帳戶

A. 會員流程圖

若訪客未登入會員，可進行瀏覽商品，當進行商品上架與下標動作時，將會導向要求會員登入，若無相關資料，將進行會員註冊後，將會員資料寫入資料庫，方可登入後進行上架下標商品。如下圖五所示。



圖五、會員流程圖

B. 會員註冊

當訪客要進行會員註冊時，可點選網頁上方的標籤列(免費註冊)，分別為首頁、會員登入、會員註冊、聯絡我們、管理介面。如下圖所示。

此程式部分：memberRegisterForm.aspx.cs

為利用 LinkButtoon 將點擊行為跳轉為應該導向之頁面，但因會員登入與否必須展現不同之標籤列，因此先將相關性標籤使用 Visible 標籤，先行隱藏，並利用 Session 紀錄會員是否登入狀態，才顯示對應之連結標籤列。如下圖所示。

```
<asp:LinkButton ID="home" runat="server" class="icon-home" OnClick="home_Click">首頁</asp:LinkButton>
<asp:LinkButton ID="member_info" runat="server" Visible="false" class="icon-user" OnClick="member_info_Click">會員資料</asp:LinkButto
<asp:LinkButton ID="order_info" runat="server" Visible="false" class="icon-edit" OnClick="order_info_Click" >訂單查詢</asp:LinkButton
<asp:LinkButton ID="my_info" runat="server" Visible="false" class="icon-user" OnClick="my_info_Click">我的會員</asp:LinkButton>
<asp:LinkButton ID="register" runat="server" Visible="false" class="icon-edit" OnClick="register_Click">免費註冊</asp:LinkButton>
<asp:LinkButton ID="contantus" runat="server" class="icon-envelope" OnClick="contantus_Click">聯絡我們</asp:LinkButton>
<asp:LinkButton ID="manager" runat="server" Visible="false" class="icon-lock" OnClick="manager_Click">管理後臺</asp:LinkButton>
<asp:LinkButton ID="logout" runat="server" Visible="false" class="icon-remove" OnClick="logout_Click">登出</asp:LinkButton>
```

會員註冊頁面進行導向後，可見到各式資料欄位進行輸入，包含資料庫中相關性資料，其中資料檢查機制，包含:

- 避免會員資料未填入之提醒功能，包含帳號、密碼、密碼確認等等資訊，若未填入就進行送出，將會跳出警告視窗。當按下送出時，將會先進行空格判斷，程式碼如下圖所示。

```
0 個參考
protected void RegisterBtn_Click(object sender, EventArgs e)
{
    if (_user_name.Text == "")
    {
        Response.Write($"<script>alert('請輸入帳號');</script>");
    }

    else if (_memberPassword.Text == "")
    {
        Response.Write($"<script>alert('請輸入密碼');</script>");
    }

    else if(_ConfirmPassword.Text == "")
    {
        Response.Write($"<script>alert('請確認密碼');</script>");
    }
    else if(_name.Text == "")
    {
        Response.Write($"<script>alert('請確認會員名稱');</script>");
    }

    else if(_email.Text == "")
    {
        Response.Write($"<script>alert('請確認會員Email');</script>");
    }

    else if(_cellphoneNo.Text == "")
    {
        Response.Write($"<script>alert('請確認會員電話');</script>");
    }
```

- 密碼確認必須輸入兩次皆為相同正確，若不同則會跳出警告視窗。同樣的當按下送出時，將會先進行密碼是否相同判斷，程式碼如下圖所示。

```
else if(_memberPassword.Text != _ConfirmPassword.Text)
{
    Response.Write("<script>alert('密碼不正確！請重新輸入');</script>
}
```

- 若已有會員註冊相通帳戶名稱，也會進行跳出警告視窗，重新進行申請。當按下送出時，將會先進行連結資料庫，並搜尋資料庫中是否找的到此帳戶，若找的到將判定為False，跳出提醒，程式碼如下圖所示。

```
protected bool bSQLDB_checkAccount(string connString, string _guiName)
{
    bool bFound = false;
    string s_data = System.Web.Configuration.WebConfigurationManager.ConnectionStrings[connString].ConnectionString;

    //new一個SqlConnection物件，是與資料庫連結的通道(其名為Connection)，以s_data內的連接字串連接所對應的資料庫。
    SqlConnection connection = new SqlConnection(s_data);

    // bug1: SQL content
    string sql_statement = $"select * from Member where user_name='{_guiName}'";

    // bug2: sqlText
    //new一個SqlCommand告訴這個物件準備要執行什麼SQL指令
    SqlCommand Command = new SqlCommand(sql_statement, connection);

    //與資料庫連接的通道開啟
    connection.Open();

    //new一個DataReader接取Execute所回傳的資料。
    SqlDataReader Reader = Command.ExecuteReader();

    //檢查是否有資料列
    if (Reader.HasRows)
    {
        //使用Read方法把資料讀進Reader，讓Reader一筆一筆順向指向資料列，並回傳是否成功。
        if (Reader.Read())
        {
            bFound = true;
        }// if (Reader.Read())

    }// if (Reader.HasRows) login name match
    else
    {
        bFound = false;
    }// if (Reader.HasRows) login name mismatch
    //關閉與資料庫連接的通道
    connection.Close();
    return bFound;
```

```
else
{
    //check memeber account not repeat
    if (bSQLDB_checkAccount("Sale_net_Jun22_2021ConnectionString2", _user_name.Text)
        == true)
    {
        Response.Write("<script>alert('帳號已有會員登記，請重新選個好聽的名稱');</script>");
    }
```

● 若會員資料註冊成功，將會顯示會員註冊成功，等提醒視窗。

當按下送出時，將會先進行連結資料庫，並輸入寫入會員資料等 SQL 語法，判斷是否成功寫入，會員註冊完成，後跳轉回前頁(利用 Session 紀錄前頁)，若無前頁則回首頁，程式碼如下圖所示。

```
else
{
    SQLDB_write("Sale_net_Jun22_2021ConnectionString2");
    // to get BusID from BusAccountTable
    SQLDB_verify("Sale_net_Jun22_2021ConnectionString2", _user_name.Text);
}
```

```
protected void SQLDB_write(string connString)
{
    string s_data = System.Web.Configuration.WebConfigurationManager.ConnectionStrings[connString].ConnectionString;

    //new一個SqlConnection物件，是與資料庫連結的通道(其名為Connection)，以s_data內的連接字串連接所對應的資料庫。
    SqlConnection connection = new SqlConnection(s_data);

    //與資料庫連接的通道開啟
    connection.Open();

    // _File
    SqlCommand sql_insert_cmd = new SqlCommand("insert into Member(user_name,password,name,mail,mobile_phone,year,month,date,address,status,balance)" +
        " values(@user_name,@password,@name,@mail,@mobile_phone,@year,@month,@date,@address,@status,@balance);", connection); //SQL語句

    sql_insert_cmd.Parameters.Add("@user_name", SqlDbType.Text);
    sql_insert_cmd.Parameters["@user_name"].Value = _user_name.Text;

    sql_insert_cmd.Parameters.Add("@password", SqlDbType.Text);
    sql_insert_cmd.Parameters["@password"].Value = _memberPassword.Text;

    sql_insert_cmd.Parameters.Add("@name", SqlDbType.Text);
    sql_insert_cmd.Parameters["@name"].Value = _name.Text;

    sql_insert_cmd.Parameters.Add("@mail", SqlDbType.Text);
    sql_insert_cmd.Parameters["@mail"].Value = _email.Text;

    sql_insert_cmd.Parameters.Add("@mobile_phone", SqlDbType.Text);
    sql_insert_cmd.Parameters["@mobile_phone"].Value = _cellphoneNo.Text;

    sql_insert_cmd.Parameters.Add("@year", SqlDbType.Text);
    sql_insert_cmd.Parameters["@year"].Value = _birthYear_list.Text;

    sql_insert_cmd.Parameters.Add("@month", SqlDbType.Text);
    sql_insert_cmd.Parameters["@month"].Value = _birthMonth_list.Text;

    sql_insert_cmd.Parameters.Add("@date", SqlDbType.Text);
    sql_insert_cmd.Parameters["@date"].Value = _birthDate_list.Text;


    sql_insert_cmd.Parameters.Add("@address", SqlDbType.Text);
    sql_insert_cmd.Parameters["@address"].Value = _address.Text;

    sql_insert_cmd.Parameters.Add("@status", SqlDbType.Text);
    sql_insert_cmd.Parameters["@status"].Value = _status.Text;

    sql_insert_cmd.Parameters.Add("@balance", SqlDbType.Int);
    sql_insert_cmd.Parameters["@balance"].Value = 100;

    sql_insert_cmd.ExecuteNonQuery();

    //關閉與資料庫連接的通道
    connection.Close();

    Response.Write($"<script>alert('寫入會員資料庫成功');</script>");
```

● 以上會員註冊頁面提醒與註冊功能，網站畫面如下圖六所示。

圖六、會員註冊檢查機制

● 若從會員登入網頁進行新會員註冊，則會先行進行會員帳戶是否重複判定後，跳轉會員註冊頁面，並將前些頁面之判斷過之會員帳戶名，輸入於帳戶空格內。程式碼如下，當按下建立新帳號時，將會連結資料庫，並進行判定，是否在資料庫找到此帳戶名稱，若無則導向會員註冊頁面，進行註冊。網站畫面如下圖七所示。

```
protected void _CreaateMemberBtn_Click(object sender, EventArgs e)
{
    if (_memberAccount.Text == "")
    {
        Response.Write($"<script>alert('請寫入帳號名稱');</script>");
        return;
    }

    // if the e-mail exists, reply the account has used
    if (bSQLDB_ifmatch("Sale_net_Jun22_2021ConnectionString2", _memberAccount.Text)
        == true)
    {
        Response.Write("<script>alert('帳號已有會員登記，請改另一個帳號名稱 ');</script>");
    }
    else
    {
        Session["NewMemberAccount"] = _memberAccount.Text;

        Response.Redirect("memberRegisterForm.aspx");
    }
```

圖七、會員註冊由會員登入頁面判定

C. 會員登入

當訪客要進行會員登入時，可點選網頁上方的標籤列(我的會員)，分別為首頁、會員登入、會員註冊、聯絡我們、管理介面。如下圖所示。

- 程式部分：memberLoginForm.aspx.cs

判斷該會原是否已經存在會員資料庫中:

讀取頁面的會員帳號及密碼資料，並與會員資料庫中的內容比對；若會員帳號在會員資料庫中出現，代表會員已登入系統。如下圖所示。

```
protected bool bSQLDB_verify(string connString, string _guiName)
{
    bool bFound = false;
    string s_data = System.Web.Configuration.WebConfigurationManager.ConnectionStr

    //new一個SqlConnection物件，是與資料庫連結的通道(其名為Connection)，以s_data內的連
    SqlConnection connection = new SqlConnection(s_data);

    // bug1: SQL content
    string sql_statement = $"select * from Member where user_name ='{_guiName}'";

    // bug2: sqlText
    //new一個SqlCommand告訴這個物件準備要執行什麼SQL指令
    SqlCommand Command = new SqlCommand(sql_statement, connection);

    //與資料庫連接的通道開啟
    connection.Open();
```

- 登入成功會跳轉前頁面或是首頁 ：

若會員帳號在會員資料庫中出現代表會員已登入系統，並將頁面轉回登入畫面前的頁面。如下圖所示。

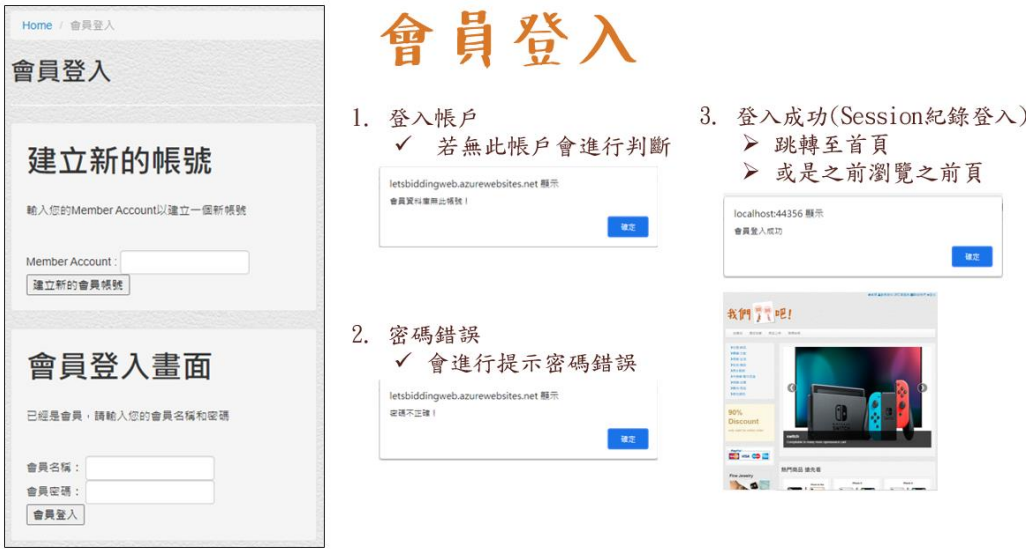
```
protected void LoginBtn_Click(object sender, EventArgs e)
{
    if (bSQLDB_verify("Sale_net_Jun22_2021ConnectionString2", _loginName.Text)
        == true)
    {

        // goto the web page after member logging
        if(Session["logged_to_page"] != null){
            string pageAfterLogging = (string) Session["logged_to_page"];

            Response.Redirect(pageAfterLogging);
        }
        else
        {
            Response.Redirect("Home.aspx");
        }
    }

}// protected void LoginBtn_Click(object sender, EventArgs e)
```

- 會員登入若密碼輸入錯誤 ：

若會員帳號在會員資料庫中比對密碼錯誤，會進行警示標籤提醒。程式碼如下圖所示。

```
        if (Reader["Password"].ToString() == Request.Form["_loginPassword"])
        {
            //DataReader讀出欄位內資料的方式，通常也可寫Reader[0]、[1]...[N]代
            Session["name"] = Request.Form[_guiName];  //"_BusName"
            int nMemberID = (int)(Reader["member_ID"]);
            Session["member_ID"] = nMemberID;

            Response.Write("<script>alert('會員登入成功');</script>");
            Session["Login"] = "logged";

            bFound = true;
            // Server.Transfer("test2.aspx");
        }
        else
        {
            Response.Write("<script>alert('密碼不正確！');</script>");
            bFound = false;
        }

    }// if (Reader.Read())

}// if (Reader.HasRows) login name match
else
{
    Response.Write("<script>alert('會員資料庫無此帳號！');</script>");
    bFound = false;
```

● 會員登入等相關提醒與系統流程網站畫面，如下圖八所示



圖八、會員登入頁面判定

二、會員系統

A. 會員資料管理

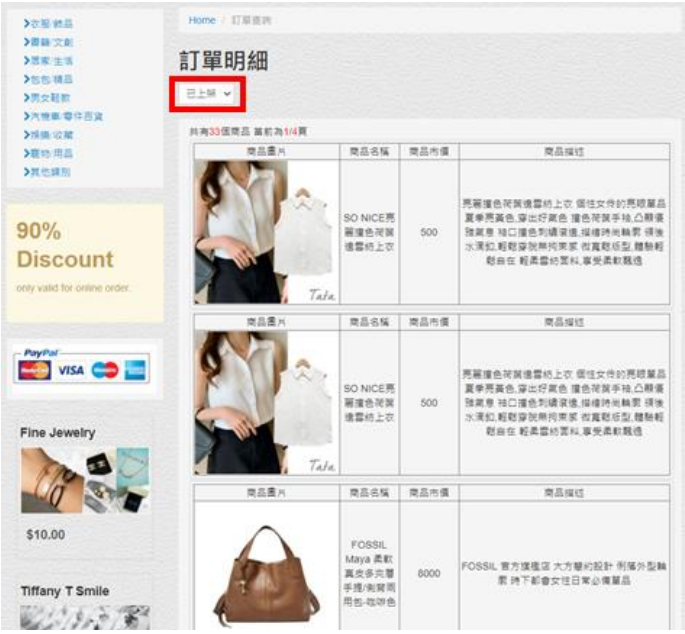
▲會員資料頁面會自動帶入登入者的資料

Home / 會員資料
會員資料
會員資料修改
會員帳號： tina
會員密碼： 0924
名　稱： tina
電子郵件： tina
請在電子郵件地址中包含「@」。「tina」未包含「@」。
居住地址： 新北市土城區明峰街
會員狀態： 正常
會員餘額： 100
資料修改
儲存資料
會員編號： 18
▲電子郵件格式會防呆

B. 會員訂單查詢

當會員進行登入完成後，即可進行過去上架商品、下標資料及得標資訊的查詢，當會員登入完成後，右上方標籤列，會轉變為會員資料查詢與訂單查詢。選擇訂單查詢即可跳轉至訂單頁面，如下圖所示。

進入訂單明細後，會先顯示過去會員已上架之商品，並可以進行選擇，已上架、競標中及已得標，三種商品列表查看。如下圖所示。



相關圖片列表，利用 Datalist 進行資料庫的讀取，並將資料庫資訊提供給前端語法顯示於網站表格當中。利用前端 table 標籤，將需要顯示的樣式進行調整後，使用 imagebutton 及 label 顯示出讀取資料。以方便後續點擊圖片可以連結到後續該商品列表。並使用分頁方法，計算共有多少商品顯示於表格上方，並將過多商品資訊於下一頁中呈現。

● 商品各列表資料庫的讀取與送入前端 Datalist 列表中，程式碼如下圖所示。

```
public void ListBind(string connString, string sql_statement1_onsale)
{
    //設定匯入的起終地址
    int StartIndex = CurrentPage * PageSize;
    cmd_page.Connection = conn;
    string s_data = System.Web.Configuration.WebConfigurationManager.ConnectionStrings[connString].ConnectionString;
    conn.ConnectionString = s_data;

    cmd_page.CommandText = sql_statement1_onsale;   //執行SQL語法進行查詢
    da.SelectCommand = cmd_page;            //da選擇資料來源，由cmd載入進來
    da.Fill(ds_page, StartIndex, PageSize, "Action_product");//da把資料填入ds裡面

    product_view.DataSource = ds_page.Tables["Action_product"].DefaultView;
    product_view.DataBind();
```

● 前端標籤 table 與 Datalist

使用 table 標籤排版進行將商品頁顯示於各列表中，並使用 datalist 夾帶 imagebutton 以及 Eval 方法將資料庫內容填入對應標籤當中。

```
<asp:DataList ID="product_view" runat="server" Width="100%" OnItemCommand="product_view_ItemCommand" OnItemDataBound="product_view_DataBound" >
        <ItemTemplate>
            <div class="thumbnail">
                <table  border="1" >
                    <tr>
                        <td align='center'>商品圖片</td>
                        <td align='center'>商品名稱</td>
                        <td align='center'>商品市價</td>
                        <td align='center'>商品描述</td>
                    </tr>
                    <tr>
                        <td >
                        <asp:ImageButton ID="pic_pathname" width="200" runat="server"  CommandName="click"  ImageUrl='<%# Eval("pic_pathname") %>' />
                        </td>

                        <td align='center'>
                        <asp:Label ID="product" width="80" runat="server" Text='<%# Eval("product") %>'/>
                        </td>

                        <td align='center'>
                        <asp:Label ID="official_price" width="80" runat="server" Text='<%# Eval("official_price") %>'/>
                        </td>

                        <td align='center'>
                        <asp:Label ID="description" width="290" runat="server" Text='<%# Eval("description") %>'/>
                        <asp:Label ID="action_product_ID" runat="server" Visible="false" Text='<%# Eval("action_product_ID") %>'/>
                        <asp:Label ID="seller_ID" runat="server" Visible="false" Text='<%# Eval("seller_ID") %>'/>
                        </td>
                    </tr>
                </table>
            </div>
        </ItemTemplate>
    </asp:DataList>
```

- 點擊商品圖案連接各商品頁

使用 datalist 中 ItemCommand 的方法，判定若 CommandName 成為點擊的行為的時候，將會將所有此商品資訊，利用 Session 存取起來後，送往各商品顯示列表，方便後續商品資料讀取。程式碼如下圖所示。

```
protected void product_view_ItemCommand(object source, DataListCommandEventArgs e)
{
    DataListItem currentItem = e.Item;

    if (e.CommandName == "click")
    {

        product_view.SelectedIndex = currentItem.ItemIndex;

        string pic_pathname = ((ImageButton)currentItem.FindControl("pic_pathname")).ImageUrl;
        string product = ((Label)currentItem.FindControl("product")).Text;
        string official_price = ((Label)currentItem.FindControl("official_price")).Text;
        string action_product_ID = ((Label)currentItem.FindControl("action_product_ID")).Text;
        string description = ((Label)currentItem.FindControl("description")).Text;
        string seller_ID = ((Label)currentItem.FindControl("seller_ID")).Text;

        Session["ProductName"] = product;
        Session["ProductDesc"] = description;
        Session["official_price"] = official_price;
        Session["ImageUrl"] = pic_pathname;
        Session["ProductID"] = action_product_ID;
        Session["SellerID"] = seller_ID;

        Response.Redirect("BitProductForm.aspx");

    }
}
```

- Datalist 分頁方法

- 因為 Datalist 與 gridview 不同，並沒有內建之分頁設定辦法，因此使用計算每頁要顯示的資訊量，進行分別輸入到 Datalist 中，藉已呈現分頁的效果。首先計算共有多少商品顯示於表格上方，並將過多商品資訊於下一頁中呈現，並利用 linkbuttion 中的 ItemCommand 狀態來進行改變 Datalist 的商品呈現。網站圖示與程式碼如下列圖所示。

- 網站示意圖 ：
  左上可以顯示，此商品狀態的商品共有多少份，並且顯示這是瀏覽中的第幾頁，以及下方可以使用 linkbutton 進行切換不同商品頁面。

- 資料庫讀取，先行設定資料庫擷取等等相關資訊，以及之後會使用到的分頁項目變數。

```
public partial class memberOrder : System.Web.UI.Page
{
    SqlDataAdapter da = new SqlDataAdapter();        //SQL 資料庫的連接與執行命令
    DataSet ds_check = new DataSet();
    DataSet ds_page = new DataSet();
    SqlCommand cmd_check = new SqlCommand();
    SqlCommand cmd_page = new SqlCommand();
    SqlConnection conn = new SqlConnection();

    //設定分頁項目
    int PageSize, RecordCount, PageCount, CurrentPage;

    //設定資料庫資訊
    string connString = "Sale_net_Jun22_2021ConnectionString";
```

● 整體分頁計算流程:

將商品資訊進行匯入到前端 Datalist 列表當中，並且計算目前 dataset 當中，於總頁數中的第幾頁以及下一頁等等相關資訊，進行分頁流程後，將本頁需要顯示的商品份數匯入到 Datalist 中。

```
//初始設定剛進頁面 匯入資料
ListBind(connString, cmdText_onsale);
CurrentPage = 0;
ViewState["PageIndex"] = 0;

//計算總共有多少記錄(onsale)
string sql_statement1_onsale = $"select count(*) as co from Action_product " +
"where status='onsale' and seller_ID='" + Session["member_ID"] + "'";
RecordCount = CalculateRecord(connString, sql_statement1_onsale);
lblRecordCount.Text = RecordCount.ToString();

//計算總共有多少頁(onsale)
PageCount = (RecordCount / PageSize)+1;
lblPageCount.Text = PageCount.ToString();
ViewState["PageCount"] = PageCount;
```

● 將資料庫中讀取到的資料，先行依照每頁需要顯示多少資訊，例如:每頁十筆，先取出前十筆資料，匯入到前端 Datalist 中，並且計算目前頁面，以及下一頁等等相關分頁變數的設定值。

```
// 匯入資料
7 個參考
public void ListBind(string connString, string sql_statement1_onsale)
{
    //設定匯入的起終地址
    int StartIndex = CurrentPage * PageSize;
    cmd_page.Connection = conn;
    string s_data = System.Web.Configuration.WebConfigurationManager.ConnectionStrings[connString].ConnectionString;
    conn.ConnectionString = s_data;

    cmd_page.CommandText = sql_statement1_onsale;   //執行SQL語法進行查詢
    da.SelectCommand = cmd_page;            //da選擇資料來源，由cmd載入進來
    da.Fill(ds_page, StartIndex, PageSize, "Action_product");//da把資料填入ds裡面

    product_view.DataSource = ds_page.Tables["Action_product"].DefaultView;
    product_view.DataBind();

    //設定頁數與按鈕顯示
    lbnNextPage.Enabled = true;
    lbnPrevPage.Enabled = true;
    if (CurrentPage == (PageCount - 1)) lbnNextPage.Enabled = false;
    if (CurrentPage == 0) lbnPrevPage.Enabled = false;
    lblCurrentPage.Text = (CurrentPage + 1).ToString();
```

● 計算目前 dataset 當中，於總頁數中的第幾頁以及下一頁等等相關資訊，進行分頁流程後，將下一頁需要的各式分頁變數的值確定，方便按下下一頁後進行選取資料的匯入。

```
//計算總共有多少條記錄
4 個參考
public int CalculateRecord(string connectiion, string sql_statement1)
{
    string s_data = System.Web.Configuration.WebConfigurationManager.ConnectionStrings[connectiion].ConnectionString;
    SqlConnection connection = new SqlConnection(s_data);
    SqlCommand Command = new SqlCommand(sql_statement1, connection);
    connection.Open();
    SqlDataReader Reader = Command.ExecuteReader();

    int intCount;

    if (Reader.HasRows)
    {
        Reader.Read();
        intCount = Int32.Parse(Reader["co"].ToString());
    }
    else
    {
        intCount = 0;
    }
    Reader.Close();
    return intCount;
```

● 當按下下一頁或是上一頁:

使用 linkbutton 中 CommandName 行為辨認，判定為前後頁選擇，並且將當前頁面所有分頁相關資訊進行讀取與設定，最後將呼叫一開始的 listBind()方法，將新頁面資訊匯入 Datalist 顯示，以此進循環，至頁面結束。

```
//按下更換分頁
0 個參考
public void Page_OnClick(Object sender, CommandEventArgs e)
{
    CurrentPage = (int)ViewState["PageIndex"];
    PageCount = (int)ViewState["PageCount"];

    string cmd = e.CommandName;
    //判斷cmd,以判定翻頁方向
    switch (cmd)
    {
        case "next":
            if (CurrentPage < (PageCount - 1)) CurrentPage++;
            break;
        case "prev":
            if (CurrentPage > 0) CurrentPage--;
            break;
    }

    ViewState["PageIndex"] = CurrentPage;
```

各種選擇商品頁狀態切換，使用 DropDownList1_SelectedIndexChanged 方法，將不同狀態的下拉式標籤，於 SQL 語法中，選擇資料庫中需要顯示或是承載資料，並利用商品狀態的不同，以及 Session 紀錄登入後的會員資訊進行篩選，並顯示於訂單列表中。

- 選擇已上架，程式碼如下圖所示。

```
protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
{

    if (DropDownList1.SelectedValue == "已上架")
    {
        product_view.Visible = true;

        //已上架
        string cmdText_onsale = "SELECT pic_pathname,product,description,total_number,seller_ID,action_product_ID,official_price " +
                "from Action_product where status='onsale' and seller_ID='" + Session["member_ID"] + "'";
        // 判斷有沒有資料
        SQL_readActionProduct(connString, cmdText_onsale);
```

- 選擇競標中，程式碼如下圖所示。

```
else if (DropDownList1.SelectedValue == "競標中")
{
    product_view.Visible = true;

    //競標中
    string cmdText_bidding = "SELECT pic_pathname,product,description,total_number,seller_ID,action_product_ID,official_price " +
        "from Action_product where status='bidding' and seller_ID='" + Session["member_ID"] + "'";
    // 判斷有沒有資料
    SQL_readActionProduct(connString, cmdText_bidding);
```

- 選擇已得標，程式碼如下圖所示。

```
else if (DropDownList1.SelectedValue == "已得標")
{
    product_view.Visible = true;

    //已結標
    string cmdText_getbid = "SELECT pic_pathname,product,description,total_number,seller_ID,action_product_ID,official_price " +
        "from Action_product where status='getbid' and seller_ID='" + Session["member_ID"] + "'";
    // 判斷有沒有資料
    SQL_readActionProduct(connString, cmdText_getbid);
```

C. 會員商品上架

商品上架提供使用者將商品資訊放置網站空間等功能。相關實作頁面有：商品上架畫面(PutGoodInShelfForm.aspx)。該頁面可設定商品名稱、商品分類、商品描述、商品描述、官方售價、商品數量、結標日期與時間、商品圖片等功能。

▲商品上架可分類群組

程式部分：PutGoodOnShelfForm.aspx.cs

在網頁第一次顯示時，先設定 商品分類、結標小時與分鐘時間等 DropDownList 等內容。並將 結標小時與分鐘 設為 目前系統時間。

```
for (int index = 0; index < aClassfyTitle.Length; index++)
{
    string item = aClassfyTitle[index];
    _Classfy.Items.Add(item);
}

// 2. preset _ClosedHr_list
for (int index = 0; index < 24; index++)
    _ClosedHr_list.Items.Add(index.ToString());

// 3. preset _ClosedMMin_list
for (int index = 0; index < 60; index++)
    _ClosedMin_list.Items.Add(index.ToString());

//給下拉框預設顯示當前小時、分鐘
_ClosedHr_list.Text = (DateTime.Now.Hour).ToString();
_ClosedMin_list.Text = (DateTime.Now.Minute).ToString();
```

若 Session 曾經設定過 Preset_PutGoodOnShelf，則表示使用者ㄘ，則表示使用者曾經頁面中留下資料，現在需要將前面輸入過的資料填回到網頁內容裡。

```
if (Session["Preset_PutGoodOnShelf"] != null)
{
    presetGUI();
}
```

```
private void presetGUI()
{
    string strYear =(string) Session["Preset_CalendarClosedDate_Year"];
    int nYear = (int.Parse(strYear));

    string strMonth =(string) Session["Preset_CalendarClosedDate_Month"];
    int nMonth = (int.Parse(strMonth));

    string strDay =(string) Session["Preset_CalendarClosedDate_Day"];
    int nDay = (int.Parse(strDay));

    DateTime closedDateTime = new DateTime(nYear, nMonth, nDay);
```

```
string strHour =(string) Session["Preset_ClosedHr_list"];
int nHour = (int.Parse(strHour));

string strMin =(string) Session["Preset_ClosedMin_list"];
int nMin = (int.Parse(strMin));
TimeSpan timespan = new TimeSpan(nHour, nMin, 0);
closedDateTime += timespan;

// keep session info of closed dateTime to class member
 _closedDateTime = closedDateTime;
```

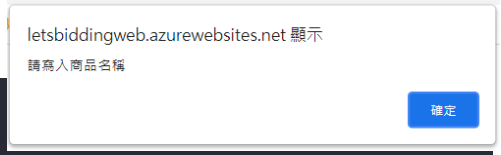
```
_ClosedHr_list.Text = (string) Session["Preset_ClosedHr_list"];
_ClosedMin_list.Text = (string)Session["Preset_ClosedMin_list"];

_Classfy.Text       = (string) Session["Preset_Classfy"];
_ProductName.Text   = (string) Session["Preset_ProductName"];
_GoodDesc.Text      = (string) Session["Preset_GoodDesc"] ;
_OfficialPrice.Text = (string) Session["Preset_OfficialPrice"];
_TotalLots.Text     = (string) Session["Preset_TotalLots"];


_ClosedHr_list.Text  = (string)Session["Preset_ClosedHr_list"];
_ClosedMin_list.Text = (string) Session["Preset_ClosedMin_list"];
_ImgGood.ImageUrl = (string) Session["Preset_ImgGood_Pathname"];
```

當使用者按下 商品上架 按鈕，程式先檢查 會員號碼是否已存在；若不存在，表示目前使用者未登入系統，則將目前已輸入的資料存在 Session 中，將本頁面寫入 Session，並將頁面轉向會員登入畫面。

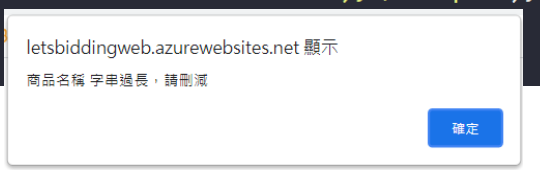
```
protected void _putOnShelfBtn_Click(object sender, EventArgs e)
{
    int nOffPrice = 0, nTotalLots = 0;

    if (Session["member_ID"] == null)
    {

        setPresetGUI();
        // to set pre logged page
        Session["logged_to_page"] = "PutGoodOnShelfForm.aspx";

        // to the member logging page
        Response.Redirect("memberLoginForm.aspx");
    }
```

```
try
{
    nOffPrice = int.Parse(_OfficialPrice.Text);
}
catch (System.FormatException)
{
    Response.Write($"<script>alert('官方售價 請填入 數字');</script>");
    return;
}

try
{
    nTotalLots = int.Parse(_TotalLots.Text);
}
catch (System.FormatException)
{
    Response.Write($"<script>alert('商品數量 請填入 數字');");
    return;
}
```

● 若會員已登入系統，則陸續檢查以下資訊：官方售價、商品數量

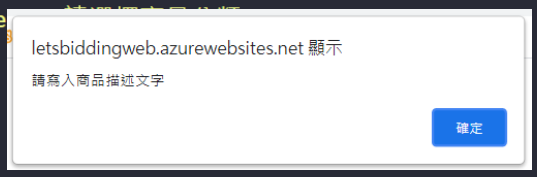


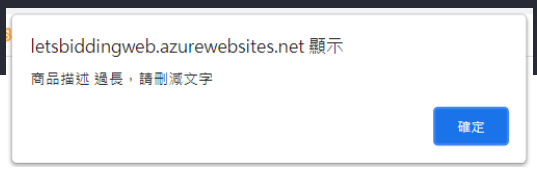

```
if (_ProductName.Text == "")
{
    Response.Write($"<script>alert('請寫入商品名稱');</script>");
    return;
}
else if(IfStringsOverLength(_ProductName.Text, 50) == true)
{
    Response.Write($"<script>alert('商品名稱 字串過長，請刪減');</script>");
    return;
}
```



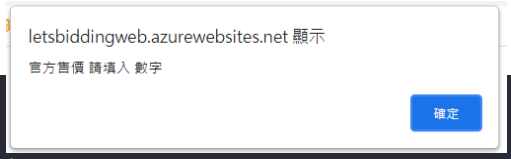
● 檢查商品名稱及名稱長度、商品描述文字

```
else if (_Classfy.Text == "")
{
    Response.Write($"<script>ale
    return;
}
else if (_GoodDesc.Text == "")
{
    Response.Write($"<script>alert('請寫入商品描述文字');</script>");
    return;
}
else if (IfStringsOverLength(_GoodDesc.Text, 500) == true)
{
    Response.Write($"<script>alert('商品描述 過長，請刪減文字');</script>");
    return;
}
```

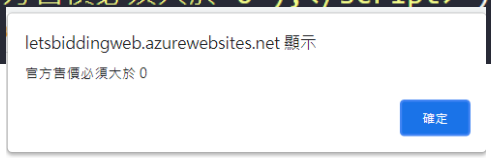




● 檢查商品描述文字及文字長度

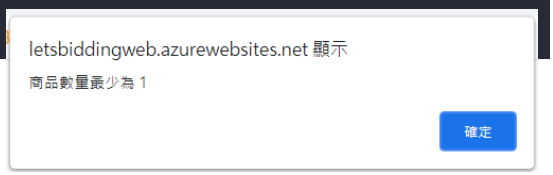



```
else if (_OfficialPrice.Text == "")
{
    Response.Write($"<script>alert('請填官方售價');</script>");
}
else if (int.Parse(_OfficialPrice.Text) <= 0)
{
    Response.Write($"<script>alert('官方售價必須大於 0');");
}
```



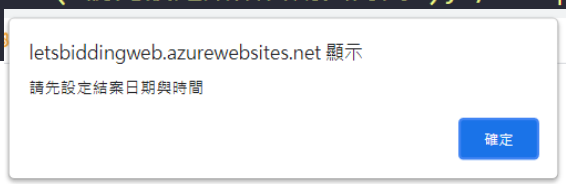
● 檢查商品官方售價，且不能小於或等於零

```
else if (_TotalLots.Text == "")
{
    Response.Write($"<script>alert('請填商品數量');</script>");
}
else if (int.Parse(_TotalLots.Text) <= 0)
{
    Response.Write($"<script>alert('商品數量最少為 1');</script>");
    return;
}
```



● 檢查商品數量，且不能小於或等於零

```
else if (_CalendarClosedDate.SelectedDate == DateTime.MinValue.Date)
{
    Response.Write($"<script>alert('請先設定結案日期與時間');</script>");
}
```



● 檢查商品結標日期：如果日期與今天相同，代表未設定

```
else if (_ImgGood.ImageUrl == "")
{
    Response.Write($"<script>alert('請上傳商品圖片');</script>");
    return;
}
```

● 檢查是否已上傳商品圖片

```
// all input items are ready
DateTime closedDateTime = _CalendarClosedDate.SelectedDate;
TimeSpan setTimespan = new TimeSpan(int.Parse(_ClosedHr_list.Text), int.Parse(_ClosedMin_list.Text), 0);
closedDateTime += setTimespan;
/**/

SQLDB_writeAutionProduct("Sale_net_Jun22_2021ConnectionString2", closedDateTime);
Response.Redirect("GoodListForm.aspx");
```

● 最後將結標時間加上小時、分鐘資訊，寫入商品資料庫。將頁面轉向商品列表頁面。

三、商品資訊

於我們的網頁中，可以在三個地方看到商品呈現，分別為：首頁熱門商品(Home.aspx)、商品分類頁(GoodListForm.aspx)、商品列表(GoodListForm.aspx)，如下圖中(1)、(2)、(3)。其中首頁為獨立的程式檔，商品分類頁及商品列表為相同的程式檔，我們透過紀錄 Session 的方式從資料庫中取出不同的商品資料，詳細程式說明於(4)拍賣系統中說明。

A. 首頁熱門商品

如下圖，我們從資料庫中取出最熱門的前三名商品，展示於首頁熱門商品中。

	action_product_ID	classify	product	total_number	low_price	high_price	status	description
1	1	life	iPhoneX	1	500	0	getbid	Apple原廠正品福利機,機況漂亮無傷,跟全新品
2	7	life	iPhoneXS	12	0	0	onsale	採用 iOS 12 作業系統6.5 吋 2,688 x 1,242pixels 解
3	8	life	Apple iPhone X 256G	37	0	0	onsale	Apple iPhone X 256GB 功能特色 ◎ 採用 iOS 11

資料庫搜尋語法如下圖紅框部分。

```
cmd.Connection = conn;    //將SQL執行的命令語法程式CMD與CONN與SQL連接

//設定連線IP位置、資料表、帳戶、密碼
string s_data = System.Web.Configuration.WebConfigurationManager.ConnectionStrings[connString].Connection
conn.ConnectionString = s_data; //"Data Source=127.0.0.1;Initial Catalog=NorthwindChinese;Persist Securit
//這一行可依連線的字串不同而去定義它該連線到哪個資料庫!!

cmd.CommandText = $"SELECT Top 3 pic_pathname,product,official_price,status,description,seller_ID," +
    $"action_product_ID from Action_product where classify = 'life' and status = 'onsale'";    //執行SQL語
da.SelectCommand = cmd;              //da選擇資料來源，由cmd載入進來
da.Fill(ds_first, "Action_product"); //da把資料填入ds裡面
```

B. 商品列表

商品列表中呈現資料庫中所有商品，每個商品皆會顯示圖片、商品名稱、商品描述等三種資訊，透過商品上架可新增商品於列表中，系統管理員可編輯目前架中之商品，

點擊商品左側 Select 可進入該商品商品頁面，如下圖所示。

C. 商品類別

商品的分類共分為九類，透過點擊左側的類別按鈕後會透過 Session 記錄選取類別資訊，透過已記錄之 Session 可進入該類別的商品列表，如下圖所示。

點擊後透過 Session 紀錄類別，並跳轉至商品列表。

```
0 個參考
protected void cloth_Click(object sender, EventArgs e)
{
    Session["classify"] = "cloth";
    Response.Redirect("GoodListForm.aspx");
}
```

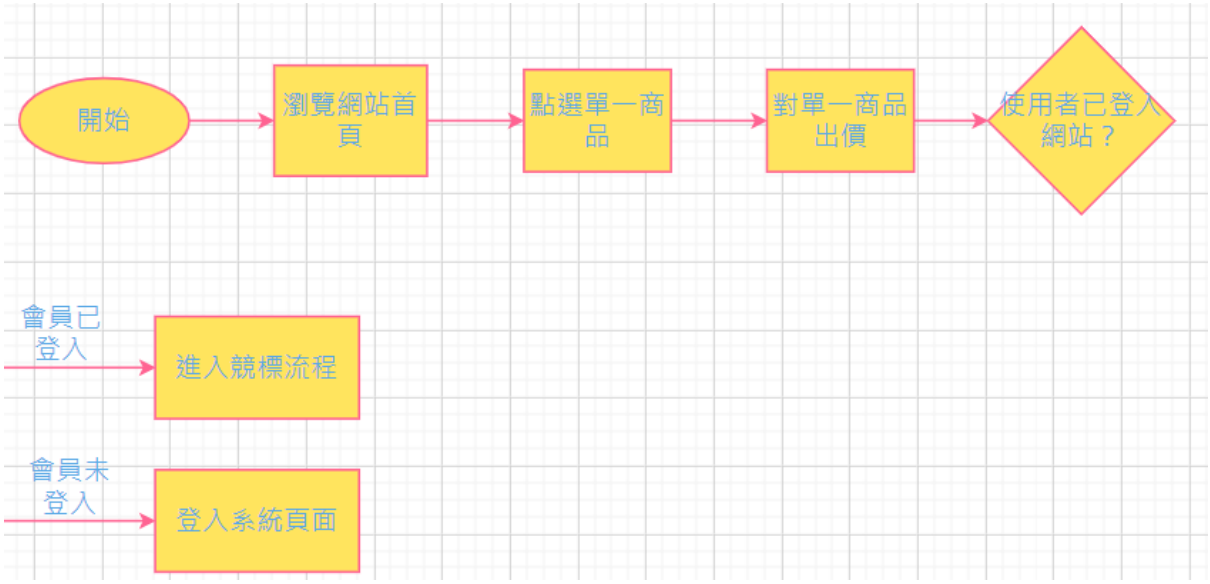
接著再透過資料庫查詢語法，查詢該類別商品，如下圖程式碼所示。

```
// bug1: SQL content with session classify
string sql_statement1_classify = $"SELECT pic_pathname,product,description,total_number,seller_ID," +
    $"Action_product_ID from Action_product where status='onsale' and classify ='" + Session["classify"] + "'";
```

四、拍賣系統

拍賣為本系統最主要的功能頁，提供使用者瀏覽商品資料、參與競標等功能。相關實作頁面有：顯示商品及下標畫面(BitProductForm.aspx)。該頁面顯示使用者選取的單一商品，並顯示商品圖片、商品名稱、商品描述、官方售價。本競標功能頁主要的兩大功能(選取商品及下標)的流程圖如下圖所示：

A. 選取商品功能



程式部分：Home.aspx.cs

先將包包顯示畫面與資料庫內容繫結，並依照使用者登入狀態改變顯示功能。

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        product_view_bag.DataBind();    // 自訂的 GridView databind函式

        //設定會員登入與否顯現標示不同
        if (Convert.ToString(Session["Login"]) == "logged")
        {
            member_info.Visible = true;
            order_info.Visible = true;
            logout.Visible = true;
        }
        else
        {
            my_info.Visible = true;
            register.Visible = true;
            manager.Visible = true;
        }
```

依照商品種類分別讀取資料庫的資料，分別放入不同的網頁顯示空間。

```
//fetchProductInfo("Sale_net_Jun18_2021_betaConnectionString3");
SQL_readActionProduct_life("Sale_net_Jun22_2021ConnectionString2");
product_view_life.DataSource = ds_first; //將DataSet的資料載入到datalist內
product_view_life.DataBind();

SQL_readActionProduct_cloth("Sale_net_Jun22_2021ConnectionString2");
product_view_cloth.DataSource = ds_sec; //將DataSet的資料載入到datalist內
product_view_cloth.DataBind();

SQL_readActionProduct_bag("Sale_net_Jun22_2021ConnectionString2");
product_view_bag.DataSource = ds_third; //將DataSet的資料載入到datalist內
product_view_bag.DataBind();
```

當使用者選取單一商品，系統將商品路徑名稱、商品名稱、商品代號、商品描述、賣家代號等資料存入 Session，並將畫面轉向單一商品頁面。

```
protected void product_view_life_ItemCommand(object source,DataListCommandEventArgs e)
{
    DataListItem currentItem = e.Item;
    if (e.CommandName == "click")
    {
        product_view_life.SelectedIndex = currentItem.ItemIndex;

        string pic_pathname = ((ImageButton)currentItem.FindControl("pic_pathname")).ImageUrl;
        string product = ((Label)currentItem.FindControl("product")).Text;
        string action_product_ID = ((Label)currentItem.FindControl("action_product_ID")).Text;
        string description = ((Label)currentItem.FindControl("description")).Text;
        string seller_ID = ((Label)currentItem.FindControl("seller_ID")).Text;

        Session["ProductName"] = product;
        Session["ProductDesc"] = description;
        Session["ImageUrl"] = pic_pathname;
        Session["ProductID"] = action_product_ID;
        Session["SellerID"] = seller_ID;

        Response.Redirect("BitProductForm.aspx");
    }
```

程式部分：BitProductFrom.aspx.cs

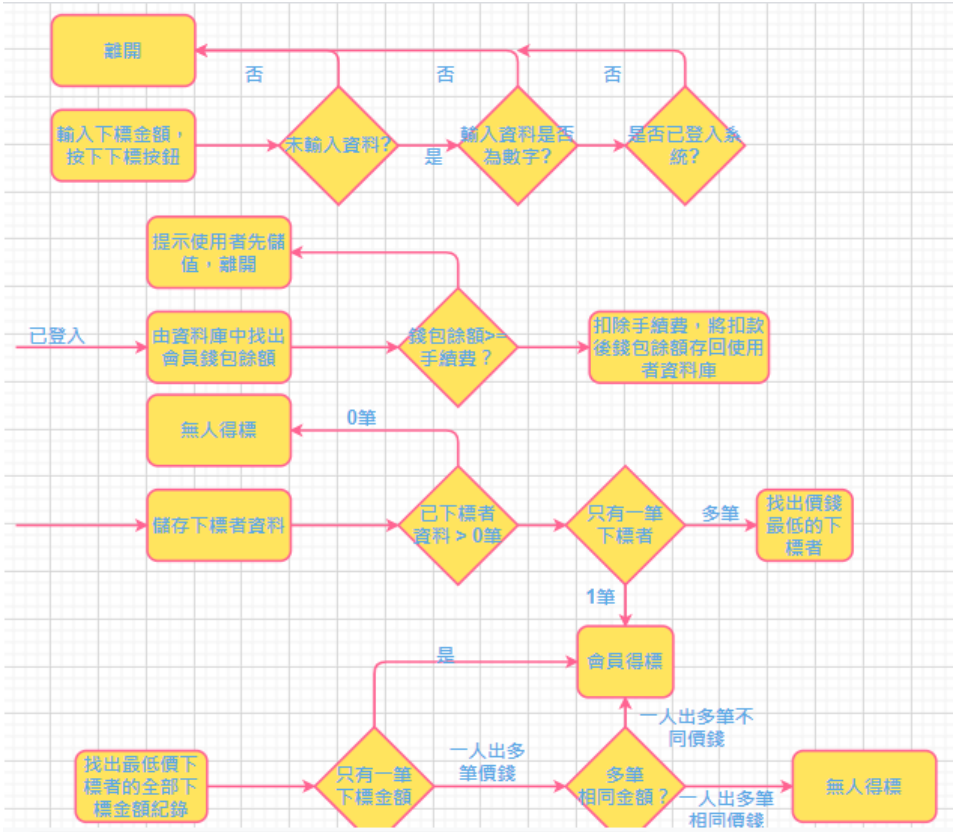
在單一商品頁面(BitProductFrom.aspx.cs)開啟網頁時，首先判斷 Session 是否已存入資料。若已存入，則讀取資料，並在頁面上顯示商品名稱、商品描述。並利用商品路徑名稱將商品圖片顯示在頁面中。

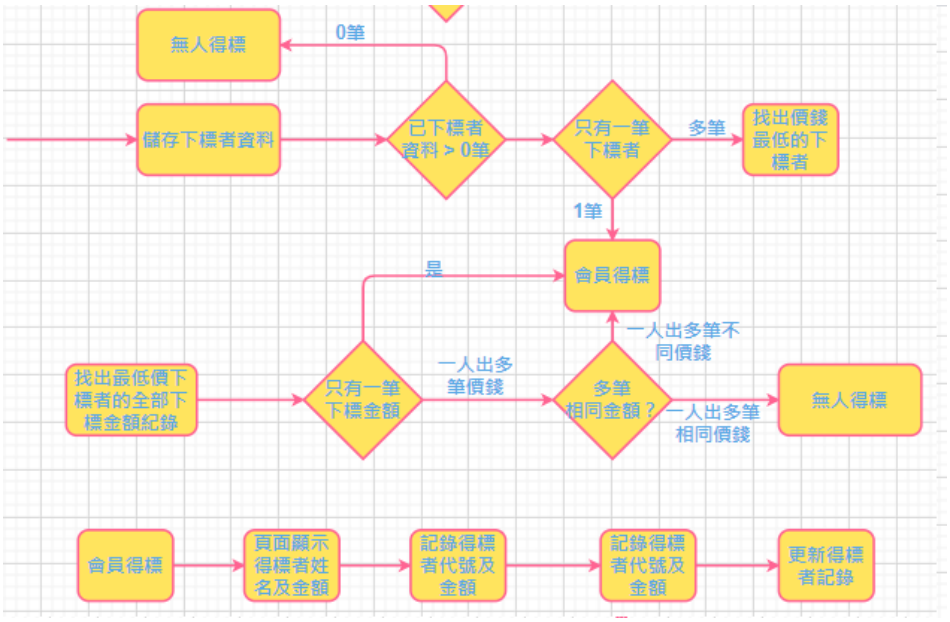
```
if (Session["ProductDesc"] != null)
{
    string strProductDesc = (string)Session["ProductDesc"];
    _ProductDesc.Text = strProductDesc;
}

if (Session["SellerID"] != null)
{
    string strSellerID = (string)Session["SellerID"];
    _sellerID = int.Parse(strSellerID);
}

if (Session["ImageUrl"] != null)
{
    string strImageUrl = (string)Session["ImageUrl"];
    _ProductImage.ImageUrl = strImageUrl;
}
```

取得商品代號後，將該商品的結標時間從資料庫中讀出，顯示在網頁上。也顯示資料庫中目前得標者的姓名及出價。

```
if (Session["ProductID"] != null)
{
    string strProductID = (string)Session["ProductID"];
    _ProductID = int.Parse(strProductID);

    getSQLDB_FindProduct_closedDateTime_officialPrice("Sale_net_Jun22_2021ConnectionString2", _ProductID);

    // to display least time
    showProductLastTime();

    // to display the bit winner until now
    ShowNowBitWinner(_ProductID);
}
```

B. 下標功能





程式部分：BitProductForm.aspx.cs

首先判斷輸入金額，如果未輸入，則提醒使用者，並留在原畫面；如果輸入的不是數字資料，則提醒使用者，並留在原畫面。

```
protected void _BitBtn_Click(object sender, EventArgs e)
{
    int nPrice = 0;
    if (_BitPrice.Text == "")
    {
        Response.Write("<script>alert('請先輸入出價金額！');</script>");
        return;
    }
    try
    {
        nPrice = int.Parse(_BitPrice.Te
    }
    catch (System.FormatException)
    {
        Response.Write($"<script>alert('出價金額 請填入 數字');</script>");
        return;
    }
```

localhost:44356 顯示
請先輸入出價金額！
確定

localhost:44356 顯示
出價金額 請填入 數字
確定

檢查使用者是否已登入系統：如未登入，則將目前頁面資訊存放 Session，以便登入系統後再回到目前頁面；提醒使用者先登入會員，並切換到會員登入頁面。

```
if(Session["member_ID"] == null){
    Session["logged_to_page"] = "BitProductForm.aspx";
    // to the member logging page
    Response.Write($"<script>alert('請先登入會員');</script>");
    Response.Redirect("memberLoginForm.aspx");
    //Server.Transfer("memberLoginForm.aspx");
    return;
}
```

檢查會員錢包餘額；如果錢包餘額比手續費還低，則提醒使用者儲值；如錢包餘額比手續費還高，則將錢包餘額扣除手續費，再存入資料庫中會員錢包餘額。

```
_bidder_ID = (int)Session["member_ID"];
int bidderBalance = getSQLDB_FindMember_balance("Sale_net_Jun22_2021ConnectionString2", _bidder_ID);

// 1. if balance is enough
if (bidderBalance < nCharge)
{
    Response.Write("<script>alert('會員手續費不足，請先儲值！');</script>");
    // ShowNowBitWinner();
    return;
}// balance is not enough
else
{
    // 2. balance minues charge
    // 2.1 balance minus charge
    bidderBalance -= nCharge;
    // 2.2 store balance to DB
    SQLDB_saveMemberBalance("Sale_net_                                                    dderBalance);
}// balance is enough
```

localhost:44356 顯示

會員手續費不足，請先儲值！

確定

	status	datetime	balance
	active	2021-05-23 18:30:01.793	1000
5	正常	2021-05-23 21:24:30.597	80

Member資料庫已扣款

結果 訊息

	action_bidder_ID	number	price	datetime	status	action_product_ID	bidder_ID
1	2	0	3000	2021-06-13 23:48:30.607	競標中	8	2
2	3	0	3000	2021-06-13 23:54:01.253	競標中	8	2
3	4	0	3000	2021-06-13 23:58:05.513	競標中	8	2

儲存下標者資料。

```
string strProductID = (string) Session["ProductID"];
int nProductID = int.Parse(strProductID);
SQLDB_saveBidder("Sale_ne                                                    ", _bidder_ID, nProductID, nPrice);
```

官方售價： 31500

目前得標者： 無

出價

結果 訊息

	action_bidder_ID	number	pri			product_ID	bidder_ID
1	3	0	30				2
2	4	0	3000	2021-06-13 23:58:05.513	競標中	8	2
3	5	0	3000	2021-06-15 16:52:36.643	競標中	8	2
4	6	0	3000	2021-06-15 17:01:20.923	競標中	8	2
5	7	0	4000	2021-06-15 17:46:46.513	競標中	8	10

取得所有下標者資料：若沒有人下標，則得標者名字和得標金額為無。

```
// 1. to open Action_bidder
SQLDB_getBidderPrice("Sale_net_Jun22_2021ConnectionString2", ref bidderData, nSelectedProdcutID);
// case1 : nobody bids
if(bidderData.Count == 0){
    _NowBitWinner.Text = "無";
    _NowBitPrice.Text =  "無";
}
```

找出所有下標記錄中最低價的下標者

```
// to get the smallest price
for (int i = 0; i < bidderData.Count; i++)
{
    if(i == 0)
        bidderDataWinner = bidderData[i];
    else{
        if(bidderDataWinner._nBidPrice > bidderData[i]._nBidPrice)
            bidderDataWinner = bidderData[i];
    }// i=1..
}// for
```

找出最低價出價者的所有出價記錄

```
// 3. to get all bid records belongs the bidder
List<BidderData>  listSameBidderPrices = bidderData.FindAll(x => x._nBidderID == bidderDataW
```

如果出價記錄只有一筆，代表是最低價且唯一標，這就是得標者。網頁顯示得標者的姓名及出價金額。儲存得標者資料，並更新得標者紀錄。

```
if(listSameBidderPrices.Count == 1){
    // he/she is winner
    nWinnerID = bidderDataWinner._nBidderID;
    // to get Member name from Member DB
    SQLDB_getBidderNameFromMember("Sale_net_Jun22_2021ConnectionString2", nWinnerID, ref str
    _NowBitWinner.Text = strWinnerName;

    string strPrice = bidderDataWinner._nBidPrice.ToString();
    _NowBitPrice.Text = strPrice;

    // save winner to Action_product
    int nProdcutID = getProductIDfromSession();
    SQLDB_saveProductBitWinner("Sale_net_Jun22_2021ConnectionString2", nWinnerID, nProdcutID

    // save winner to Action_bidder
    SQLDB_updateBidderWinner(bidderDataWinner._nBidderID, bidderDataWinner._nBidPrice, nProd
```

Jack	789456	Jack	jack@gmail.com	0935333444	1989	8	12
Tom	456123	Tom	tom@gmail.com	0933111222	1992	6	1

若一個人出了很多筆競標單，多筆標單又是同一價錢，那就無人得標。網頁顯示得標者：無。

```
// 1 person multiple bids
//  if each of those prices are the same: no winner
if(ifAllPriceAreSame(listSameBidderPrices) == true){
    _NowBitWinner.Text = "無";
    _NowBitPrice.Text =  "無";
}
```

結果 訊息

	action_bidder_ID	number	price	datetime	status	action_product_ID	bidder_ID
1	3	0	3000	2021-06-13 23:54:01.253	競標中	8	2
2	4	0	3000	2021-06-13 23:58:05.513	競標中	8	2
3	5	0	3000	2021-06-15 16:52:36.643	競標中	8	2
4	6	0	3000	2021-06-15 17:01:20.923	競標中	8	2

若最低價只有一人，他/她卻出了很多價錢的標單，也算是得標者。網頁顯示得標者的姓名及出價金額。儲存得標者資料，並更新得標者紀錄。

```
else{
    nWinnerID = bidderDataWinner._nBidderID;
    // to get Member name from Member DB
    SQLDB_getBidderNameFromMember("Sale_net_Jun22_2021ConnectionString2", nWinnerID,
    _NowBitWinner.Text = strWinnerName;

    string strPrice = bidderDataWinner._nBidPrice.ToString();
    _NowBitPrice.Text = strPrice;

    int nProdcutID = getProductIDfromSession();
    // save winner to Action_product
    SQLDB_saveProductBitWinner("Sale_net_Jun22_2021ConnectionString2", nWinnerID, nP

    // save winner to Action_bidder
    SQLDB_updateBidderWinner(bidderDataWinner._nBidderID, bidderDataWinner._nBidPric
}// 1 person diff bids, he/she is winner
```

```
11  FROM [Sale_net].[dbo].[Action_bidder]
```

20 %

結果 訊息

	action_bidder_ID	number	price	datetime	status	action_product_ID	bidder_ID	low_price	bit_winner
1	3	0	3000	2021-06-13 23:54:01.253	競標中	8	2	NULL	NULL
2	4	0	3000	2021-06-13 23:58:05.513	競標中	8	2	NULL	NULL
3	5	0	3000	2021-06-15 16:52:36.643	競標中	8	2	NULL	NULL
4	6	0	3000	2021-06-15 17:01:20.923	競標中	8	2	NULL	NULL
5	7	0	4000	2021-06-15 17:46:46.513	競標中	8	10	NULL	NULL
6	8	0	2800	2021-06-15 21:08:14.990	競標中	8	13	2800	Yes

五、管理系統(待完成)

6. 實作困難點

a. 會員修改-電話防呆

b. 會員修改-dropdownlist 可以打字

c. 拍賣得標者無法被後來同價錢的下標者修改

d. 首頁跑馬燈商品可以點擊瀏覽商品下標

e. 雲端 azure 資料庫寫入時中文換變成??亂碼:

f. 將寫入雲端之 Sqldatatype 須改為符合資料庫設計之資料類型(由 TEXT 改為 NVarChar)，否則寫入資料庫時就會變成亂碼，而無法正確顯示，非網頁 meta 標籤中 utf8 問題。如下圖所示。

資料行名稱	資料類型	允許 Null
action_product_ID	int	☐
classify	nvarchar(50)	☑
product	nvarchar(50)	☑
total_number	nchar(10)	☑
low_price	nchar(10)	☑
high_price	nchar(10)	☑
status	nchar(10)	☑
description	nvarchar(500)	☑
official_price	int	☑
seller_ID	int	☑
datetime	datetime	☑
pic_pathname	nvarchar(255)	☑
closedDateTime	datetime	☑
bid_winner_ID	int	☑
bid_price	int	☑
bidder_ID	int	☑
receiver_name	nvarchar(50)	☑
receiver_phone	nvarchar(50)	☑
receiver_address	nvarchar(50)	☑
bit_product_ID	int	☑
		☐

```
else if (_Classfy.Text == "其他類別")
{
    classify_temp = "other";
}

sql_insert_cmd.Parameters.Add("@classify", SqlDbType.NVarChar);
sql_insert_cmd.Parameters["@classify"].Value = classify_temp;

sql_insert_cmd.Parameters.Add("@product", SqlDbType.NVarChar);
sql_insert_cmd.Parameters["@product"].Value = _ProductName.Text;

sql_insert_cmd.Parameters.Add("@total_number", SqlDbType.NVarChar);
sql_insert_cmd.Parameters["@total_number"].Value = _TotalLots.Text;

sql_insert_cmd.Parameters.Add("@low_price", SqlDbType.NVarChar);
sql_insert_cmd.Parameters["@low_price"].Value = "0";

sql_insert_cmd.Parameters.Add("@high_price", SqlDbType.NVarChar);
sql_insert_cmd.Parameters["@high_price"].Value = "0";

sql_insert_cmd.Parameters.Add("@status", SqlDbType.NVarChar);
sql_insert_cmd.Parameters["@status"].Value = "onsale";

sql_insert_cmd.Parameters.Add("@official_price", SqlDbType.Int);
sql_insert_cmd.Parameters["@official_price"].Value = int.Parse(_OfficialPrice.Text);

sql_insert_cmd.Parameters.Add("@seller_ID", SqlDbType.Int);
sql_insert_cmd.Parameters["@seller_ID"].Value = seller_ID;

sql_insert_cmd.Parameters.Add("@pic_pathname", SqlDbType.NVarChar);
sql_insert_cmd.Parameters["@pic_pathname"].Value = _ImgGood.ImageUrl;

sql_insert_cmd.Parameters.Add("@description", SqlDbType.NVarChar);
sql_insert_cmd.Parameters["@description"].Value = _GoodDesc.Text;

sql_insert_cmd.ExecuteNonQuery();
```

g.

h.

i. Datalist 顯示資料字數過多時，使用 DataBound 隱藏部分資料:

j. 資料量讀取過多時，影響欄位高度及整齊度，使用 Datalist 中 OnItemDataBound 與 ToolTip 功能，可以將字數過多的商品描述，進行隱藏，已方便排版。如下圖所示。

```
0 個參考
protected void product_view_DataBound(object sender, EventArgs e)
{
    // 演示ToolTip，使用GridView自帶的ToolTip
    for (int i = 0; i < product_view.Items.Count; i++)
    {
        product_view.Items[i].ToolTip = ((Label)product_view.Items[i].FindControl("description")).Text;
        if (((Label)product_view.Items[i].FindControl("description")).Text.Length > 150)
        {
            ((Label)product_view.Items[i].FindControl("description")).Text =
                ((Label)product_view.Items[i].FindControl("description")).Text.Substring(0, 125) + ".....";
        }
    }
}
```

k.

l. 熱門商品選擇 SQL 語法篩選機制(結標時間越近越前面)(可用 downlist)

m. 商品列表改為搜尋結標時間快結束篩選條件

n. 首頁左側下方固定商品圖案改成近期已結標商品展示(沒功能東西刪除)

o. 聯絡我們(email)

p. 當商品得標者被更換時通知新舊得標者(email)

q. 當商品結標通知得標者(email)
r. 當得標者付款通知得標商品資訊(email)
s. 管理後臺(會員訂單部分不給予修改、刪除，已上架、已下標以及已得標商品，對於其他下標購款手續費會員不公平，若上架後悔標之情況給予懲罰等等條件規則，因此就不增加會員商品列表查詢中的修改刪除。相關功能會員商品修改刪除，將坐落於管理者介面當中。)
t. 得標頁面-計算此筆下標共花了多少手續費
u. 得標頁面-修改顯示內容物
v. 商品數量為一個 set 不計算數量刪減
w. 商品列表分類列於 home/商品列表/分類
x. 美化網站 CSS

7. 未來展望

a. 原本的規劃為：
打造一個結合購物商城、拍賣網、低價競標網站的三合一平台，礙於時間及人力因素考量，捨棄了購物商城及拍賣網兩部分，專心完成低價競標網站，希望未來有能力也把另外兩部分做完。

b. 部分網頁功能尚未完善，如：
更為直觀的商品列表呈現方式、商品列表多頁呈現、下標方式人性化(如：一次可下多標、下標價位可用區間選取等等)，皆是我們未來可以完善、做更好的地方，希望未來能將這些功能做的更完善。

c. 目前尚未實測我們網站對於流量過大時是否會過度負荷，造成頁面載入時間過長，影響用戶體驗。未來希望能調整我們網站的效能，縮短用戶載入頁面時間，才能帶給用戶良好的體驗。

8. 甘特圖

5/19~5/23 製作關聯圖

5/23~5/26 製作資料庫

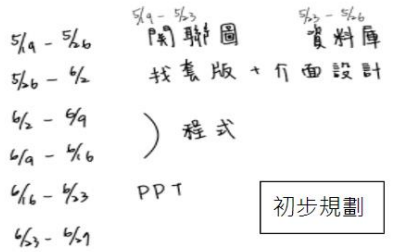
5/26~6/2 找網頁套版、網站介面設計

6/2~6/9 程式製作

6/9~6/16 程式製作

6/16~6/23 程式Debug、Azure雲端部署

6/23~6/27 PPT簡報製作



初步規劃

實際執行

日期	6/28 - 7/5	7/5 - 7/12	7/12 - 7/19	7/19 - 7/26
工作內容	寫出 word 初版 找出待補充功能	補充撰寫程式碼 修正功能 管理系統	更新 word ppt 檔 更新程式碼 上雲端 azure	交出去 debug 及時修正