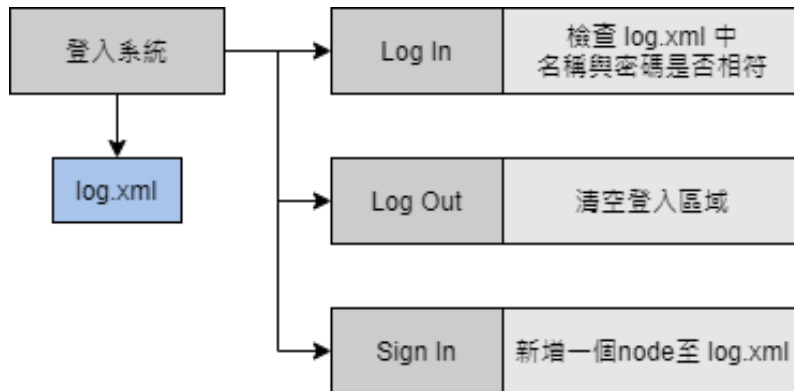


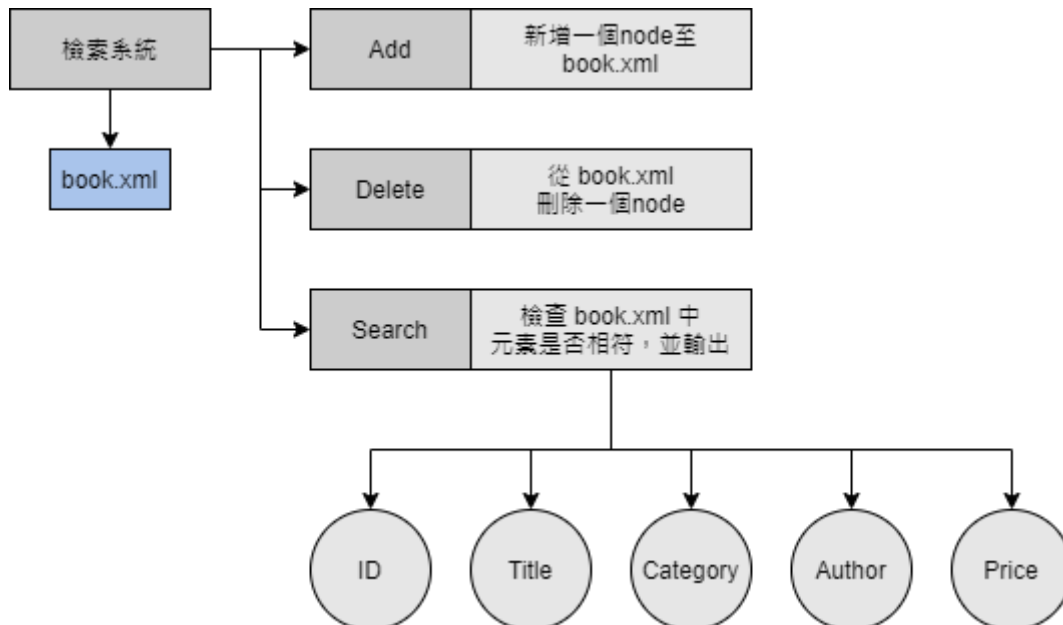
XML 動態網頁實習(作業三)

一、 功能簡介

1. 登入系統



2. 檢索系統



二、實作概念

1. XML 檔建立 (為方便，先以英文資料測試)

```
2. <?xml version="1.0" encoding="UTF-8"?>
```

(a) 登入系統(log.xml)

```
# coding_style.css  log.xml  X  book.xml  JS goto.js  <> main.html

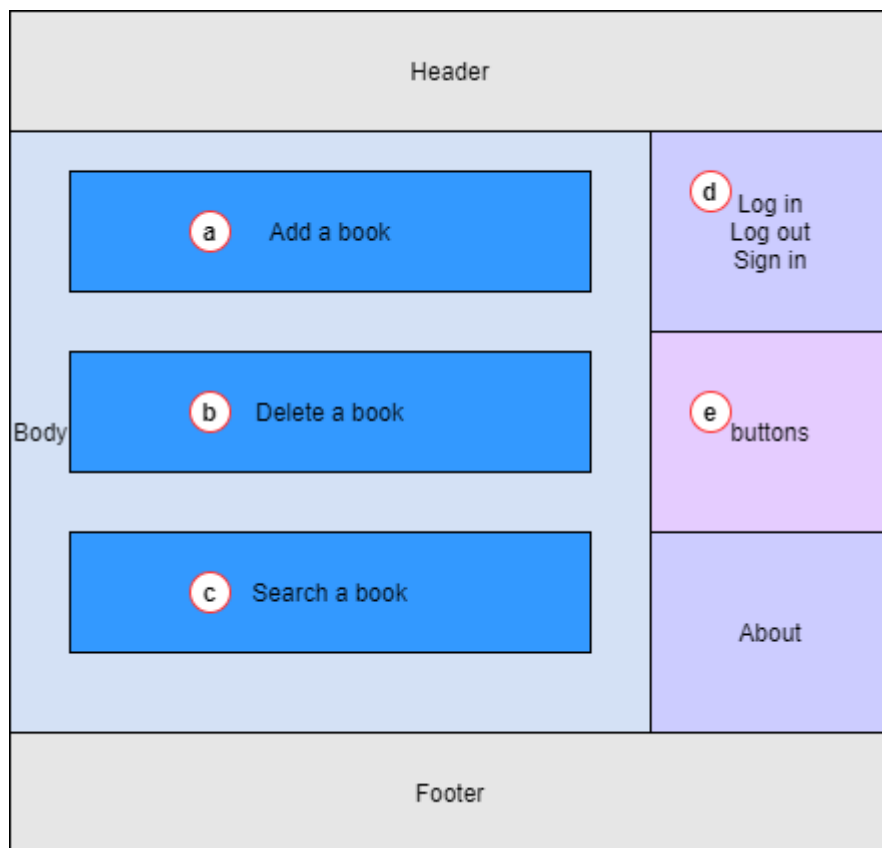
log.xml
1  <root>
2      <person id="01">
3          <user>test1</user>
4          <password>0001</password>
5      </person>
6      <person id="02">
7          <user>test2</user>
8          <password>0003</password>
9      </person>
10     <person id="04">
11         <user>test4</user>
12         <password>0004</password>
13     </person>
14     <person id="05">
15         <user>test5</user>
16         <password>0005</password>
17     </person>
18 </root>
```

(b) 檢索系統(book.xml)

```
# coding_style.css  log.xml  book.xml  X  JS goto.js  <> main.html

book.xml
1  <root>
2      <book id="01" category="textbook">
3          <author>Larson</author>
4          <title>Essential Calculus</title>
5          <price>975</price>
6      </book>
7      <book id="02" category="textbook">
8          <author>Rubin</author>
9          <title>Essential Scrum</title>
10         <price>1025</price>
11     </book>
12     <book id="03" category="textbook">
13         <author>carlson</author>
14         <title>Curcuits</title>
15         <price>575</price>
16     </book>
17     <book id="04" category="novel">
18         <author>Collins</author>
19         <title>Catching Fire</title>
20         <price>975</price>
21     </book>
22     <book id="05" category="novel">
23         <author>Paterson</author>
24         <title>Terabithia</title>
25         <price>975</price>
26     </book>
27 </root>
```

3. 主頁面建立



- (a) 輸入各細項，以在 xml 檔案中新增 node。
- (b) 輸入 Book ID，藉 ID 查詢書籍資料，並加以刪除。
- (c) 輸入某一項資料，並進行查詢，以表格行式輸出符合條件的 node。
- (d) 建立兩個 block，一個是 Log in 和 Sign in 畫面，另一個是 Log out 畫面，藉由兩個畫面交互出現，以實現登入登出的狀態改變
- (e) 普通按鈕（原本想根據功能分網頁，但因為寫在一起比較好展示也比較方便 debug，所以這部份僅裝飾用。）

三、 實作說明

1. 主網頁樣式

Add a book

Book ID:

Title:

Category:

Auther:

Price:

User Name:

Password:

Relative post

Delete a book

Book ID:

Search a book

Book ID:


Title:

Category:

Auther:

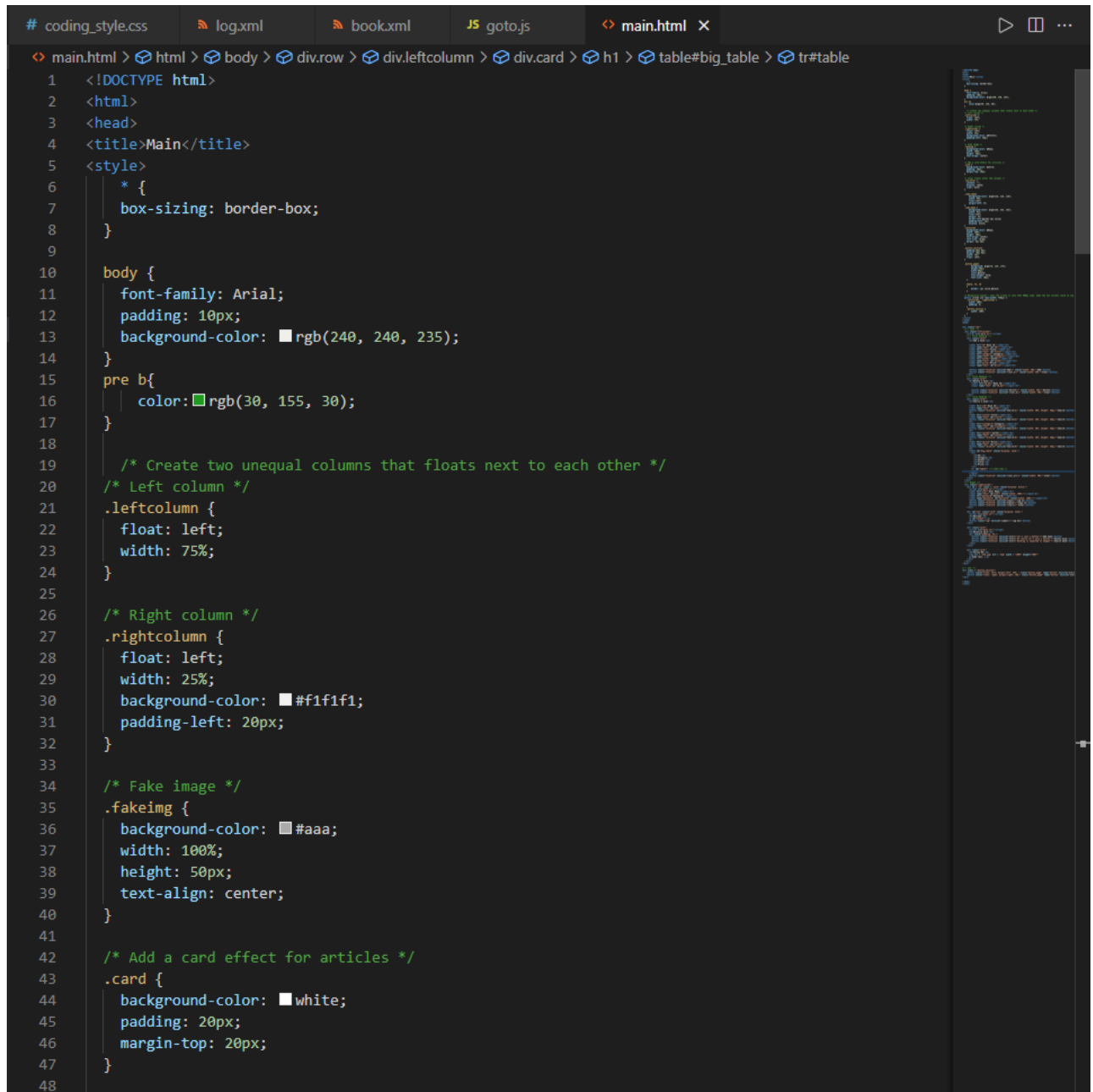
Price:

Follow Me



Some text..

2. HTML



The image shows a code editor with a dark theme. The top bar displays several open files: `# coding_style.css`, `log.xml`, `book.xml`, `JS goto.js`, and `main.html` (which is the active file). The breadcrumb navigation for the active file is: `main.html > html > body > div.row > div.leftcolumn > div.card > h1 > table#big_table > tr#table`.

The code in the editor is as follows:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Main</title>
5 <style>
6     * {
7         box-sizing: border-box;
8     }
9
10    body {
11        font-family: Arial;
12        padding: 10px;
13        background-color: #rgb(240, 240, 235);
14    }
15    pre b{
16        color: #rgb(30, 155, 30);
17    }
18
19    /* Create two unequal columns that floats next to each other */
20    /* Left column */
21    .leftcolumn {
22        float: left;
23        width: 75%;
24    }
25
26    /* Right column */
27    .rightcolumn {
28        float: left;
29        width: 25%;
30        background-color: #f1f1f1;
31        padding-left: 20px;
32    }
33
34    /* Fake image */
35    .fakeimg {
36        background-color: #aaa;
37        width: 100%;
38        height: 50px;
39        text-align: center;
40    }
41
42    /* Add a card effect for articles */
43    .card {
44        background-color: #white;
45        padding: 20px;
46        margin-top: 20px;
47    }
48
```

```

49  /* Clear floats after the columns */
50  .row:after {
51      content: "";
52      display: table;
53      clear: both;
54  }
55
56  .code_name{
57      background-color: #rgb(218, 218, 218);
58      width: 96%;
59      float:left;
60      margin-left: 2%;
61  }
62  .code_demo {
63      background-color: #rgb(233, 233, 233);
64      width: 92%;
65      float:left;
66      margin: 4%;
67      border-left: 5px solid green;
68      padding-left: 5px;
69      display: block;
70  }
71  .relative{
72      background-color: #aaa;
73      width: 100%;
74      height: 50px;
75      text-align: center;
76      font-size: large;
77      margin: 5px 0px;
78  }
79
80  .button_section{
81      padding:10px 0px;
82      margin: 10px 0px;
83      width: 75%;
84      float: left;
85  }
86
87  .button_page{
88      background: #rgb(76, 129, 179);
89      height:40px;
90      width:80px;
91      color:white;
92      font-weight: bold;
93      font-size: 20px;
94  }
95

```

```

96  table, th, td
97  {
98      border: 1px solid black;
99  }
100
101  /* Responsive layout - when the screen is less than 800px wide, make the two columns stack on top
102  @media screen and (max-width: 750px) {
103      .leftcolumn, .rightcolumn {
104          width: 100%;
105          padding: 0;
106      }
107      .button_section {
108          width: 100%;
109      }
110  }
111 </style>
112 </head>

```

```

113
114 <body>
115 <div class="row">
116   <!-- Left -->
117   <div class="leftcolumn">
118     <script src="goto.js"></script>
119     <!-- Title Heading -->
120     <div class="card">
121       <h1>Add a book</h2>
122
123       <label for="ID">Book ID:</label><br>
124       <input type="text" id="ID"></input><br>
125       <label for="title">Title:</label><br>
126       <input type="text" id="title"></input><br>
127       <label for="category">Category:</label><br>
128       <input type="text" id="category"></input><br>
129       <label for="auther">Auther:</label><br>
130       <input type="text" id="auther"></input><br>
131       <label for="price">Price:</label><br>
132       <input type="text" id="price"></input><br>
133
134       <button class="relative" onclick="Add()" style="width: 30%;">Add</button>
135       <button class="relative" onclick="clear_a()" style="width: 30%;">Clear</button>
136     </div>
137     <!-- Title Heading -->
138     <div class="card">
139       <h1>Delete a book</h2>
140       <label for="ID_del">Book ID:</label><br>
141       <input type="text" id="ID_del"></input><br>
142
143       <button class="relative" onclick="Delete()" style="width: 30%;">Delete</button>
144       <button class="relative" onclick="clear_d()" style="width: 30%;">Clear</button>
145     </div>
146   <!-- Title Heading -->
147   <div class="card">
148     <h1>Search a book</h2>
149
150     <label for="sID">Book ID:</label><br>
151     <input type="text" id="sID"></input>
152     <button class="relative" onclick="Search(1)" style="width: 30%; height: 30px;">Search</button>
153     <br>
154     <label for="stitle">Title:</label><br>
155     <input type="text" id="stitle"></input>
156     <button class="relative" onclick="Search(2)" style="width: 30%; height: 30px;">Search</button>
157     <br>
158     <label for="scategory">Category:</label><br>
159     <input type="text" id="scategory"></input>
160     <button class="relative" onclick="Search(3)" style="width: 30%; height: 30px;">Search</button>
161     <br>
162     <label for="sauther">Auther:</label><br>
163     <input type="text" id="sauther"></input>
164     <button class="relative" onclick="Search(4)" style="width: 30%; height: 30px;">Search</button>
165     <br>
166     <label for="sprice">Price:</label><br>
167     <input type="text" id="sprice"></input>
168     <button class="relative" onclick="Search(5)" style="width: 30%; height: 30px;">Search</button>
169     <br>
170     <table id="big_table" style="display: none;">
171       <tr>
172         <th>ID</th>
173         <th>Category</th>
174         <th>Author</th>
175         <th>Title</th>
176         <th>Price</th>
177       </tr>
178       <tr id="table"> <!--<td></td>-->
179     </tr>
180   </table>
181   <button class="relative" onclick="clear_all()" style="width: 30%;">Clear</button>
182 </div>
183 </div>

```

```

184 <!-- Right -->
185 <div class="rightcolumn">
186   <div id = "in" class = "card" style="display: block;">
187     <script src="goto.js"></script>
188     <label for="user">User Name:</label><br>
189     <input type="text" id="user" style="width: 100%;"></input><br>
190     <label for="password">Password:</label><br>
191     <input type="password" id="password" style="width: 100%;"></input><br>
192     <button class="relative" onclick="LogIn()">Log In</button>
193     <button class="relative" onclick="SignIn()">Sign In</button>
194     <button class="relative" onclick="clear1()">Clear</button>
195   </div>
196
197   <div id="out" class="card" style="display: none;">
198     <script src="goto.js"></script>
199     <h2>Welcome</h2>
200     <p id="output"></p>
201     <button class="log" onclick="LogOut()">Log Out</button>
202   </div>
203
204   <div class="card">
205     <script src="goto.js"></script>
206     <h3>Relative post</h3>
207     <div style="padding: 5%;">
208       <button class="relative" onclick="alert('It\'s just a button')">Add book</button>
209       <button class="relative" onclick="alert('Of course it\'s also a button')">Delete book</button>
210       <button class="relative" onclick="alert('Nothing is expected to happen')">Search book</button>
211     </div>
212   </div>
213
214   <div class="card">
215     <h3>Follow Me</h3>
216     <img src = "135.jpg" alt = "img" width = "100%" height="100%">
217     <p>Some text..</p>
218   </div>
219 </div>
220 </div>
221
222 <!-- end -->
223 <div class = "button_section">
224   <button style="float: left; margin-left: 10%; " class="button_page" type="button"
225     onclick="alert('previous')">&lt; Pre</button>
226   <button style="float: right; margin-right: 10%;" class="button_page" type="button"
227     onclick="alert('next')">Next &gt;</button>
228 </div>
229
230 </body>
231 </html>

```


3. CSS

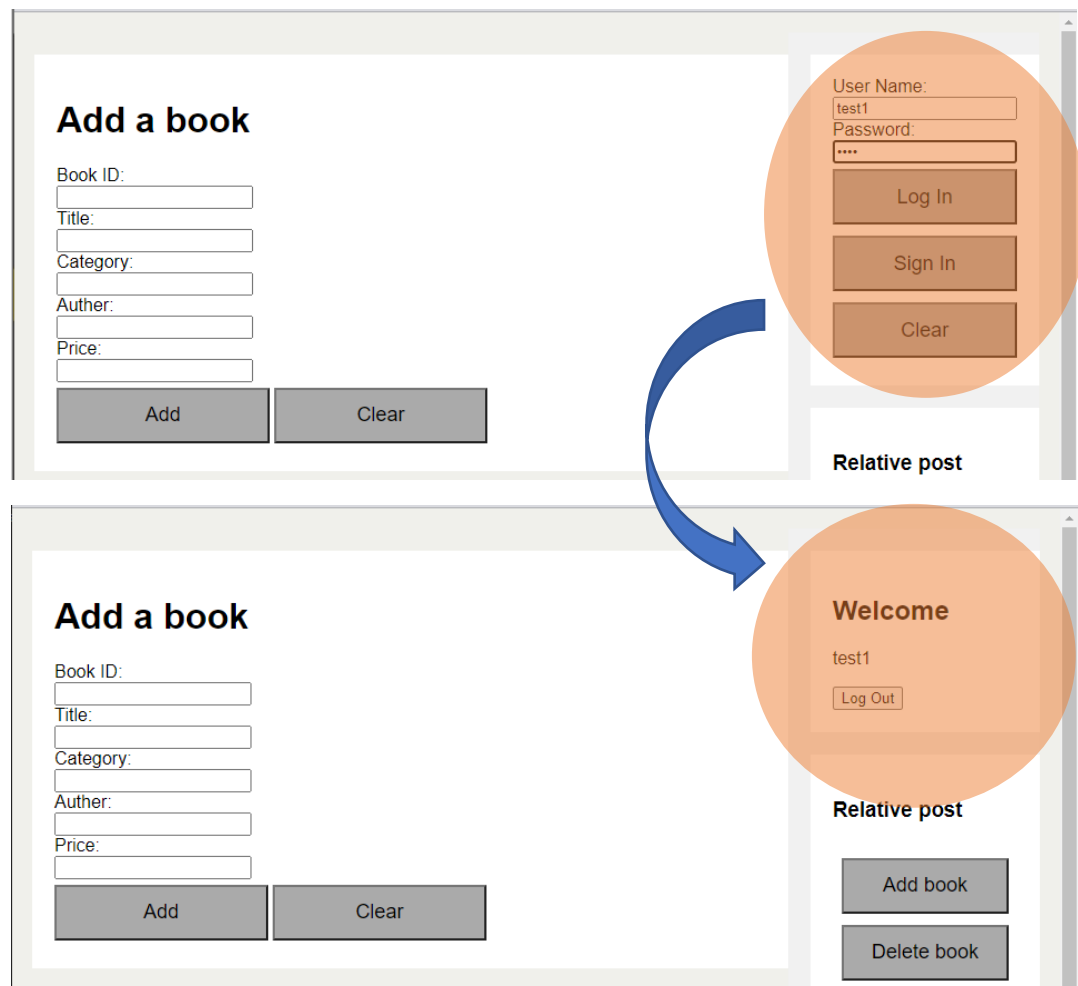
```
# coding_style.css X log.xml book.xml JS goto.js main.html
# coding_style.css >
1  * {
2    box-sizing: border-box;
3  }
4
5  body {
6    font-family: Arial;
7    padding: 10px;
8    background-color: #rgb(240, 240, 235);
9  }
10 pre b{
11   color: #rgb(30, 155, 30);
12 }
13
14 /* Create two unequal columns that floats next to each other */
15 /* Left column */
16 .leftcolumn {
17   float: left;
18   width: 75%;
19 }
20
21 /* Right column */
22 .rightcolumn {
23   float: left;
24   width: 25%;
25   background-color: #f1f1f1;
26   padding-left: 20px;
27 }
28
29 /* Fake image */
30 .fakeimg {
31   background-color: #aaa;
32   width: 100%;
33 }
34
35 /* Add a card effect for articles */
36 .card {
37   background-color: #white;
38   padding: 20px;
39   margin-top: 20px;
40 }
41
42 /* Clear floats after the columns */
43 .row:after {
44   content: "";
45   display: table;
46   clear: both;
47 }
```

```
48
49 .code_name{
50   background-color: #rgb(218, 218, 218);
51   width: 96%;
52   float:left;
53   margin-left: 2%;
54 }
55 .code_demo {
56   background-color: #rgb(233, 233, 233);
57   width: 92%;
58   float:left;
59   margin-left: 4%;
60   margin-bottom: 4%;
61   border-left: 5px solid green;
62   padding-left: 5px;
63   display: block;
64 }
65
66 .button_section{
67   padding:10px 0px;
68   margin: 10px 0px;
69   width: 75%;
70   float: left;
71 }
72
73 .button_page{
74   background: #rgb(76, 129, 179);
75   height:40px;
76   width:80px;
77   color:#white;
78   font-weight: bold;
79   font-size: 20px;
80 }
81
82 /* Responsive layout - when the screen is less than 800px wide, make the two columns stack on top
83 @media screen and (max-width: 600px) {
84   .leftcolumn, .rightcolumn {
85     width: 100%;
86     padding: 0;
87   }
88   .button_section {
89     width: 100%;
90   }
91 }
92
93 /*
94 <1 Title Heading >
```

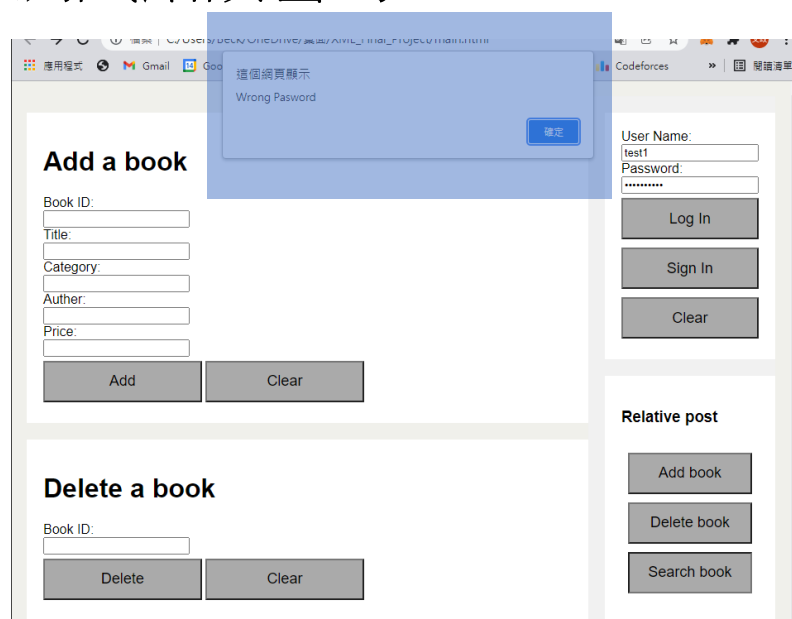
4. Javascript 功能實作

(a) Log in

(i) 測試正確密碼



(ii) 測試錯誤密碼



(iii) Code :

```
9  function LogIn()
10 {
11     // request
12     const xhttp = new XMLHttpRequest();
13     xhttp.onload = function()
14     {
15         log_in(this);
16     }
17     xhttp.open("GET", "log.xml");
18     xhttp.send();
19 }
20 function log_in(xml)
21 {
22     const xmlDoc = xml.responseXML;
23     const x = xmlDoc.getElementsByTagName("person");
24     var n = document.getElementById("user").value;
25     var p = document.getElementById("password").value;
26     var i = 0;
27     // go through all <person>
28     for (i = 0; i < x.length; i++)
29     {
30         // check user name
31         if( x[i].getElementsByTagName("user")[0].childNodes[0].nodeValue == n )
32         {
33             // check password
34             if( x[i].getElementsByTagName("password")[0].childNodes[0].nodeValue == p )
35             {
36                 // close the log in block
37                 document.getElementById('in').style.display='none';
38                 // show the log out block
39                 document.getElementById('out').style.display='block';
40                 // display user name
41                 document.getElementById("output").textContent = p;
42             }
43             else
44             {
45                 alert("Wrong Password");
46             }
47         }
48     }
49     if( i == x.length)
50     {
51         alert("No this person");
52     }
53 }
```

(b) Sign in

(i) 設定密碼後再登入

The diagram illustrates the user flow for signing in after setting a password. It consists of two screenshots of a web application interface, connected by a blue curved arrow indicating the transition.

Top Screenshot (Registration Page):

- Add a book** section with input fields for Book ID, Title, Category, Author, and Price, and 'Add' and 'Clear' buttons.
- Right sidebar:**
 - User Name:**
 - Password:**
 - Log In** button
 - Sign In** button
 - Clear** button
- Relative post** section (empty).

Bottom Screenshot (Login Page):

- Add a book** section (identical to the top screenshot).
- Right sidebar:**
 - Welcome** message
 - addnewmem** (username)
 - Log Out** button
- Relative post** section with buttons: **Add book**, **Delete book**, and **Search book**.

A blue curved arrow points from the 'Sign In' button in the top screenshot to the 'Welcome' message in the bottom screenshot, indicating the successful login process.

Code :

```
64
65 function SignIn()
66 {
67     // request
68     const xhttp = new XMLHttpRequest();
69     xhttp.onload = function()
70     {
71         sign_in(this);
72     }
73     xhttp.open("GET", "log.xml");
74     xhttp.send();
75 }
76
```

```
77 function sign_in(xml)
78 {
79     var xmlDoc = xml.responseXML;
80
81     var name = document.getElementById("user").value;
82     var password = document.getElementById("password").value;
83     var num = xmlDoc.length;
84     // num shows the last index after adding new <person>
85
86     // set <person> node
87     var newElement = xmlDoc.createElement("person");
88     var x = xmlDoc.getElementsByTagName("root")[0];
89     x.appendChild(newElement);
90     // set <person> attr
91     var y = xmlDoc.getElementsByTagName("person");
92     var y_len = y.length;
93     // set the person id every time
94     // it doesn't really matter, cause only sever know the id
95     for(let i = 0; i < y_len; i++)
96     {
97         newAtt = xmlDoc.createAttribute("id");
98         newAtt.value = "num";
99         x[i].setAttributeNode(newAtt);
100     }
101     // set user
102     newEle = xmlDoc.createElement("user");
103     newText = xmlDoc.createTextNode(name);
104     newEle.appendChild(newText);
105     x = xmlDoc.getElementsByTagName("person")[num];
106     x.appendChild(newEle);
107
108     // set password
109     newEle = xmlDoc.createElement("password");
110     newText = xmlDoc.createTextNode(password);
111     newEle.appendChild(newText);
112     x = xmlDoc.getElementsByTagName("person")[num];
113     x.appendChild(newEle);
114
115 }
```

(c) Log out

The diagram illustrates the state change during a log out operation. The top interface shows a user logged in as 'test1' with a 'Log Out' button. The bottom interface shows the user logged out, with the 'Log Out' button replaced by a login/sign-in section containing 'User Name', 'Password', 'Log In', 'Sign In', and 'Clear' buttons. The 'Add a book' form remains visible in both states.

Top Interface (Logged In):

- Add a book**
- Book ID:
- Title:
- Category:
- Author:
- Price:
-
- Welcome**
test1
- Relative post**

Bottom Interface (Logged Out):

- Add a book**
- Book ID:
- Title:
- Category:
- Author:
- Price:
-
- User Name:**
Password:
-
- Relative post**
- Delete a book**

Code :

```
55 function Logout()
56 {
57     // close log out block
58     document.getElementById('out').style.display='none';
59     // show the log i block
60     document.getElementById('in').style.display='block';
61     // clear the user name in log out block
62     document.getElementById("output").textContent = "";
63 }
64
```

(d) Add

Add a book

Book ID:

Title:

Category:

Author:

Price:

User Name:

Password:

Relative post

See the xml file

```
book.xml
1  <?xml version="1.0" encoding="UTF-8"?>
2  <root>
3      <book id="01" category="textbook">
4          <author>Larson</author>
5          <title>Essential Calculus</title>
6          <price>975</price>
7      </book>
8      <book id="02" category="textbook">
9          <author>Rubin</author>
10         <title>Essential Scrum</title>
11         <price>1025</price>
12     </book>
13     <book id="03" category="textbook">
14         <author>carlson</author>
15         <title>Curcuits</title>
16         <price>575</price>
17     </book>
18     <book id="04" category="novel">
19         <author>Collins</author>
20         <title>Catching Fire</title>
21         <price>975</price>
22     </book>
23     <book id="05" category="novel">
24         <author>Paterson</author>
25         <title>Terabithia</title>
26         <price>975</price>
27     </book>
28     <book id="14" category="novel">
29         <title>addabook</title>
30         <author>me</author>
31         <price>1</price>
32     </book>
33 </root>
```

(我們可以看到新資料被 **append** 到最後一項，但順序錯誤，因為我的 javascript 在新增<book>的 child 時順序寫反了，但不影響整體程式邏輯。)

Code :

```
160 function clear_a()
161 {
162     document.getElementById("ID").value = "";
163     document.getElementById("category").value = "";
164     document.getElementById("author").value = "";
165     document.getElementById("title").value = "";
166     document.getElementById("price").value = "";
167 }
168 function Add(): void
169 function Add()
170 {
171     // request
172     var xhttp = new XMLHttpRequest();
173     xhttp.onreadystatechange = function()
174     {
175         if(this.readyState == 4 && this.status == 200)
176         {
177             add(this);
178         }
179     };
180     xhttp.open("GET", "book.xml", true);
181     xhttp.send();
182 }
183
184 function add(xml)
185 {
186     var xmlDoc = xml.responseXML;
187
188     // get value from <input>
189     var id = document.getElementById("ID").value;
190     var cat = document.getElementById("category").value;
191     var author = document.getElementById("author").value;
192     var title = document.getElementById("title").value;
193     var price = document.getElementById("price").value;
194     var num = xmlDoc.length;
195
196     // set <book> node
197     var newElement = xmlDoc.createElement("book");
198     var x = xmlDoc.getElementsByTagName("root")[0];
199     x.appendChild(newElement);
200
201     // s (local var) y: any
202     var y = xmlDoc.getElementsByTagName("book");
203     var y_len = y.length;
204
205     newAtt = xmlDoc.createAttribute("id");
206     newAtt.value = id;
207     x[y_len-1].setAttributeNode(newAtt);
208
209     // set <book> attr cat
210     var y = xmlDoc.getElementsByTagName("book");
211     var y_len = y.length;
212
213     newAtt = xmlDoc.createAttribute("category");
214     newAtt.value = cat;
215     x[y_len-1].setAttributeNode(newAtt);
216
217     // set title
218     newEle = xmlDoc.createElement("title");
219     newText = xmlDoc.createTextNode(title);
220     newEle.appendChild(newText);
221     x = xmlDoc.getElementsByTagName("book")[num];
222     x.appendChild(newEle);
223
224     // set author
225     newEle = xmlDoc.createElement("author");
226     newText = xmlDoc.createTextNode(author);
227     newEle.appendChild(newText);
228     x = xmlDoc.getElementsByTagName("book")[num];
229     x.appendChild(newEle);
230
231     // set price
232     newEle = xmlDoc.createElement("price");
233     newText = xmlDoc.createTextNode(price);
234     newEle.appendChild(newText);
235     x = xmlDoc.getElementsByTagName("book")[num];
236     x.appendChild(newEle);
237 }
238
```


(e) Delete

Delete a book

Book ID:

Search a book

[Follow Me](#)

See the xml file

```
book.xml
1  <?xml version="1.0" encoding="UTF-8"?>
2  <root>
3    <book id="01" category="textbook">
4      <author>Larson</author>
5      <title>Essential Calculus</title>
6      <price>975</price>
7    </book>
8    <book id="02" category="textbook">
9      <author>Rubin</author>
10     <title>Essential Scrum</title>
11     <price>1025</price>
12   </book>
13   <book id="04" category="novel">
14     <author>Collins</author>
15     <title>Catching Fire</title>
16     <price>975</price>
17   </book>
18   <book id="05" category="novel">
19     <author>Paterson</author>
20     <title>Terabithia</title>
21     <price>975</price>
22   </book>
23   <book id="14" category="novel">
24     <title>addabook</title>
25     <author>me</author>
26     <price>1</price>
27   </book>
28 </root>
```

(我們可以看到 id = 3 的 node 不見了)

Code :

```
119 function clear_d()
120 {
121     document.getElementById("ID_del").value = "";
122 }
123
124 function Delete()
125 {
126     // request
127     var xhttp = new XMLHttpRequest();
128     xhttp.onreadystatechange = function()
129     {
130         if(this.readyState == 4 && this.status == 200)
131         {
132             del(this);
133         }
134     };
135     // need 'true' if we're changing XML File
136     xhttp.open("GET", "book.xml", true);
137     xhttp.send();
138 }
139
140 function del(xml)
141 {
142     var get_id = document.getElementById("ID_del").value;
143
144     var xmlDoc = xml.responseXML;
145     var x = xmlDoc.getElementsByTagName("book");
146
147     // find node
148     for(let i = 0; i < x.length; i++)
149     {
150         if(x[i].getAttribute('id') == get_id )
151             // because the id is the only value
152             {
153                 // delete it
154                 x = x[i];
155                 x.parentNode.removeChild(x);
156             }
157     }
158 }
159
```

(f) Search

(在此我只測試，title 和 category，因為兩者分別為找 node 和找 attribute，而其他三項都能被相同邏輯涵蓋。)

Search a book

Book ID:

Title:


Category:

Author:

Price:

ID	Category	Author	Title	Price
14	novel	me	addabook	1

Follow Me



Some text..

Search a book

Book ID:

Title:


Category:

Author:

Price:

ID	Category	Author	Title	Price
04	novel	Collins	Catching Fire	975
05	novel	Paterson	Terabithia	975
14	novel	me	addabook	1

Follow Me



Some text..

And of course to clear

Search a book

Book ID:


Title:

Category:

Author:

Price:

Follow Me



Some text..

[< Pre](#)[Next >](#)

Code :

```
239 // search book
240
241 function clear_all()
242 {
243     document.getElementById("sID").value = "";
244     document.getElementById("scategory").value = "";
245     document.getElementById("sauthor").value = "";
246     document.getElementById("stitle").value = "";
247     document.getElementById("sprice").value = "";
248 }
249
250 function Search(op)
251 {
252     // request
253     const xhttp = new XMLHttpRequest();
254     xhttp.onload = function()
255     {
256         document.getElementById('big_table').style.display='block';
257         switch(op)
258         {
259             case 1:
260                 search1(this);// id
261                 break;
262             case 2:
263                 search2(this);// title
264                 break;
265             case 3:
266                 search3(this);// category
267                 break;
268             case 4:
269                 search4(this);// author
270                 break;
271             case 5:
272                 search5(this);// price
273                 break;
274             default:
275                 break;
276         }
277     }
278     xhttp.open("GET", "book.xml");
279     xhttp.send();
280 }
281
```

```

281
282 function search1(xml)
283 {
284     var get_id = document.getElementById("sID").value;
285
286     var xmlDoc = xml.responseXML;
287     var x = xmlDoc.getElementsByTagName("book");
288
289     var txt = "";
290
291     // find node
292     for(let i = 0; i < x.length; i++)
293     {
294         if(x[i].getAttribute('id') == get_id )
295             // because the id is the only value
296             {
297                 txt = "<td>" + x[i].getAttribute('id') + "</td>" ;
298                 txt = "<td>" + x[i].getAttribute('category') + "</td>" ;
299                 txt += "<td>" + x[i].getElementsByTagName("author") + "</td>" ;
300                 txt += "<td>" + x[i].getElementsByTagName("title") + "</td>" ;
301                 txt += "<td>" + x[i].getElementsByTagName("price") + "</td>" ;
302                 txt += "</tr><tr>" ;
303                 document.getElementById('table').innerHTML = txt;
304             }
305     }
306 }

```

```

307 function search2(xml)
308 {
309     var get = document.getElementById("stitle").value;
310
311     var xmlDoc = xml.responseXML;
312     var x = xmlDoc.getElementsByTagName("book");
313
314     var txt = "";
315
316     // find node
317     for(let i = 0; i < x.length; i++)
318     {
319         if(x[i].getElementsByTagName("title") == get )
320             // because the id is the only value
321             {
322                 txt = "<td>" + x[i].getAttribute('id') + "</td>" ;
323                 txt = "<td>" + x[i].getAttribute('category') + "</td>" ;
324                 txt += "<td>" + x[i].getElementsByTagName("author") + "</td>" ;
325                 txt += "<td>" + x[i].getElementsByTagName("title") + "</td>" ;
326                 txt += "<td>" + x[i].getElementsByTagName("price") + "</td>" + "</tr><tr>";
327                 document.getElementById('table').innerHTML = txt;
328             }
329     }
330 }

```

```

332 function search3(xml)
333 {
334     var get = document.getElementById("scategory").value;
335
336     var xmlDoc = xml.responseXML;
337     var x = xmlDoc.getElementsByTagName("book");
338
339     var txt = "";
340
341     // find node
342     for(let i = 0; i < x.length; i++)
343     {
344         if(x[i].getAttribute('category') == get )
345             // because the id is the only value
346             {
347                 txt = "<td>" + x[i].getAttribute('id') + "</td>" ;
348                 txt = "<td>" + x[i].getAttribute('category') + "</td>" ;
349                 txt += "<td>" + x[i].getElementsByTagName("author") + "</td>" ;
350                 txt += "<td>" + x[i].getElementsByTagName("title") + "</td>" ;
351                 var document: Document ElementsByTagName("price") + "</td>" + "</tr><tr>" ;
352                 document.getElementById('table').innerHTML = txt;
353             }
354     }
355 }

```

```

356 function search4(xml)
357 {
358     var get = document.getElementById("sauthor").value;
359
360     var xmlDoc = xml.responseXML;
361     var x = xmlDoc.getElementsByTagName("book");
362
363     var txt = "";
364
365     // find node
366     for(let i = 0; i < x.length; i++)
367     {
368         if(x[i].getElementsByTagName("author") == get )
369             // because the id is the only value
370             {
371                 txt = "<td>" + x[i].getAttribute('id') + "</td>" ;
372                 txt = "<td>" + x[i].getAttribute('category') + "</td>" ;
373                 txt += "<td>" + x[i].getElementsByTagName("author") + "</td>" ;
374                 txt += "<td>" + x[i].getElementsByTagName("title") + "</td>" ;
375                 txt += "<td>" + x[i].getElementsByTagName("price") + "</td>" + "</tr><tr>" ;
376                 document.getElementById('table').innerHTML = txt;
377             }
378     }
379 }

```

```

380 function search5(xml)
381 {
382     var get = document.getElementById("sprice").value;
383
384     var xmlDoc = xml.responseXML;
385     var x = xmlDoc.getElementsByTagName("book");
386
387     var txt = "";
388
389     // find node
390     for(let i = 0; i < x.length; i++)
391     {
392         if(x[i].getElementsByTagName("price") == get )
393             // because the id is the only value
394             {
395                 txt = "<td>" + x[i].getAttribute('id') + "</td>" ;
396                 txt = "<td>" + x[i].getAttribute('category') + "</td>" ;
397                 txt += "<td>" + x[i].getElementsByTagName("author") + "</td>" ;
398                 txt += "<td>" + x[i].getElementsByTagName("title") + "</td>" ;
399                 txt += "<td>" + x[i].getElementsByTagName("price") + "</td>" + "</tr><tr>" ;
400                 document.getElementById('table').innerHTML = txt;
401             }
402     }
403 }
404

```

補充：

在這次作業中 XML 檔案我是用 HttpRequest 取得，而在前面的 xml 檔是我為了一致性才 pull 到 local 用 VScode 打開並截圖的。

四、討論(心得?)

先從簡單的開始好了，整個版型我是沿用了我自己習慣的版型，在前幾次作業也可以看到，然後這次將所有的功能都實作在同一個網頁上，因為我認為這樣很一目了然，版面也不會空空的，當然我一開始構想是希望能實現網頁跳轉的部分，所以才有右下的三個按鈕。

那不使用網頁跳轉就會出現一個問題，要怎麼顯示登入前登入後的差異，那我想到的方式是把兩種顯示都做出來，然後藉由改變 `"display:none;"` 和 `"display: block;"` 來實現轉換的功能。

接下來就是資料輸出輸入的部分，老實說，我原本寫了兩個 c，能轉換 csv 形式的文檔變成 xml 檔，但我最後卡在 c 沒辦法跑進我的網頁裡，我試了很多方法，最後放棄回去用 DOM(主要是因為我對 DOM 的輸入部分沒那麼熟悉)。

在使用 DOM 接收檔案時，又發現不能直接使用 local xml file，唯一的辦法是手動選取 xml file，我覺得很麻煩，所以我把 xml 丟上 github，直接用 Httprequest，自動抓資料進去。

最後測試部分，除了一些常見的小問題，就是我常常忘記每個 node 現在在哪裡，有一次刪除不小心連 root 都不見了，還有刪除 node 要 `"x.parentNode.removeChild(x)"`，其實蠻不直觀的。