

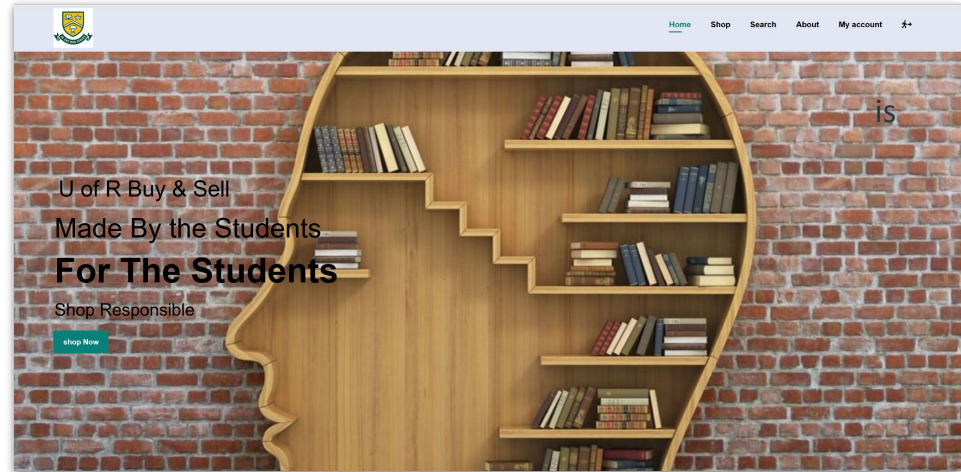
U of R Buy & Sell

CS476 Final Project

Ibrahim Adogba | Wilbur Dulce | James Sieben

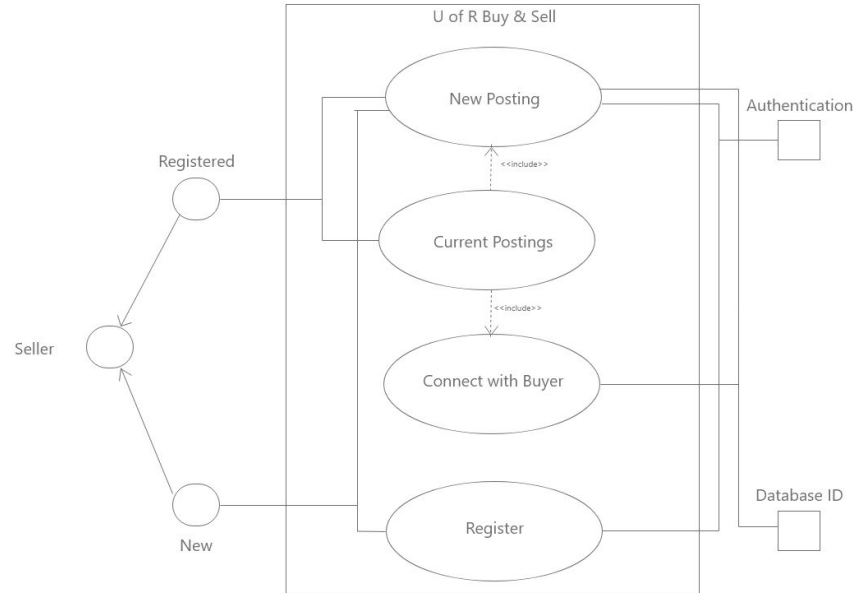
Purpose and Goals

- The purpose of our project is to create a space where students can directly buy and sell textbooks from each other
- Our Buy & Sell website makes it easy for U of R students to upload their old textbooks and connect with other students
- In contrast with Facebook marketplace or Kijiji, our site only allows books and only open to U of R students
- Unlike the school bookstore, books of any quality can be posted and there is no additional markup



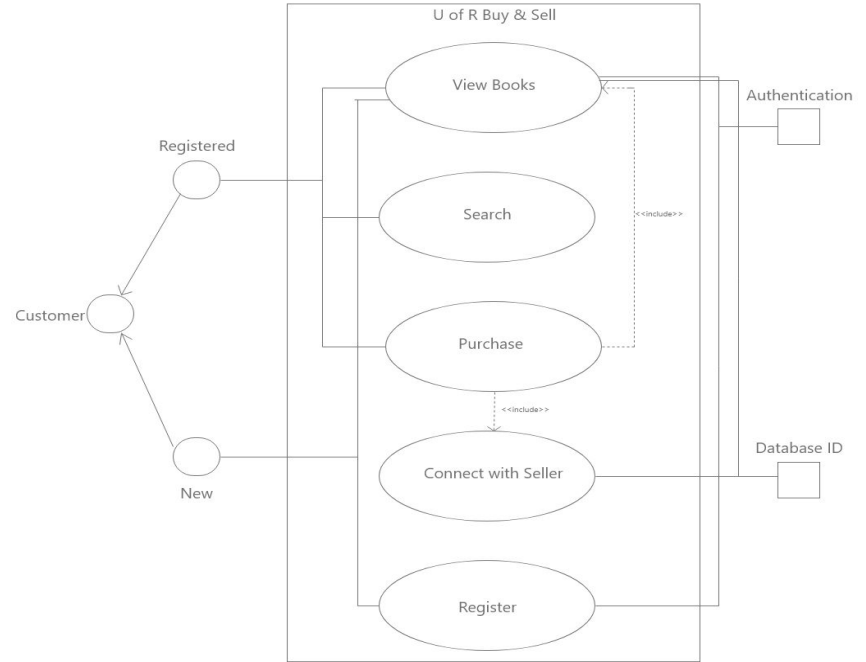
Use Case: Seller

- Sellers must sign up before using our site and must be registered U of R students
- Registered users are asked to login before viewing the site
- Sellers may:
 - Create a new post
 - View their current postings
 - Connect with potential buyers through their student email



Use Case: Buyer

- Similarly, students shopping for books must login before viewing the site
- Users may:
 - View a specific listing
 - Search the site for books by title, author, genre, or ISBN
 - Purchase books by connecting with sellers
- To buy a book, the user will be shown the seller's email so that they may directly communicate



MVC: Controller

- The Controller is responsible for processing input and directing data flow
- In our web app, it handles requests to view books, upload books, and communicate with other users, as well as managing the database to store and organise user and book information
- Ex. If a book is bought, the Controller removes it from the database and updates the View so that book isn't shown anymore
- The screenshot shows the code to search the Model and returns results to the View

```
Back-End > controllers > search.php
16
17
18 }
19
20 $db = new PDO("mysql:host=localhost; dbname=loa388", "loa388", "Dantell12");
21
22 // Retrieve latest three items from the database
23 $sql = "SELECT sellid, selltitle, sellimage, price, selldescription, sellassum, sellishn, sellauthor, created_at FROM sellcreation ORDER BY created_at DESC LIMIT 3";
24 $r1 = $db->query($sql, PDO::FETCH_ASSOC);
25 // Retrieve all items from the database
26
27 $sql2 = "SELECT sellid, selltitle, sellimage, price, selldescription, sellassum, sellishn, sellauthor, created_at FROM sellcreation";
28 $r2 = $db->query($sql2, PDO::FETCH_ASSOC);
29
30 $row = $r2->fetch();
31 $selltitle = $row["selltitle"];
32 $sellauthor = $row["sellauthor"];
33 $sellimage = $row["sellimage"];
34 $selldescription = $row["selldescription"];
35 $sellishn = $row["sellishn"];
36 $price = $row["price"];
37
38 if (isset($_POST["submit"])) {
39     $result = trim($_POST["search"]);
40     $search_by = $_POST["search-by"];
41
42     // Prepare SQL query based on the selected search criteria
43     if ($search_by == "name") {
44         $query = "SELECT * FROM sellcreation WHERE selltitle LIKE :search_term";
45     } else if ($search_by == "author") {
46         $query = "SELECT * FROM sellcreation WHERE sellauthor LIKE :search_term";
47     } else if ($search_by == "isbn") {
48         $query = "SELECT * FROM sellcreation WHERE sellishn LIKE :search_term";
49     } else if ($search_by == "class") {
50         $query = "SELECT * FROM sellcreation WHERE sellassum LIKE :search_term";
51     }
52
53     // Execute the search query
54     $stmt = $db->prepare($query);
55     $stmt->bindValue(":search_term", "%".$result."%");
56     $stmt->setFetchMode(PDO::FETCH_ASSOC);
57     $stmt->execute();
58     $search_results = $stmt->fetchAll();
59 }
60 }
61
```

MVC: Model

- The Model is the data layer, it's responsible for storing, retrieving, and manipulating the data stored in the database
- For our use, it stores info about books and users, such as the book titles, authors, descriptions, and usernames, emails, and addresses
- Our site uses MySQL and JSON requests
- The code shown creates the Users table and the favorites table within the Model

```
[Preview] README.md  Users.sql  X
Database > Users.sql
1 DROP TABLE IF EXISTS Users;
2
3 CREATE TABLE Users (
4     userId int(11) NOT NULL AUTO_INCREMENT,
5     email varchar (50) NOT NULL,
6     photo_img varchar(10000) NOT NULL,
7     pswd varchar (50) NOT NULL,
8     username varchar (50) NOT NULL,
9     PRIMARY KEY (userId)
10 );

[Preview] README.md  favorites.sql  X
Database > favorites.sql
1 DROP TABLE IF EXISTS favorites;
2
3
4 CREATE TABLE favorites (
5     favouriteId int(50) NOT NULL AUTO_INCREMENT,
6     sellId int NOT NULL,
7     userId int NOT NULL,
8     created_dt datetime,
9     PRIMARY KEY (favouriteId),
10    FOREIGN KEY (sellId) REFERENCES sellcreation (sellId),
11    FOREIGN KEY (userId) REFERENCES Users (userId)
12 );
13
14 insert into favorites (favouriteId, sellId, userId, created_dt)
15 values ('1','14','1', '2023-03-17 18:24:03');
16
```

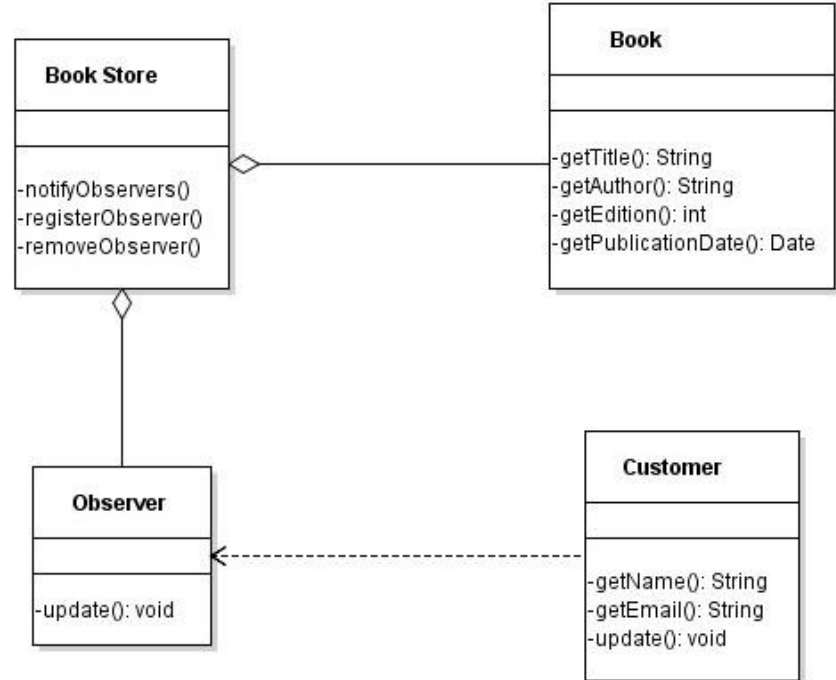
MVC: View

- The View is the user interface of the app
- We used HTML, PHP, and CSS to display our website to users
- Updated by the Controller so if a book is uploaded or deleted the View will display the correct info
- This code displays the featured books on the homepage - it relies on the Controller to pull titles and images from the Model

```
[Preview] README.md index.php x
views > pages > index.php
81 <!-- Featured books -->
82 <section id="product1">
83 <div class="container my-5">
84   <h1 class="text-center">Featured Textbooks</h1>
85   <hr>
86   <div class="row">
87     <?php foreach ($r2 as $row) { ?>
88       <div class="col-md-4">
89         <div class="card">
90           <?php if (isset($row["sellimage"])) { ?>
91             " height="350" />
92           <?php } else { ?>
93             
94           <?php } ?>
95           <div class="card-body">
96             <h5 class="card-title"><?php echo $row["selltitle"] ?? '' ?></h5>
97             <h4><?php echo $row["price"] ?? '' ?></h4>
98             <a href="#" class="btn btn-primary" onclick="window.location.href='learnmore.php?id=<?php echo $row["sellid"] ?>" id="learnmore">Learn More</a>
99             <div class="d-flex justify-content-between">
100               <div class="seller-info">
101                 " height=40 width=40 />
102                 <p class="seller-name"><?php echo $row["username"]; ?></p>
103               </div></div>
104             </div>
105           </div>
106         </div>
107       <?php } ?>
108     </div>
109   </div>
110 </section>
111
```

Design Pattern: Observer

- The observer design pattern is focused on the interactivity of the site
- User's are assigned "Observers" which notify them of state changes
 - For sellers, an Observer notifies them if a user has added their book to the user's favourites list
 - For buyers, an Observer notifies them if a book in their favourites has been sold or if a book they want has become available



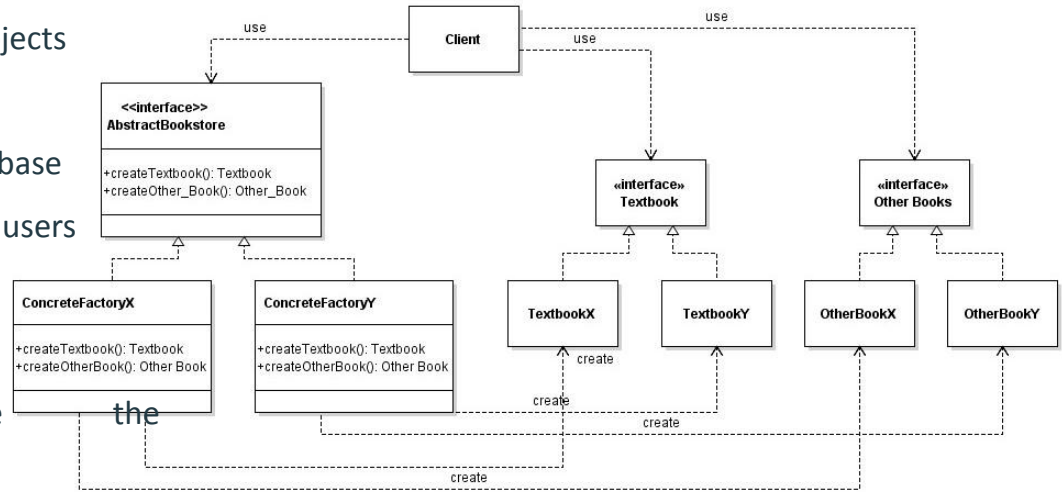
Design Pattern: Observer Cont.

- Implementation of this design pattern was limited due to complexity and time constraints
- At the moment, user's can add books to a favourites list or upload new posts but they will not be notified if any state change occurs
- The code shown pulls the user's favourite books from the database to be displayed. Ideally, an Observer would be here to occasionally check for changes and notify the user if there is

```
[Preview] README.md  myaccount.php X
Back-End > myaccount.php
191
192 <div class="favorites-section">
193   <h2>Favorite Books</h2>
194   <?php
195
196   // code to retrieve favorites goes here
197   if (count($row) > 0) {
198     echo '<ul class="favorites-list">';
199     foreach ($row as $r) {
200       echo '<li class="favorites-item">';
201       echo '' . $r['selltitle'] . '</h3>';
203       echo '</li>';
204     }
205     echo '</ul>';
206   } else {
207     echo "<p>You haven't added any favorites yet.</p>";
208   }
209   ?>
210 </div>
211 <div class="favorites-section">
212   <h2>Posted Books</h2>
213   <p>P.S: You can click on post to delete them.
214   <?php
215
216   // code to retrieve favorites goes here
217   if (count($rows) > 0) {
218     echo '<ul class="favorites-list">';
219     foreach ($rows as $row) {
220       echo '<li class="favorites-item">';
221       echo '' . $row['selltitle'] . '</h3>';
223       echo '<form method="POST">';
224       echo '</li>';
225     }
226     echo '</ul>';
227   } else {
228     echo "<p>You haven't added any posts yet.</p>";
229   }
230   ?>
231
232
233 </div>
```

Design Pattern: Factory

- Makes creating or deleting book objects simpler and keeps code organised
- Similar for adding users to the database
- A book can be uploaded by normal users or by administrators - then the create function is called and a new book object is created with info like the title, author, and ISBN
- The client is only able to see the interfaces for the bookstore and books



Design Pattern: Factory Cont.

- Sellers can be thought of as the factories creating new books by making new posts
- Buyers only interact with the View which uses the Controller to display the bookstore website and books
- The code shown takes user-entered info to make a new post and saves it in the database - this can be thought of as the create function in the factory

```
[Preview] README.md myaccount.php X
views > pages > myaccount.php
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

if (isset($_POST["Createsell"]) && $_POST["Createsell"])
{
    $selltitle = trim($_POST["book-title"]);
    $sellauthor = trim($_POST["book-author"]);
    $sellisbn = trim($_POST["book-isbn"]);
    $sellclassnum = trim($_POST["book-class"]);
    $selldescription = trim($_POST["post-content"]);
    $price = trim($_POST["book-price"]);
    $currenttime=time();
    try {

        if($selltitle == null || $selltitle == "" || $selltitle == false) {
            $validate = false;
            $error .= "Book Title Is empty.\n<br />";
        }

        if($validate == true) {
            $created_dt=date("Y-m-d H:i:s",$currenttime);
            $q2 = "INSERT INTO sellcreation (selltitle, sellauthor, sellisbn, sellimage, selldescription, created
            VALUES ('$selltitle', '$sellauthor', '$sellisbn', '$target_file', '$selldescription', '$created_dt',

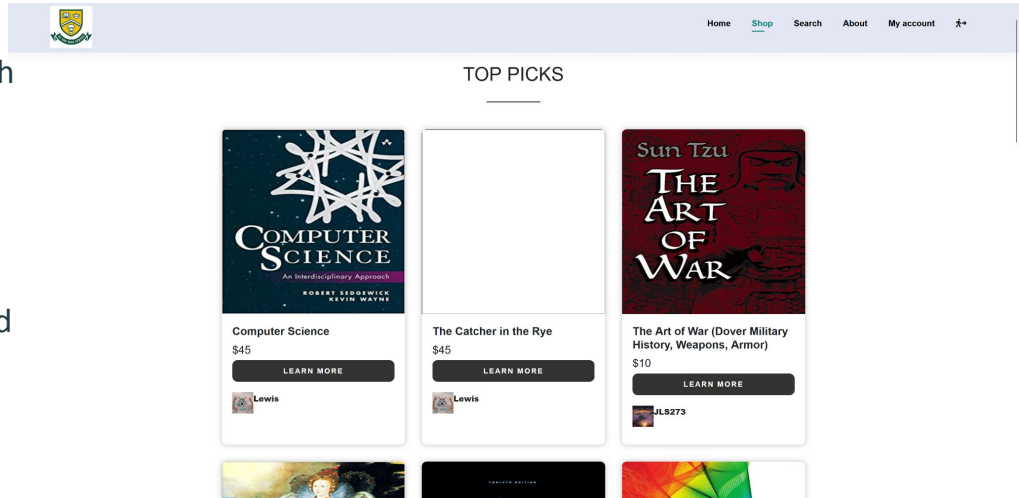
            $r2 = $db->exec($q2);

            if ($r2 != false) {
                header("Location: myaccount.php");

                $r2 = null;
                $db = null;
                exit();
            } else {
                $r2 = null;
                $validate = false;
                $error .= "Trouble adding product to database!\n<br />";
            }
        }
    }
}
```

Software Behaviour

- The site functions in a similar way to Kijiji or Facebook marketplace
- Sellers upload their own books to sell and buyers browse books and message sellers directly
- Transactions do not take place on our website, instead users are expected to work out payment methods and book exchanges themselves after connecting through our site
- The site allows users to search, view top picks, view a specific book, add a book to favourites, and upload





Thanks!