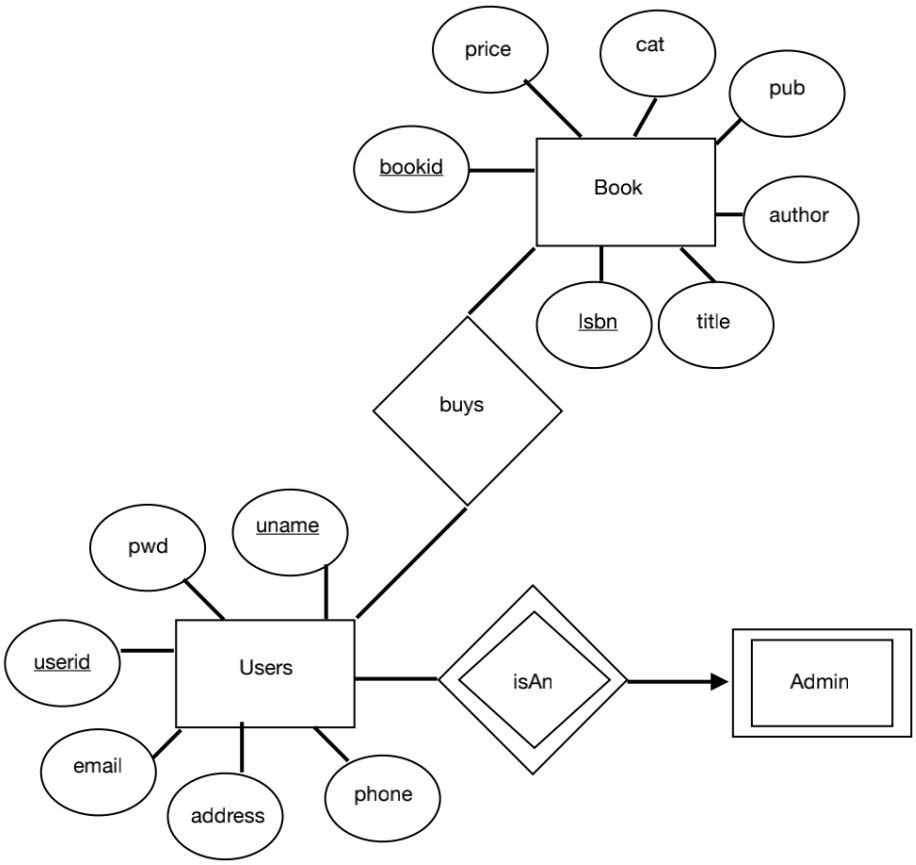
91815855E109 奇迹

ONLINE BOOK STORE
J2EE

Platform used : J2EE
IDE used: Intelli J Ultimate developed by Google https://www.jetbrains.com/
Databse provider used: MySql Workbench
Server used: Apache Tomcat 9.0.37 https://tomcat.apache.org/
Languages types used: Java,Html 5,Css 3,Java Server Page(JSP),Java Script,SQL.



ER - DIAGRAM

Architecture

INTRODUCTION

This project took off by creating a MySQL Database called **Online Book Store**.This database initially included a users and books table as required per the instructor.

I was to add at least 10 books and 5 providers(publishers) etc. into my database, of which I took the liberty of adding more than that.

NOTE: MOST OF THESE BOOKS WHERE ADDED USING THE WEBSITE THROUGH THE ADD BOOK FORM THAT WOULD BE DISCUSSED LATER IN THIS REPORT.

	bookid	isbn	title	author	pub	cat	price
1	1	974-7-119-10061-8	Prof. Jasp	Hawkins	WanerBros	History	150
2	2	340-2-003-45061-6	Computer Science Overview	Glenn BrookShear	Post & Telecom Press	Science	200
3	3	963-2-503-2382-3	Understanding Worship	David Oyedepo	Dominion Publishing House	Religious	359
4	4	583-4-819-352-8	Learning J2EE	张博士	南京理工大学	Software	499
5	5	743-5-178-606-3	Calculus	Mr.Huang .M	南京理工大学	Mathematics	315
6	6	514-8-137-74839-6	When we First met	Amber.Rose	Disney	Romance	203
7	7	654-3-190-0698-5	Lets go back in Time	Authur Benjamin	Universals	Adventure	58
8	8	549-6-132-08478-1	The Shakseparean Truth	William Southend	Visionaries	Literature	874
9	9	864-0-275-42829-3	What if it Could be Destroyed?	Min He	The Physicists	Science	387
10	10	321-5-312-9123-5	Agriculture	Miracle Lee	Nations Treasure	Science	100
11	11	203-4-693-8169-4	Organic-Chemistry	Mr Yuhe	Oxford	Science	489
12	12	507-1-420-7491-0	Geography	Gabriel Barthalomew	Harvad	Science	234
13	13	117-3-283-3591-0	JNDI	Noah k.	Port Publisher	java	209
14	14	155-0-684-8302-3	Java Beans	Dr. Himel .I	Asia Inc	java	521
15	15	143-2-859-3102-2	Servlets	Dr. David .N	Oxford	java	21
16	16	204-5-569-7502-9	And it was named JavaScript	Dr.Julia .W	Realm	java	341
17	17	314-7-081-89415-3	Pick a Side	Dr.Smith .x	Democrats	politics	341
18	18	405-2-185-128-2	Black Lives Matter	Mr.John Doe	The Movement	politics	10
19	19	229-15-585-1-2340	Nigerian Politics	Mr.Victor Olanyinka	Nigeria Eagles	politics	156
20	20	000-13-542-4-2210	Bolivia	Mr.Makrov Damme	The Status	politics	309
21	21	134-2-345-23563-0	Bible	Jesus	HolySpirit	Religious	0
22	22	548-6-209-11339-4	China	Dr.Xiao Li	Chinese Press	History	3890
23	23	526-27398-2-1-980	Miracle	Miracle	Pierson	History	283

Book Table (pub means publishers or providers and cat means Category)

	userid	uname	pwd	email	address	phone
1	1	scott	tiger	scott@hotmail.com	12-2-23,Dwarakanagar,VSP-16	3434343
2	2	mike	tyson	tyson@hotmail.com	12-5-23,MVP Colony,VSP-43	2234343
3	3	admin	admin	admin@hotmail.com	12-5-23,MVP Colony,VSP-43	2234343

Users Table

From Database to Login page

The next part of this project was to create a User class for a user to be able to log into his/her account.This was done using java programming language. I started this by creating a connection to the database.

Firstly, I created the Getters and Setters of my variables which represented the columns of my database and also a Boolean variable to check if the user log in was successful.

After that, I then implemented a get connection function. Inside this get connection function, I instantiated a get connection object from **java.Sql** then I added my driver class **com.mysql.cj.jdbc.Driver**. Next I assigned my connection instance equals to the my driver's get connection function which required my database url and its password.In the end I returned the connection because the get function I was creating was of type **Connection**.

```java
public Connection getConnection()  {
    Connection con = null;
    try {

        Class.forName("com.mysql.cj.jdbc.Driver");
          Driver driver = com.mysql.cj.jdbc.Driver;
    } catch (ClassNotFoundException e) {
        System.out.print(e.getMessage());
    }
    try {
        con = DriverManager.getConnection( url: "jdbc:mysql://localhost:3306/OnlineBookStore",  user: "root",  password: "root");
    } catch (SQLException ex) {
        System.out.print(ex.getMessage());
    }
    return con;
}
```

With this I could move to the log in function.This part was a little bit tricky because it required me to use sql query language.I had to read and do some research for the documentation of what I was about to do.

The log in function returned a boolean value.I created a new instance of Connection and a new instance **Prepared Statement**.The Connection instance was used to store my get connection function I created earlier and the Prepared Statement was used to hold mysql query gotten from my connection I created earlier. For the sql query, I told the program to get a row of values where the user name and password gotten from the log in form in my **index.html** existed then I checked if the values where true by passing the **Prepared Statement - execute query** function into my a **Result Set instance object**. Then I issued a **Result Set-next function** to check if the values existed as the Result set-next function **points at before the row then moves to the next row which is an actual row**. If this was true then it would return my logged in variable which I initially created as true or if not false.

Note: All this was done with a try and catch exception.

```java
public boolean login() {
    Connection con = null;
    PreparedStatement prep = null;
    try {
        con = getConnection();
        prep = con.prepareStatement( sql: "Select userid,email,address,phone from users where uname = ? and pwd = ? ");
        prep.setString( parameterIndex: 1, userName);
        prep.setString( parameterIndex: 2, password);

        ResultSet rs = prep.executeQuery();

        if (rs.next()) {
            userId = rs.getInt( columnIndex: 1);
            email = rs.getString( columnIndex: 2);
            address = rs.getString( columnIndex: 3);
            phone = rs.getString( columnIndex: 4);
            loggedIn = true;
        } else {
            loggedIn = false;
            clean(con, prep);
        }
    } catch (Exception e) {
        loggedIn = false;
        System.out.print(e.getMessage());

    }
    return loggedIn;
}
```

Log in function

The clean function below was implemented to close all java.sql classes and interfaces

```java
public void clean(Connection con, PreparedStatement ps) {
    try {
        if (ps != null) ps.close();
        if (con != null) con.close();
    } catch (Exception ex) {
        System.out.println(ex.getMessage());
    }
}
```

Clean connection Function

With all this said and done, an **index.jsp** page was made to help me redirect users either to the error page when he/she logs in incorrectly or the home page when the user does otherwise.

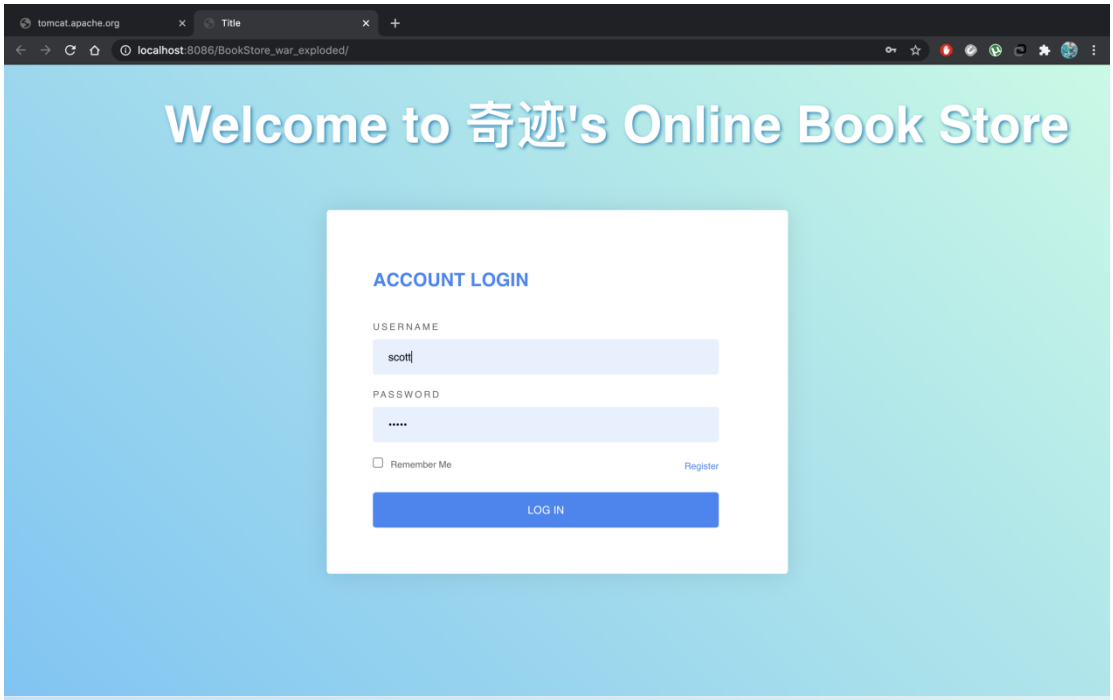
```
<%@ page contentType="text/html;charset=UTF-8" %>
<jsp:useBean id="user" class="BS.User" scope="session"/>
<jsp:setProperty name="user" property="*"/>

<html>
<head>
    <title>Login</title>
</head>
<body>
<%

    user.login();
    if (user.isLoggedIn()) {
        response.sendRedirect("home.jsp");
    } else {
        response.sendRedirect("error.html");
    }
%>
</body>
</html>
```

index.jsp

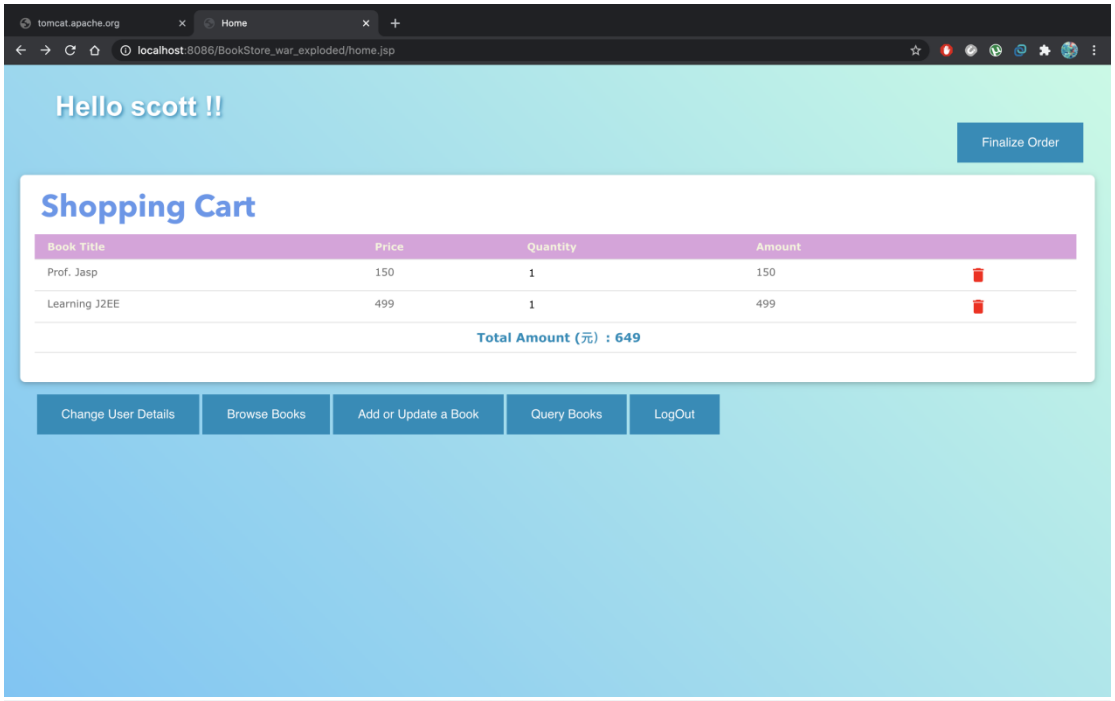
Also with the help of **Html and Css,** I was able to setup the log in form which had the input name the same as the as that of the variables I created which would enable me to pass the information to from my log in page to sql for inspection if the user log in details where true or false.

```html
<div class="form">
    <div class="form-toggle"></div>
    <div class="form-panel one">
        <div class="form-header">
            <h1>Account Login</h1>
        </div>
        <div class="form-content">
            <form action="index.jsp" method="post">
                <div class="form-group">
                    <label for="username">Username</label>
                    <input type="text" id="username" name="userName" required="required"/>
                </div>
                <div class="form-group">
                    <label for="password">Password</label>
                    <input type="password" id="password" name="password" required="required"/>
                </div>
                <div class="form-group">
                    <label class="form-remember">
                        <input type="checkbox"/>Remember Me
                    </label><a class="form-recovery" href="register.html">Register</a>
                </div>
                <div class="form-group">
                    <button type="submit">Log In</button>
                </div>

            </form>
```

The Log in form index.html (HTML 5 and Css 3)



User Scott wants to log in



After User Scott Logs In

REGISTRATION PAGE: MY FRIST INSERT FUNCTION

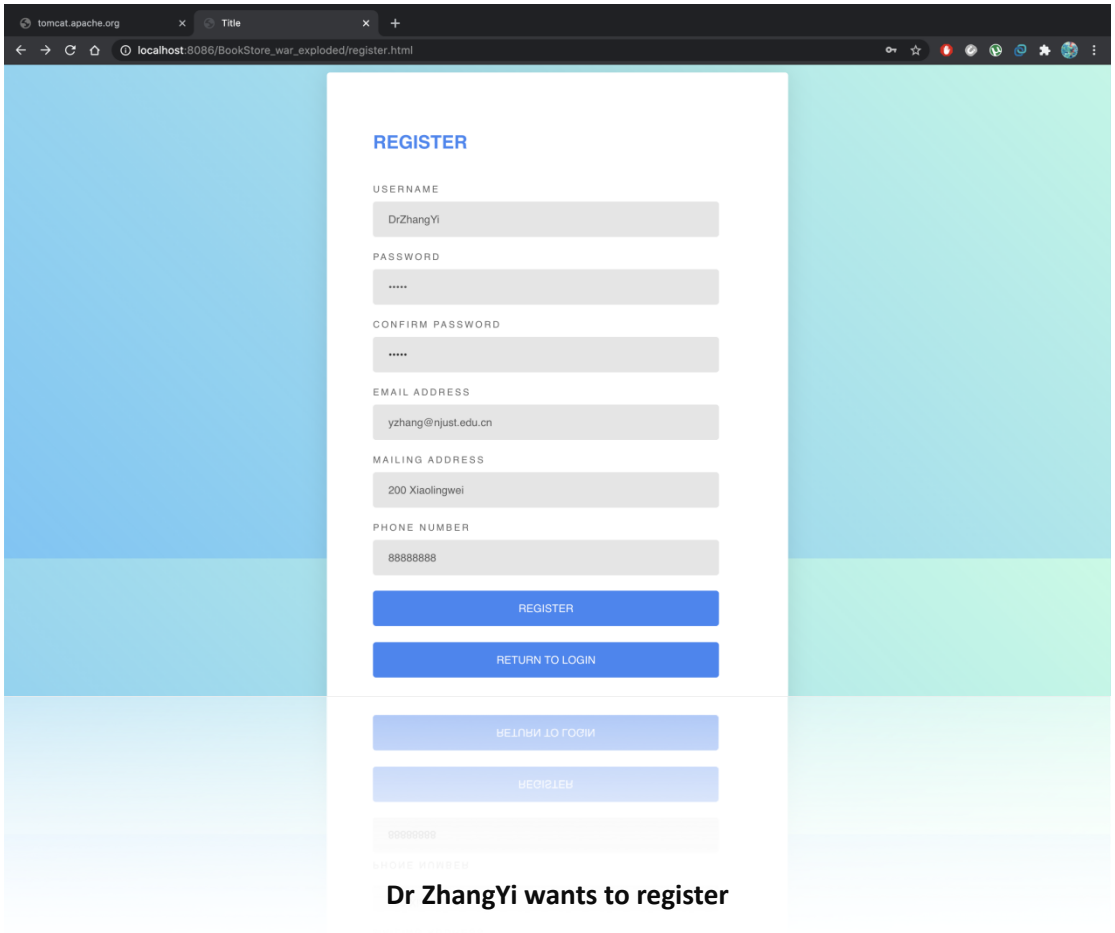
The Registration page basically has the same design and most part of its code like same as the log in but the difference was in the sql query and also I had to create an **auto - increment function** for user id: the name of this function was called getNextUserId which returned an integer.

```
userId = getNextUserid(con);
ps = con.prepareStatement( sql: "insert into users values (?,?,?,?,?,?)");
ps.setInt( parameterIndex: 1, userId);
ps.setString( parameterIndex: 2, userName);
ps.setString( parameterIndex: 3, password);
ps.setString( parameterIndex: 4, email);
ps.setString( parameterIndex: 5, address);
ps.setString( parameterIndex: 6, phone);
ps.executeUpdate();
loggedIn = true;
```

The Crucial aspect of registration function

```
public int getNextUserid(Connection con) throws Exception {
    PreparedStatement ps = null;
    ps = con.prepareStatement( sql: "select max(userid) + 1 from users");
    ResultSet rs = ps.executeQuery();
    rs.next();
    int nextuserid = rs.getInt( columnIndex: 1);
    rs.close();

    return nextuserid;

}
```

The AUTO INCREMENT ID function



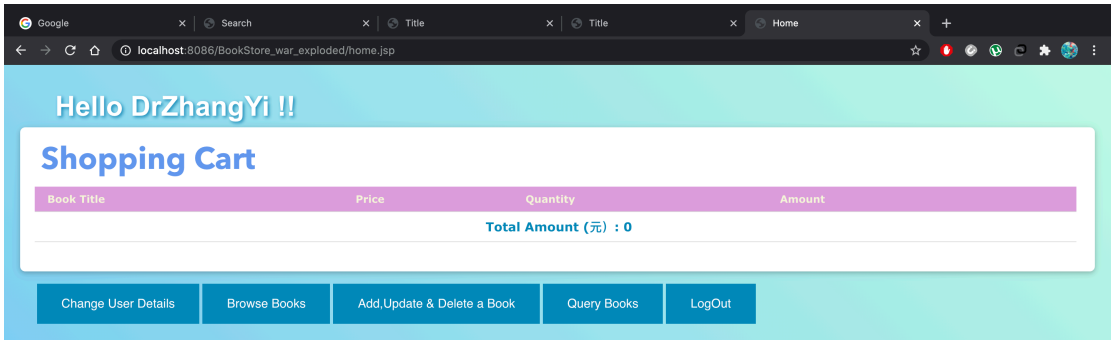
Dr ZhangYi wants to register

4 rows | Tx: Auto | DDL | DML | Comma-...d

<Filter Criteria>

	userid	uname	pwd	email	address	phone
1	1	scott	tiger	scott@hotmail.com	12-2-23,Dwarakanagar,VSP-16	3434343
2	2	mike	tyson	tyson@hotmail.com	12-5-23,MVP Colony,VSP-43	2234343
3	3	admin	admin	admin@hotmail.com	12-5-23,MVP Colony,VSP-43	2234343
4	4	DrZhangYi	zhang	yzhang@njust.edu.cn	12-2-23,Dwarakanagar,VSP-16	88888888

MySql Output



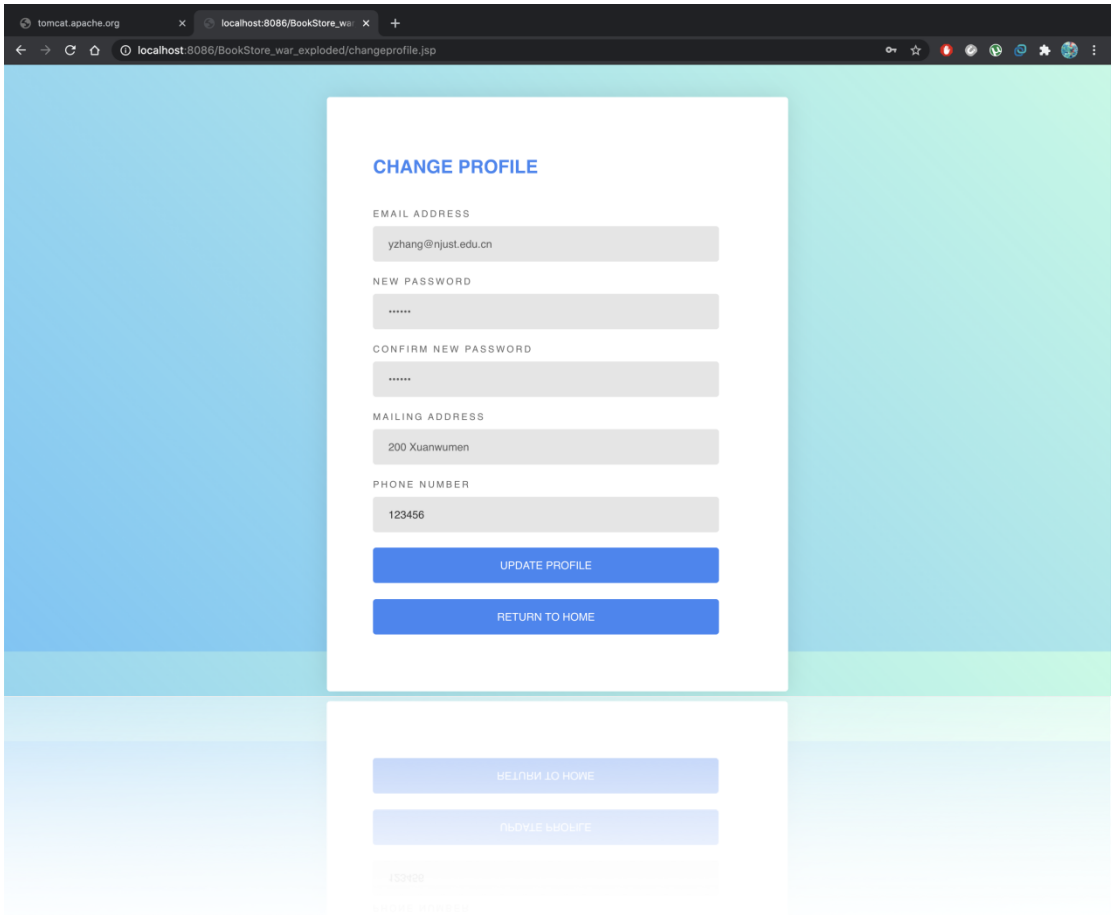
Server Output: The Home Page

The Change User Profile Page: My First Update function

This page required me to create my first update function.This function was meant to get a parameter of type String named new password: the reason will be revealed later. Just like before new connection and prepared statement objects were made for getting the sql query.The String new password I mentioned earlier would be passed to check if the user decides to create a new password or not. If the user did, a **concatenation of string** would be made to inform the sql query of this decision or otherwise.

```
con = getConnection();
String cmd = "update users set email=?, phone  = ? , address =? ";

if (! newpwd.equals(""))
    cmd += " , pwd = '" + newpwd + "'";

cmd = cmd + " where userid  = ?";

ps = con.prepareStatement(cmd);
ps.setString( parameterIndex: 1,email);
ps.setString( parameterIndex: 2,phone);
ps.setString( parameterIndex: 3,address);
ps.setInt( parameterIndex: 4,userId);

int cnt = ps.executeUpdate();
if ( cnt==1 )
{
    if ( ! newpwd.equals("") )
        password = newpwd;
    return null;
}
else
    return "Invalid User. Unable to update profile.";
```

Crucial Part of the of the Update Profile Connection



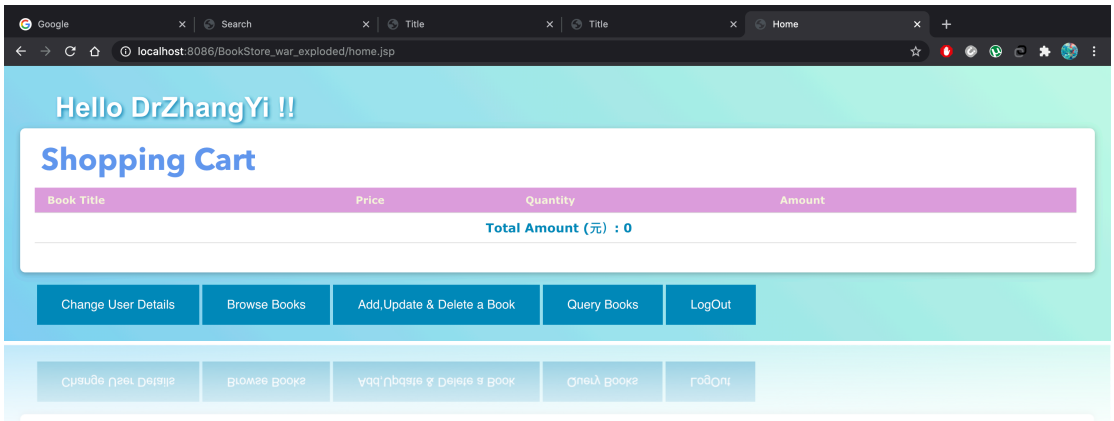
Dr.Zhang wants to change his password and address

<Filter Criteria>

	userid	uname	pwd	email	address	phone
1	1	scott	tiger	scott@hotmail.com	12-2-23,Dwarakanagar,VSP-16	3434343
2	2	mike	tyson	tyson@hotmail.com	12-5-23,MVP Colony,VSP-43	2234343
3	3	admin	admin	admin@hotmail.com	12-5-23,MVP Colony,VSP-43	2234343
4	4	DrZhangYi	yzhang	yzhang@njust.edu.cn	200 Xuanwumen	88888888

MySQL Output

BOOKS SECTION



Home Page

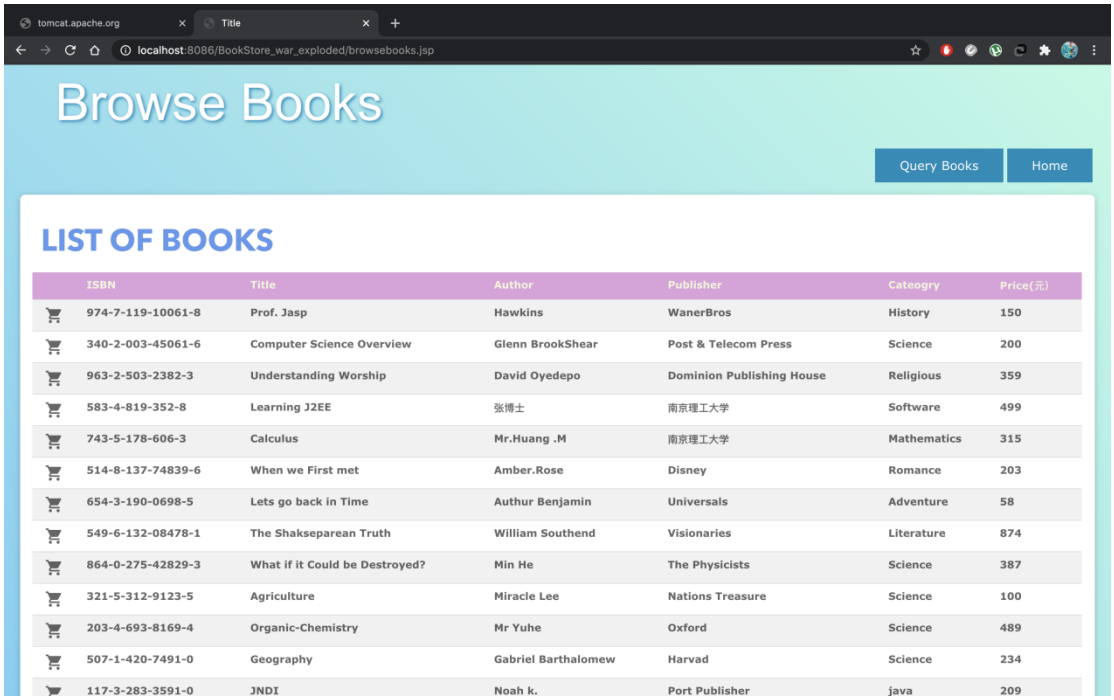
If you take a look at the home page above you will see a set of buttons that are reserved for certain functions.Besides the Change user details button already visited, all other buttons will be visited to see their various functions.This will also include the Shopping Cart Section

The Browse Book Button:

The function of this button is meant for revealing all books.I started this by creating a html table in my java server page called **listbooks.jsp** to display the required columns.
Up next was the data it self. This was done by executing a **Statement** instance of my sql query and passing it into my Result Set . For me to be able to display my data, I implemented a while loop of which I got each column values of my result set using their index position. The condition for my while was the result set-next function.Finally I closed the java.sql classes and designed it using css.

```
<table class=" w3-table w3-striped w3-bordered" >
    <tr style="...">
        <th>
        <th>ISBN
        <th>Title
        <th>Author
        <th>Publisher
        <th>Cateogry
        <th>Price(元)
    </tr>
    <%

        Connection con = user.getConnection();
        Statement st = con.createStatement();

        ResultSet rs = st.executeQuery("select isbn,title,author,pub,cat,price from books " );

        while (rs.next()) {

    %>
    <tr>
        <td style="...">
            <a href=addbook.jsp?isbn=<%=rs.getString("isbn")%>><i class="material-icons" >shopping_cart</i></a>
        <td style="..."><%=rs.getString(1)%>
        <td style="..."><%=rs.getString(2)%>
        <td style="..."><%=rs.getString(3)%>
        <td style="..."><%=rs.getString(4)%>
        <td style="..."><%=rs.getString(5)%>
        <td style="..."><%=rs.getString(6)%>

    </tr>

    <%
        }
        rs.close();
        st.close();
        con.close();
```

The Crucial Part of the Browse books code.



ISBN	Title	Author	Publisher	Cateogry	Price(元)
974-7-119-10061-8	Prof. Jasp	Hawkins	WanerBros	History	150
340-2-003-45061-6	Computer Science Overview	Glenn BrookShear	Post & Telecom Press	Science	200
963-2-503-2382-3	Understanding Worship	David Oyedepo	Dominion Publishing House	Religious	359
583-4-819-352-8	Learning J2EE	张博士	南京理工大学	Software	499
743-5-178-606-3	Calculus	Mr.Huang .M	南京理工大学	Mathematics	315
514-8-137-74839-6	When we First met	Amber.Rose	Disney	Romance	203
654-3-190-0698-5	Lets go back in Time	Authur Benjamin	Universals	Adventure	58
549-6-132-08478-1	The Shakseparean Truth	William Southend	Visionaries	Literature	874
864-0-275-42829-3	What if it Could be Destroyed?	Min He	The Physicists	Science	387
321-5-312-9123-5	Agriculture	Miracle Lee	Nations Treasure	Science	100
203-4-693-8169-4	Organic-Chemistry	Mr Yuhe	Oxford	Science	489
507-1-420-7491-0	Geography	Gabriel Barthalomew	Harvad	Science	234
117-3-283-3591-0	JNDI	Noah k.	Port Publisher	java	209

Browse book Page

NOTE: For brevity I took the liberty of not showing all the books...

ADD TO SHOPPING CART BUTTON: MY SECOND UPDATE FUNCTION

This update function's job was to add into the cart any book the user selects through the C**art icon.**With this in mind, I updated the cart in a different way (**Using Iterators:You could say it updates on run time)**. I first created an Item class for the cart item. This class included a constructor to get 3 parameters the isbn, title, price, and also a third variable which was quantity = 1; for later explanation. Then I created a getter for the isbn, title, price, quantity, and also a setter function and add function for quantity.All these were done for the later purpose of accessing them in my home.jsp page.

Next, I created a Cart class.This Cart Class entails an addItem function.The addItem function's purpose was to insert an item into the cart if it doesn't already exist and if the item did exist ,it would update its quantity count.

For this, I decided to create a find function of parameter String with variable name isbn.The purpose of this was to check if the item already existed in the cart and to return null if it did not (All this was done with the help of an **Iterator instance and an Array List instance**). By passing this find function into my addFunction, I checked for the existence of such an item then if it did exist it would call my add quantity function which I created earlier in my Item class and if not a new connection was made for an sql function to display the new set of items function in the cart.

I also created a function of type array list called getItems that returned items from the array list that I initially created as shown below.

```java
ArrayList items = new ArrayList();

public Item find(String isbn) {
    Iterator itr = items.iterator();
    Item itm;
    while (itr.hasNext()) {
        itm = (Item) itr.next();
        if (itm.getIsbn().equals(isbn)) {
            return itm;
        }
    }  // end of while

    return null;
}
```

The find function

```java
public void addItem(String isbn) {
    //check whether isbn is already present
    Item item = find(isbn);
    if (item != null)
        item.addQty(1);
    else {
        // get detais from Books tables

        Connection con = null;
        BS.User user = new BS.User();
        try {
            con = user.getConnection();
            PreparedStatement ps = con.prepareStatement( sql: "select title,price from books where isbn = ? ");
            ps.setString( parameterIndex: 1, isbn);

            ResultSet rs = ps.executeQuery();
            if (rs.next()) {
                item = new Item(isbn, rs.getString( columnIndex: 1), rs.getInt( columnIndex: 2));
                items.add(item);
            }
```

The Crucial Part of the Add Item function
You can see the getters created earlier form the Item class being used.

In my home.jsp, I created an instance of my item class then also one for my array list instance and an iterator instance.I also assigned my array list to my cart items and then passed it into my Iterator instance. I addition to that, I created a variable of type integer named it total so I could be able to get the total price of the items in the cart.

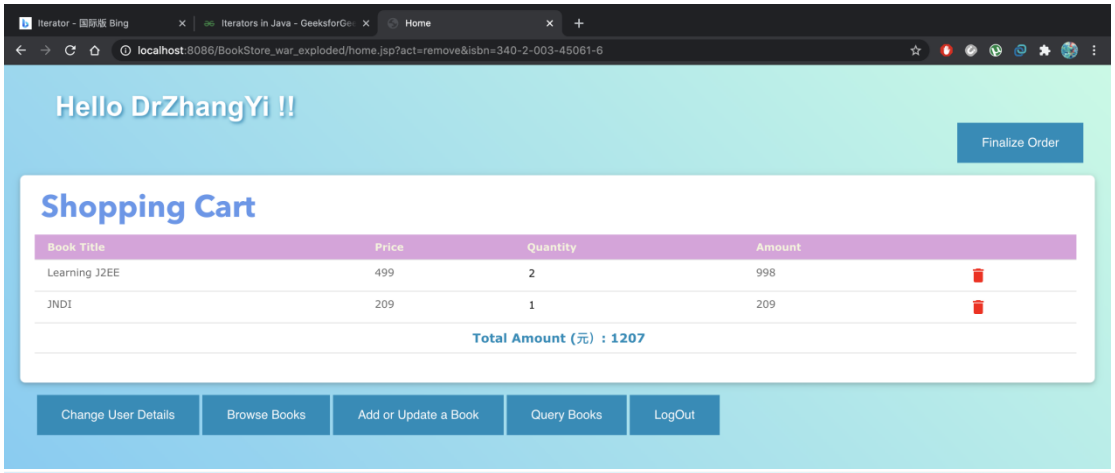
Using a while loop I was able to get the total items which required a little calculation from me. The while loop had a condition of type iterator **hasNex**t function for me to check the whole list. Then finally I designed the outputs.The deletion or removing item aspect of the cart was done with the help of array list **remove** function.This was aided with the help of the set up I made in my html.

```
BS.Item item;

ArrayList items = cart.getItems();
Iterator itr = items.iterator();
int total = 0;

while (itr.hasNext()) {
    item = (Item) itr.next();
    total += item.getQty() * item.getPrice();


r>
  <td>
      <input type=hidden value=<%=item.getIsbn()%> name=isbn>
          <%= item.getTitle()%>
  <td align="right"><%= item.getPrice()%>
  <td align="center"><input style="..." type=text name=qty size=5 value=<%=item.getQty()%>>
  <td align="right"><%=item.getQty() * item.getPrice()%>
  <td align="center"><a
          href=home.jsp?act=remove&isbn=<%=item.getIsbn()%>><i style="..." class="material-icons" >delete</i></a>
tr>


  }

r>
  <td colspan=5 style="...">
      Total Amount (元) : <%=total%>
```

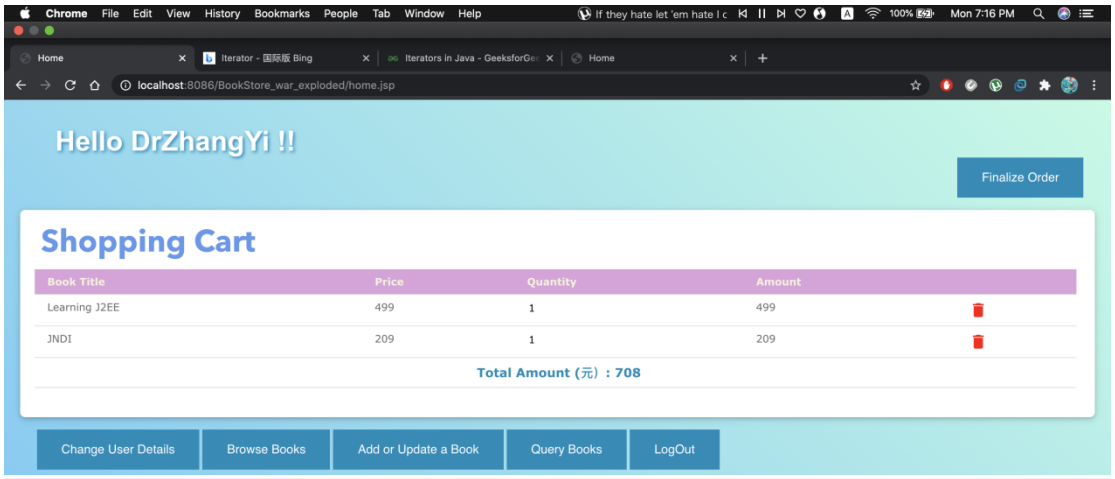
The Crucial Part of the home.jsp

```
public void removeItem(String isbn) {
    Item item = find(isbn);
    if (item != null)
        items.remove(item);
} // end or removeItem
```

The Remove function in Cart.java



Dr. Zhang decides to buy 2 books from J2EE and one on JNDI



Dr.Zhang decides to remove one of J2EE books he bought

The ADD ,UPDATE or DELETE BOOK BUTTON:

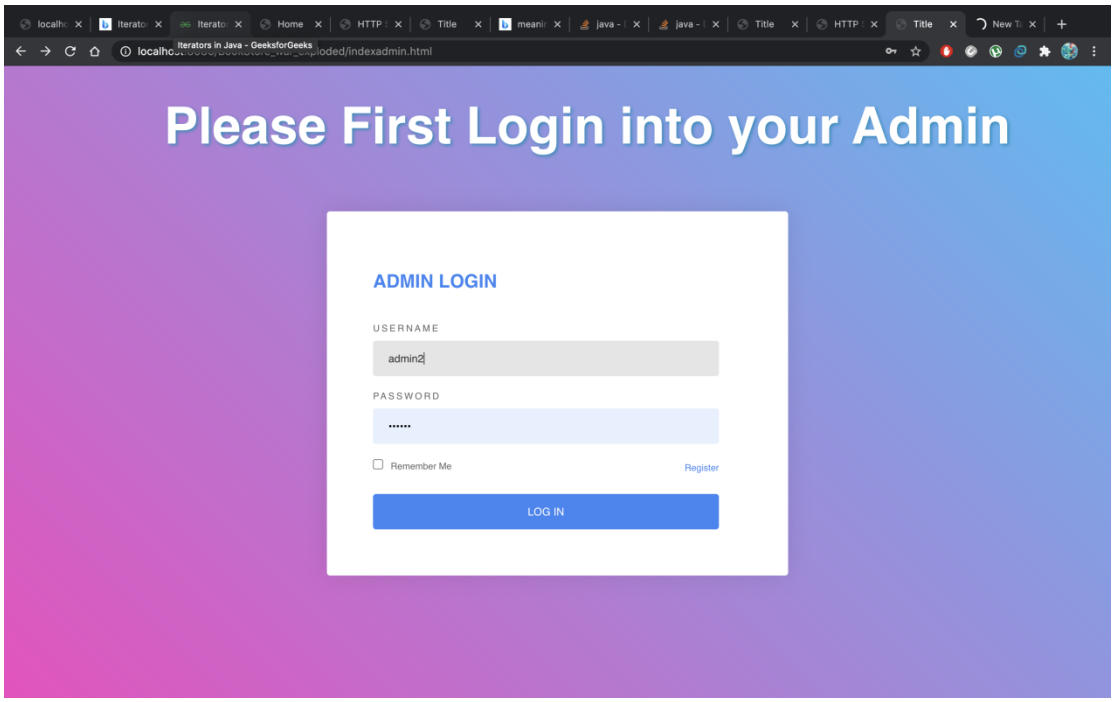
This button takes the user to the **Admin log in account**. Due to brevity, as required by the instructor,I will not be displaying the code for the admin section.But **note that the admin section required me to create a new sql** table called **admin**. After this comes the **choice page**. I will go through them one by one:

•ADD BOOK: MY SECOND INSERT FUNCTION

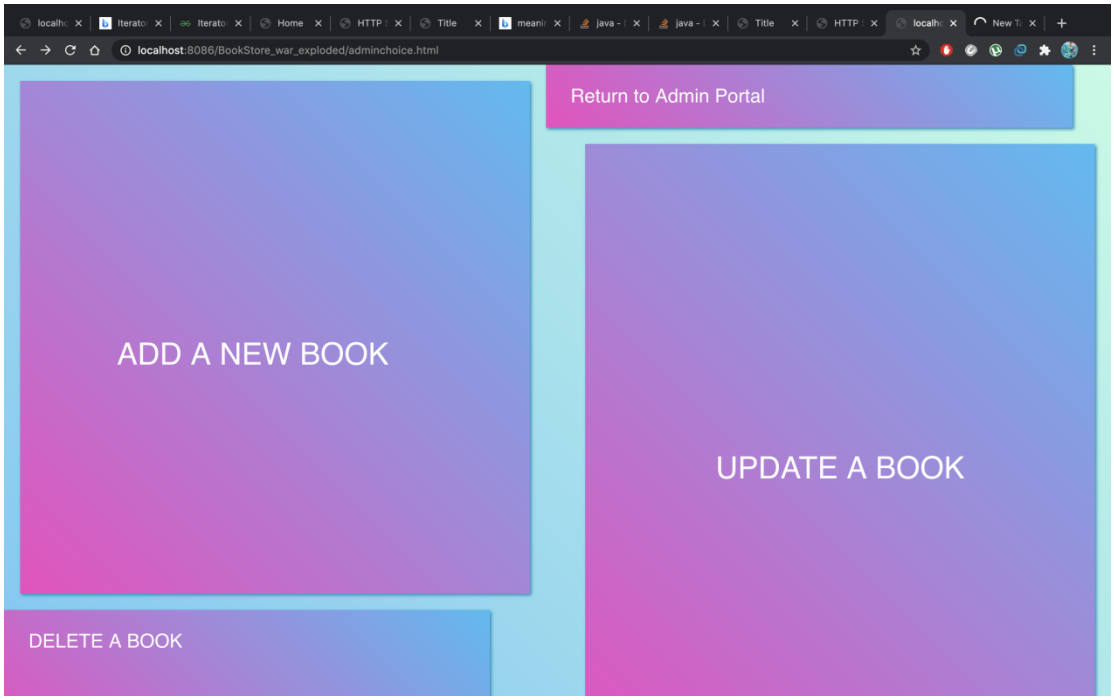
This is the part where the admin is required to add a book.I will not be explaining much of this part as it is the same procedure as the Registration section of this Online book store

```
public String addBook() {
    Connection con = null;
    PreparedStatement ps = null;

    try {
        con = getConnection();
        bookId = getNextBookid(con);
        ps = con.prepareStatement( sql: "insert into books values (?,?,?,?,?,?,?)");
        ps.setInt( parameterIndex: 1, bookId);
        ps.setString( parameterIndex: 2, isbn);
        ps.setString( parameterIndex: 3, title);
        ps.setString( parameterIndex: 4, author);
        ps.setString( parameterIndex: 5, pub);
        ps.setString( parameterIndex: 6, cat);
        ps.setInt( parameterIndex: 7, price);

        ps.executeUpdate();
```

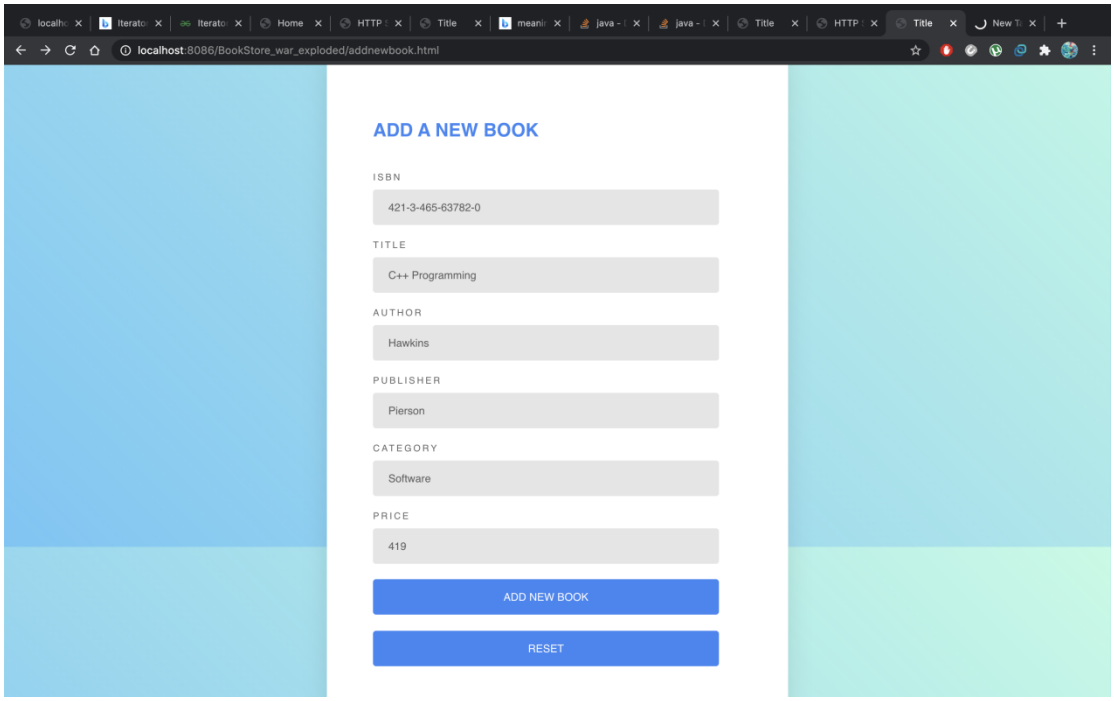
ADD BOOK FUNCTION



Admin Page



Choice Page



The Admin wants to add a new book

526-27398-2-1-980	Miracle	Miracle	Pierson	History	283
421-3-465-63782-0	C++ Programming	Hawkins	Pierson	Software	419

The Book is added

22	23	526-27398-2-1-980	Miracle	Miracle	Pierson	History	283
23	24	421-3-465-63782-0	C++ Programming	Hawkins	Pierson	Software	419

Sql Output

UPDATE BOOK BUTTON : MY THIRD UPDATE FUNCTION

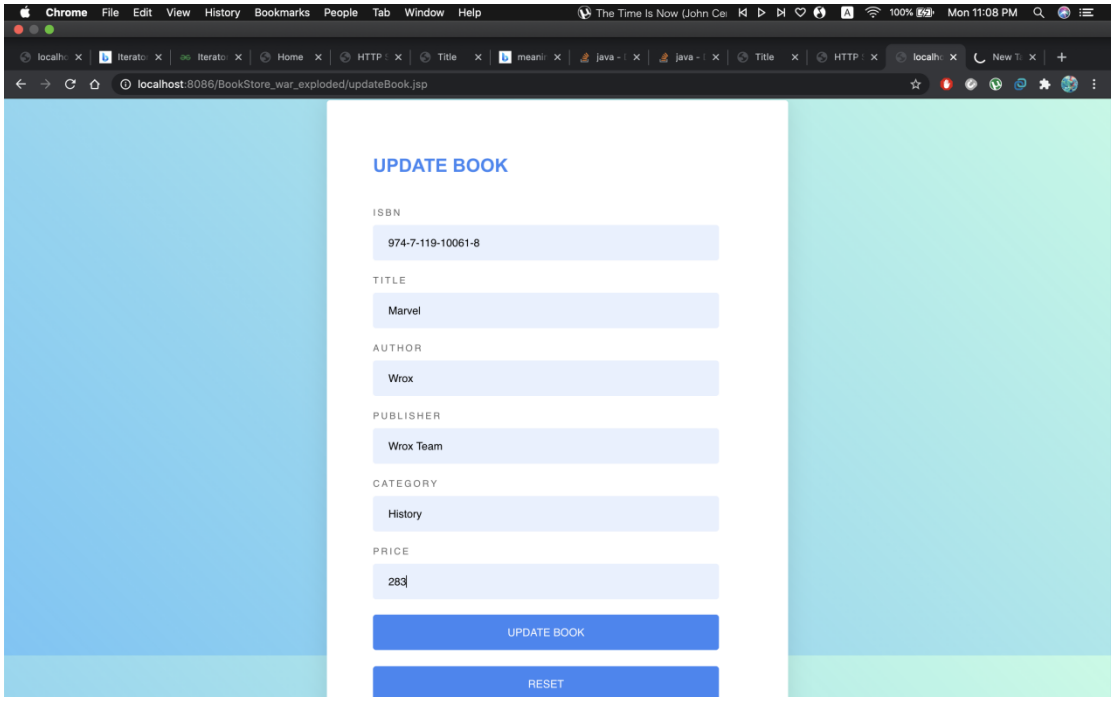
This was used to update books. Like before I will not be explaining this aspect because it is the same as that of change user profile . The Prepared statement assignment is of type integer that returns the row count of the sql query and or zero if it is null.

```
public void updateBook() {
    Connection con = null;
    PreparedStatement ps = null;

    try {
        con = getConnection();
        String cmd = "update books set title= ?,author =?,pub=?,cat=?,price=? where isbn = ?";
        ps = con.prepareStatement(cmd);
        ps.setString( parameterIndex: 1, title);
        ps.setString( parameterIndex: 2, author);
        ps.setString( parameterIndex: 3, pub);
        ps.setString( parameterIndex: 4, cat);
        ps.setInt( parameterIndex: 5, price);
        ps.setString( parameterIndex: 6, isbn);
        int i = ps.executeUpdate();
        if (i > 0) {
            System.out.print("Record Updated Successfully");
        } else {
            System.out.print("There is a problem in updating Record.");
        }
```

The Update Function

	bookid	isbn	title	author	pub	cat	price
1	1	974-7-119-10061-8	Prof. Jasp	Hawkins	WanerBros	History	150

Sql Before the Update



Update page

	bookid	isbn	title	author	pub	cat	price
1	1	974-7-119-10061-8	Marvel	Wrox	Wrox Team	History	283

Sql After the update

DELETE BOOK BUTTON FUNCTION

This basically runs like the update function but it returns a bool value andI also changed the sql query. This would be checked by delete.jsp and then sent to the deletebook.jsp.

```
public boolean deleteBook() {
    Connection con = null;
    PreparedStatement ps = null;
    try {
        con = getConnection();
        ps = con.prepareStatement( sql: "delete from books where isbn = ?");
        ps.setString( parameterIndex: 1,  isbn);
        int i = ps.executeUpdate();
        System.out.print(i);
        if (i > 0) {
        deleted = true;
        } else {
            deleted = false;
        }
    } catch (SQLException ex) {
       System.out.println(ex.getMessage());
    } catch (Exception exception) {
        exception.printStackTrace();
    }
    return deleted;
}
```

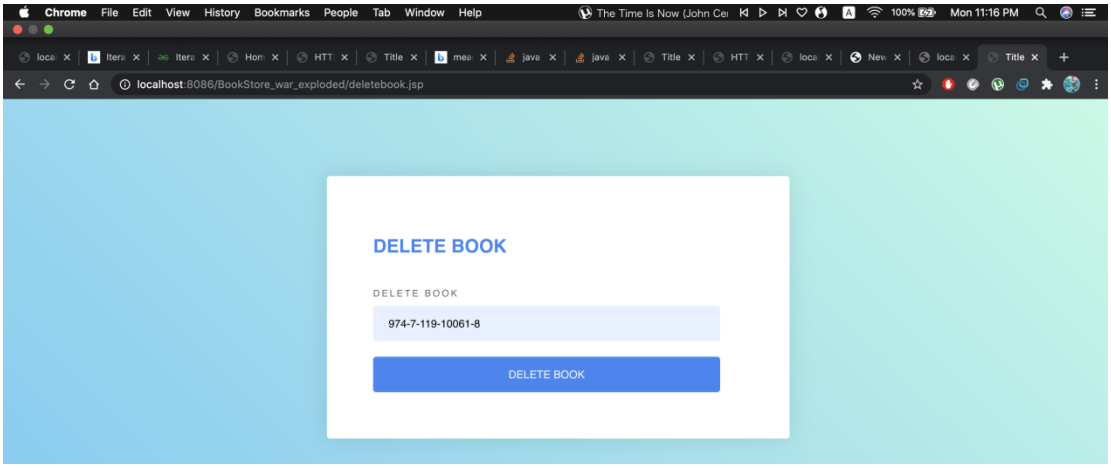
Delete Function

```
<%
    deletebook.deleteBook();
    if(deletebook.isDeleted())
    {
        response.sendRedirect("succes.html");
    }else{
        response.sendRedirect("error.html");
    }
```

delete.jsp

	bookid	isbn	title	author	pub	cat	price
1	1	974-7-119-10061-8	Marvel	Wrox	Wrox Team	History	283
2	2	340-2-003-45061-6	Computer Science Overview	Glenn BrookShear	Post & Telecom Press	Science	200

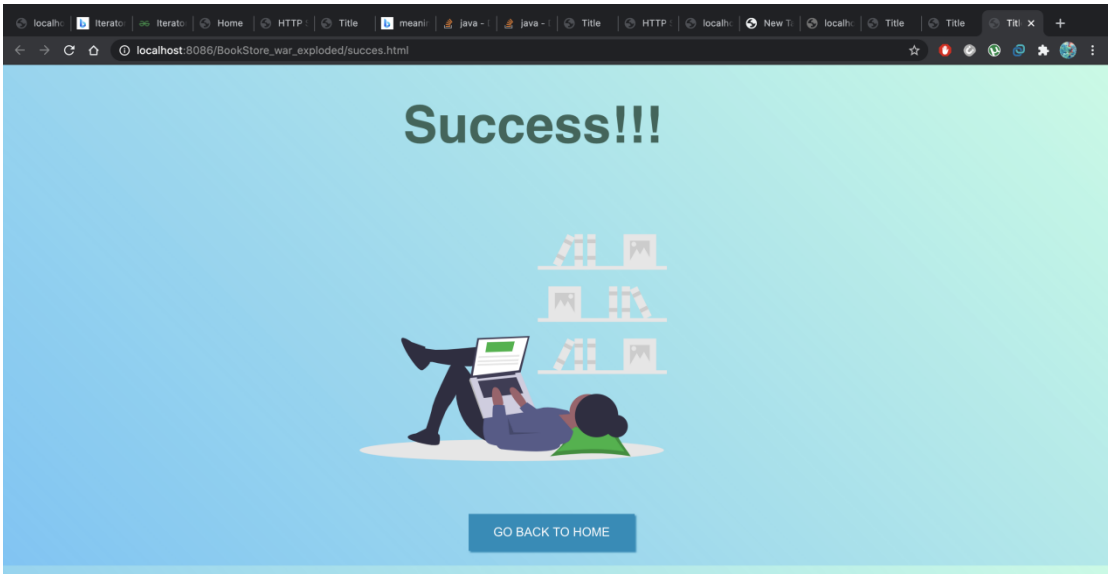
Sql before deletion



Deletebook.jsp

	bookid	isbn	title	author	pub	cat	price
1	2	340-2-003-45061-6	Computer Science Overview	Glenn BrookShear	Post & Telecom Press	Science	200
2	3	963-2-503-2382-3	Understanding Worship	David Oyedepo	Dominion Publishing House	Religious	359

After Deletion



Success page

- # QUERY/SEARCH BOOKS BUTTON

For this part, a form was created to search by title ,author, publisher, and also the range of price. This required me to setup a condition for mysql. It was done by first getting the required parameters via the **request.getParameter** function provided by the java server page.
After this for each one of them I setup an sql query with the "**like%**" key word of the sql query language. This was to check for related titles, authors, etc. from the users input.

Then for the price, because I was going to use a range function, I set up a **from price - to price** string variable , and compared them. For the from price, it was price greater than or equal to while for the to price it was less than or equal to. After this I added a concatenation for each search parameter which should be used if the user decides to input more than one search parameter.
All this was to take effect by including my **listbooks.jsp** into my **querybooks.jsp** file.
The reason for this was that the list books file was the one meant for displaying books so I decided to pass all the possible conditions into its where clause as a parameter called condition.

BookStore 〉 src 〉 main 〉 webapp 〉 JspScriptlet

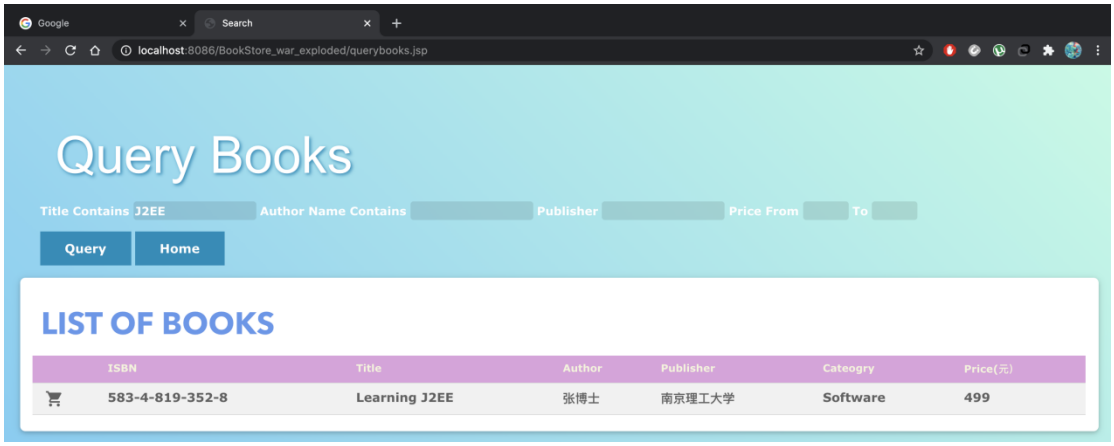
console | succes.html | User.java | querybooks.jsp | error.html | changeprofile.jsp

```
87  </div>
88
89  <%
90
91      String cond = "1=1";
92      String title = request.getParameter("title");
93
94      if (!title.equals(""))
95          cond = cond + " and title like '%" + title + "%'";
96
97
98      String author = request.getParameter("author");
99
100     if (!author.equals(""))
101         cond = cond + " and author like '%" + author + "%'";
102
103     String pub = request.getParameter("pub");
104
105     if (!pub.equals(""))
106         cond = cond + " and pub like '%" + pub + "%'";
107
108     String fromprice = request.getParameter("fromprice");
109
110     if (!fromprice.equals(""))
111         cond = cond + " and price >= " + fromprice;
112
113     String toprice = request.getParameter("toprice");
114
115     if (!toprice.equals(""))
116         cond = cond + " and price <= " + toprice;
117
118     request.setAttribute("cond", cond);
119
120 %>
121
122 <jsp:include page="listbooks.jsp" flush="true"/>
```

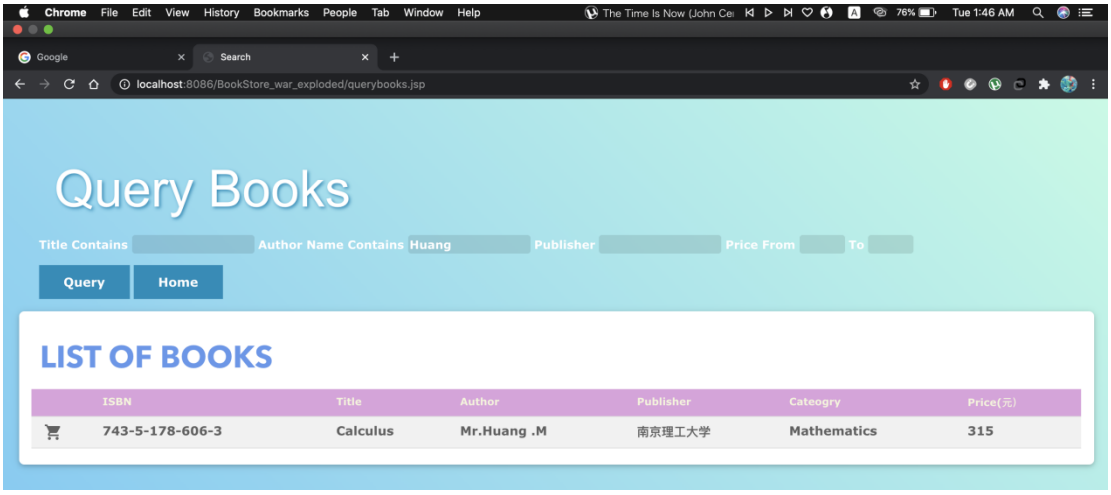
In total there are 4 search parameters

```
ResultSet rs = st.executeQuery("select isbn,title,author,pub,cat,price from books where  "+cond );
```

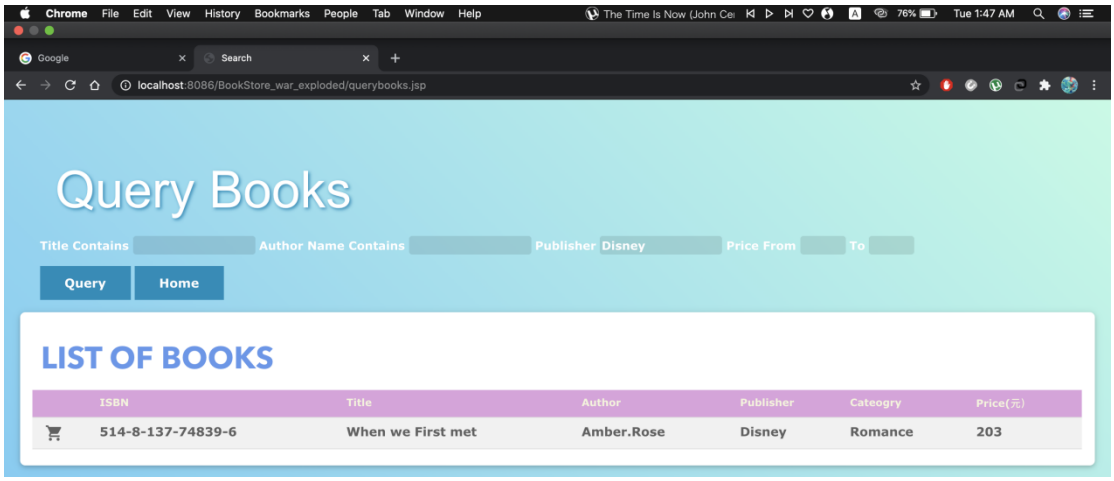
Crucial part from the listbooks.jsp



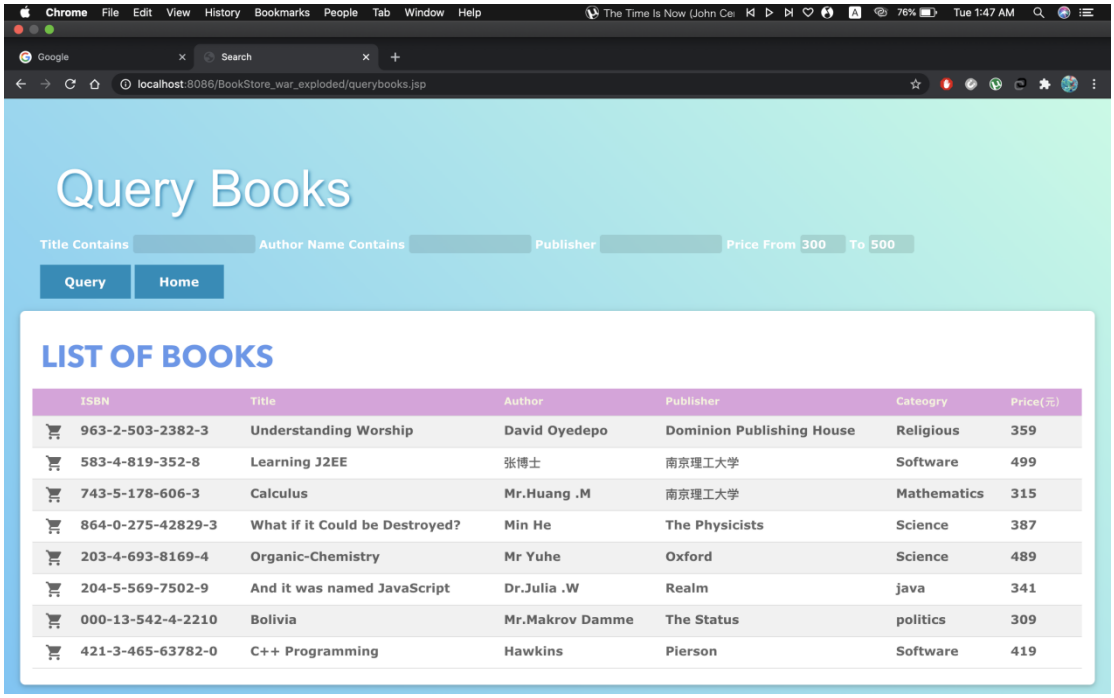
Searching by Title



Searching by Author



Searching by publisher



Searching by price range

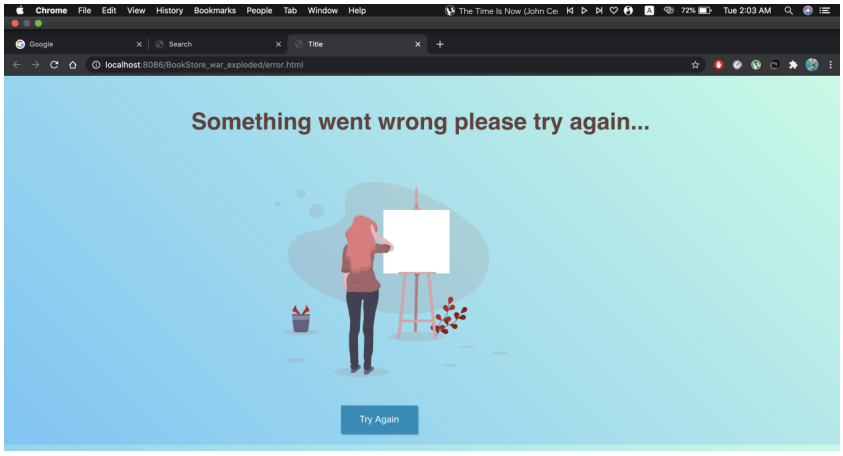
Finally the Log out page

This page did nothing more than redirect the user to the log in page.Thank you for your time.

Challenges:

- The first challenge I faced was understanding what was asked of me.
-The second one was finding the right tools to use to do this.
- The third one was the ability to connect to a server.This was really difficult.(It took me 3 days to do this).
- The fourth part was running into errors of sorts mostly by MySQL queries.
- Fifth was the logic reasoning behind doing all of this.

Conclusion: I learned a lot in this through the several J2EE documentations and research I had to read grasp the concept of what I was asked to do.



This is the error page of the website incase anything goes wrong