**Assignment6**

***Tang Victor – Jin James***

**1 Assignment Requirements**

A – Make a website connected with a database

B – Fill the database with information about books

C – Create login, sign in and logout pages

D – Able to connect to database through php programs

**2 Platform and Tools**

Windows 10

Atom (CSS/HTLM/PHP/MySQL)

WAMPSERVER (Database)

**3 Background application**

We were assigned to build a website which manages books. It’s a library website so it includes information about books (price, title, synopsis, author and ISBN) and we can also buy the books we want through a system of online payment.

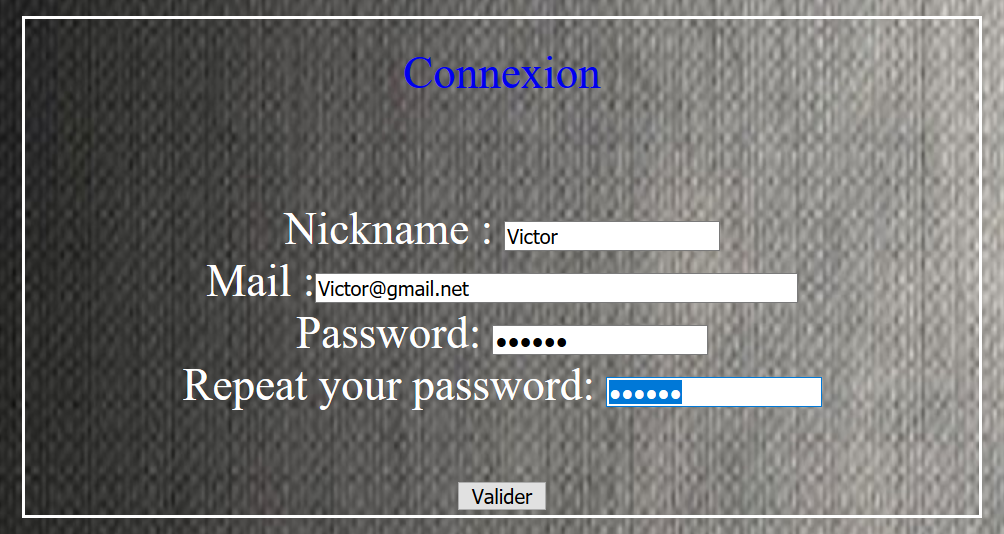
In order to do such a project, we need to be familiar with php language which is the main language for connecting a database with an internet page.

Our application is a website where we can find information about books, so we called it Library.

**Main content**

**1 Login/Signin page**

Before accessing the functionalities of out website, the users will be needed to have an account in our database. To do so, the first page that an user will be presented is login/sign in page:

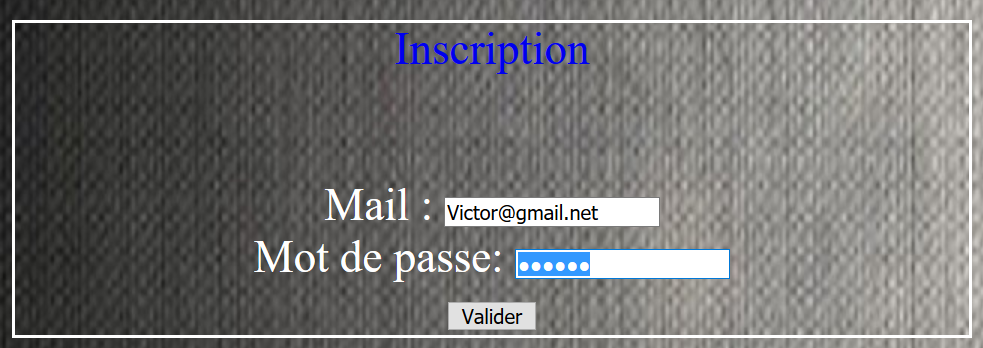


In this sign in page, we can fill our nickname, mail, password. We added a security: the user will have to fill his password a second time to avoid mistyping.

Once we valid the information typed, it will be added in our database:



And now, we can connect into our website:



Here is the code for login and sign in pages:

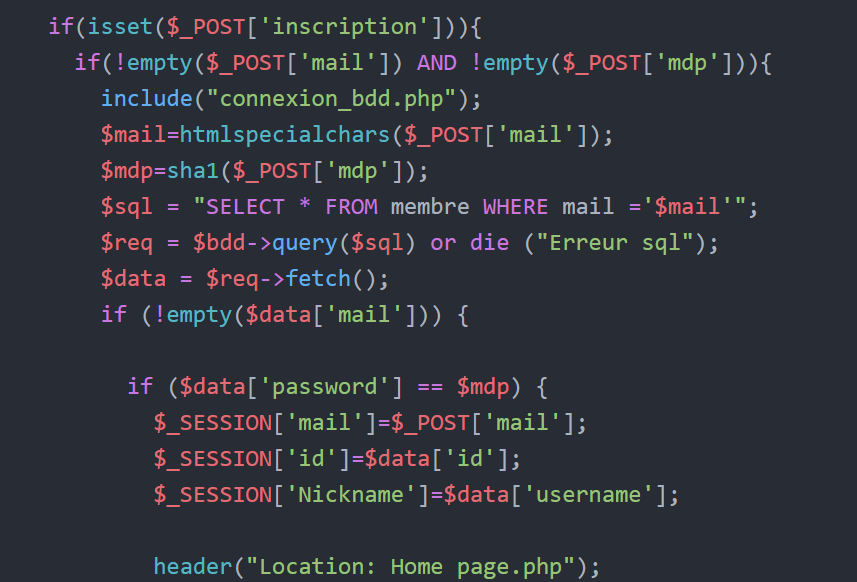
Sign in page:



We are going to focus on the php/SQL part. First of all we check if the user has filled all the forms by using “!empty(…)”. Then we collect what he typed into different variables such as $mdp, $nickname and $mail. After that, we make a request to the database: “**INSERT INTO** membre […]”.

So after collecting the user’s information, we insert them into our table “membre” (which means member).

Login page:



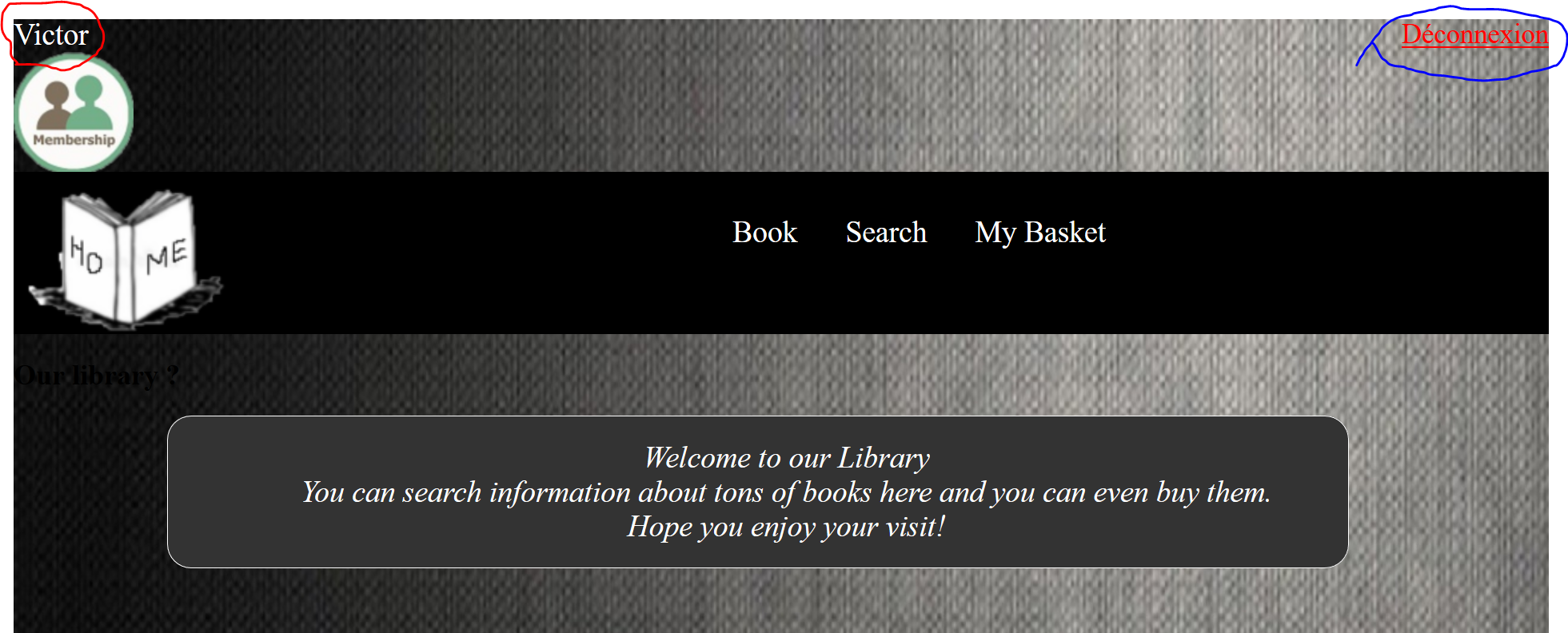
Once again, we only focus on the SQL part.

We set the variable $sql with the value “**SELECT \* FROM** member **WHERE** mail = ‘$mail’”. So we chose everything from the table member where the mail typed by the user is the same as the mail registered through the sign in page. After that we also verify if the password input by the user is also the same as in the database.

If everything is good, then the current session will be given all the information saved in the database. For example, if the username is “Victor”, then the session will have “Victor” as nickname. That’s why it’ll be printed in the top left corner of our website.

**2 Home page**

Once connected, the first page shown is the welcome page:

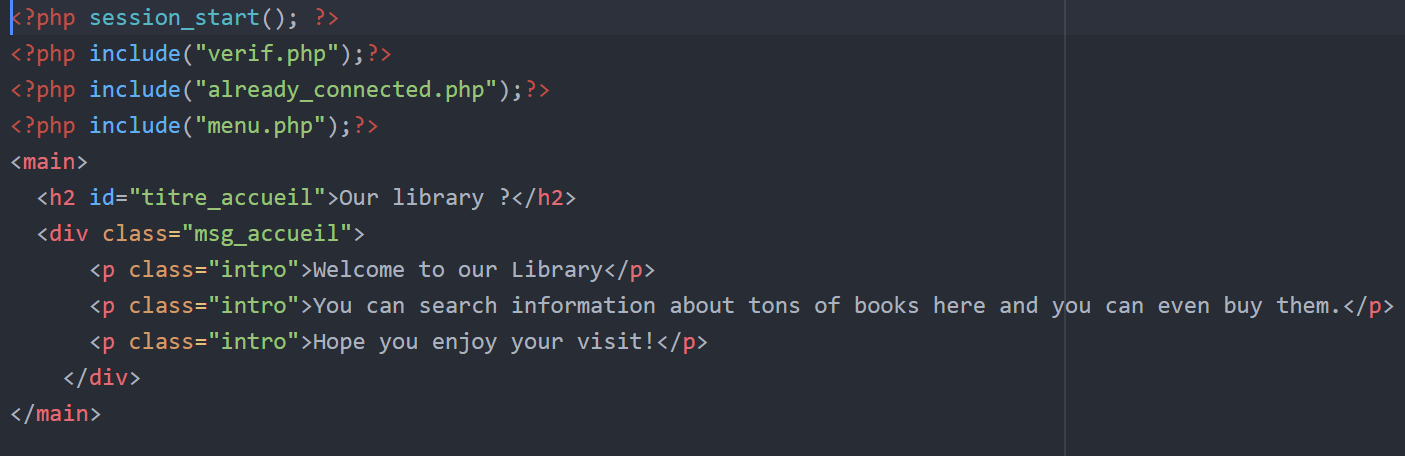


The design is quite basic, we have a navigation menu where there are “book” (to see all the books stored in our database), “Search” (to check whether a book exists in our database) and “My basket” (to see all the books we added into our basket in order to buy them later).

As we can see, in the top-left corner we have our nickname displayed and in the top-right corner we have “Déconnexion” which means log out.

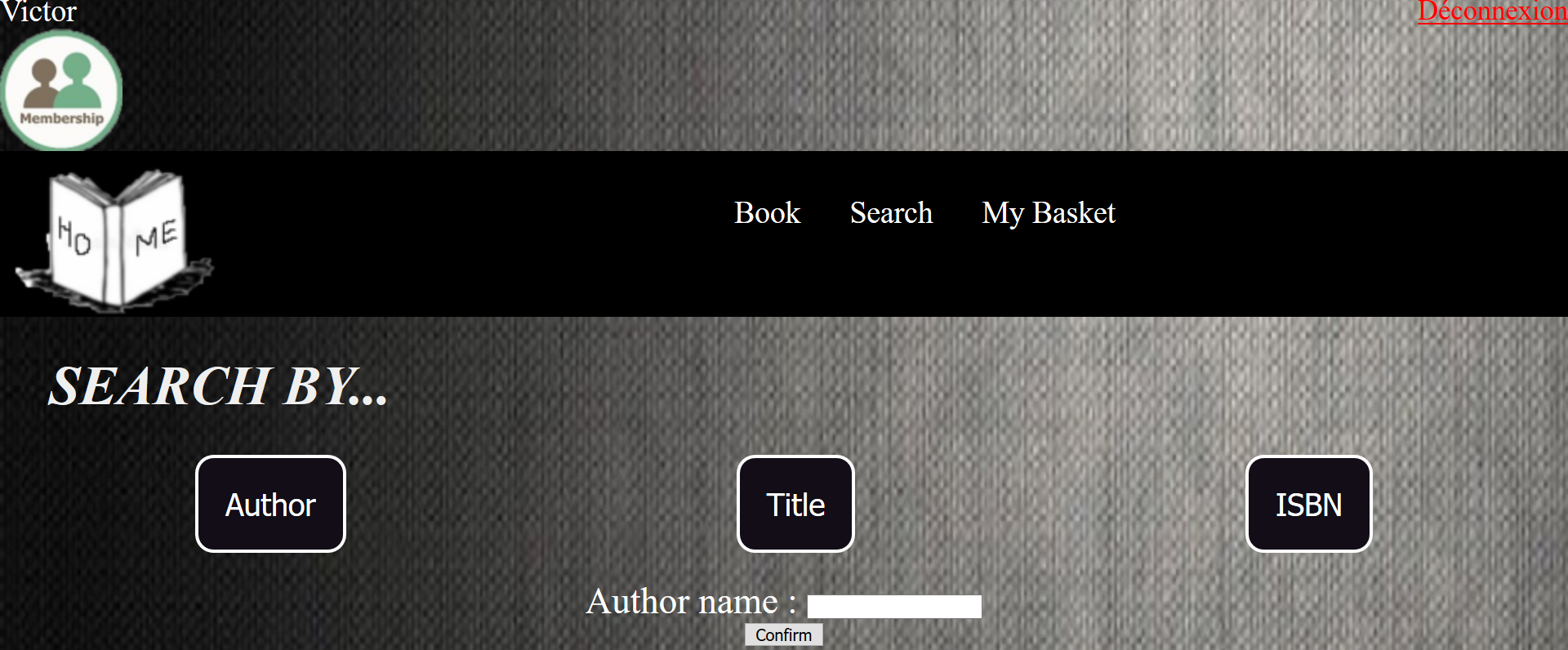
Then we have a message displayed for welcoming our user.

Here is the code for Home page:

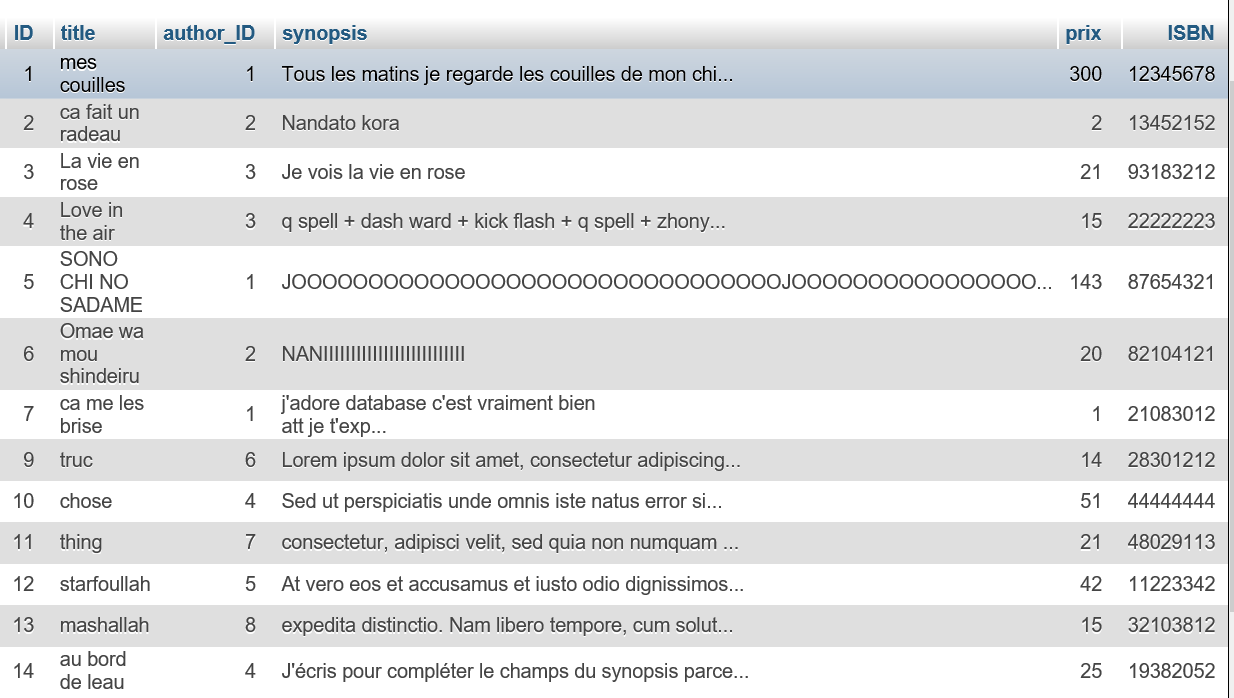


Nothing interesting to say about this page since it’s all html.

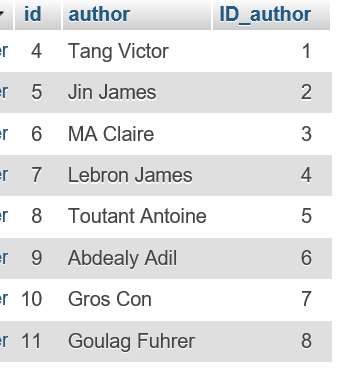
**3 Search page**



In our Search page we can search books by searching author name, title name, or ISBN number. First, let’s take a look in our database, to see which books I have:



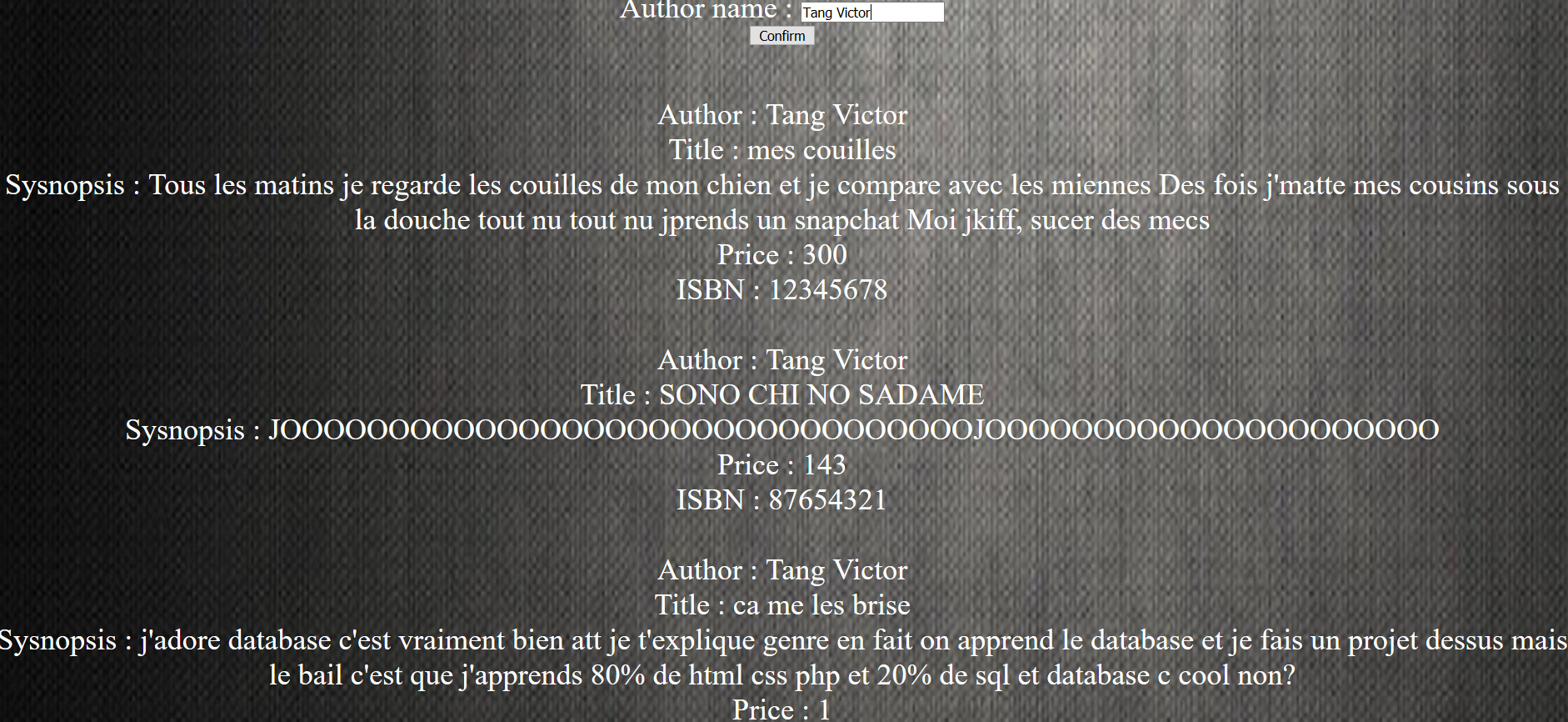
And the author table:



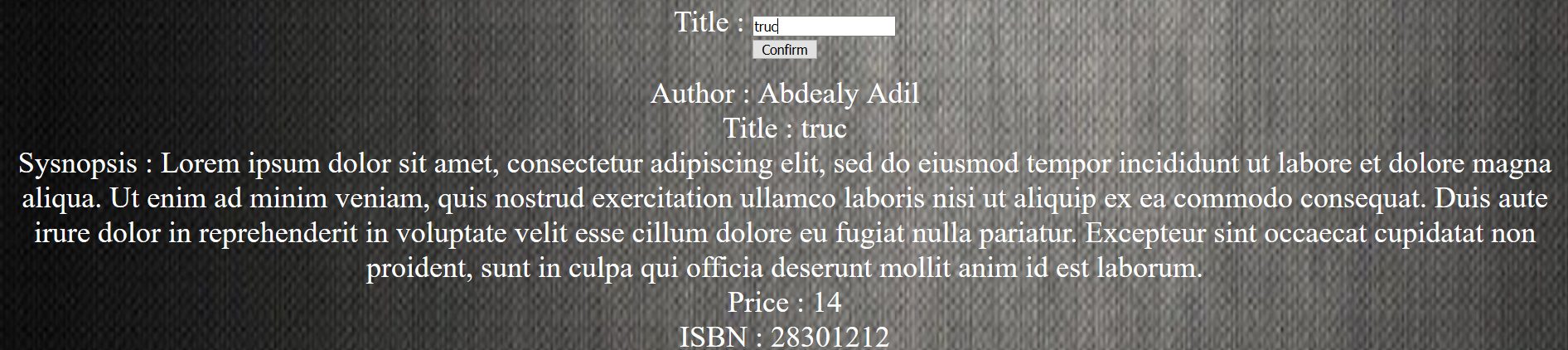
As we can see, for each book there is an “author\_ID” which obviously corresponds to a “ID\_author” in author table.

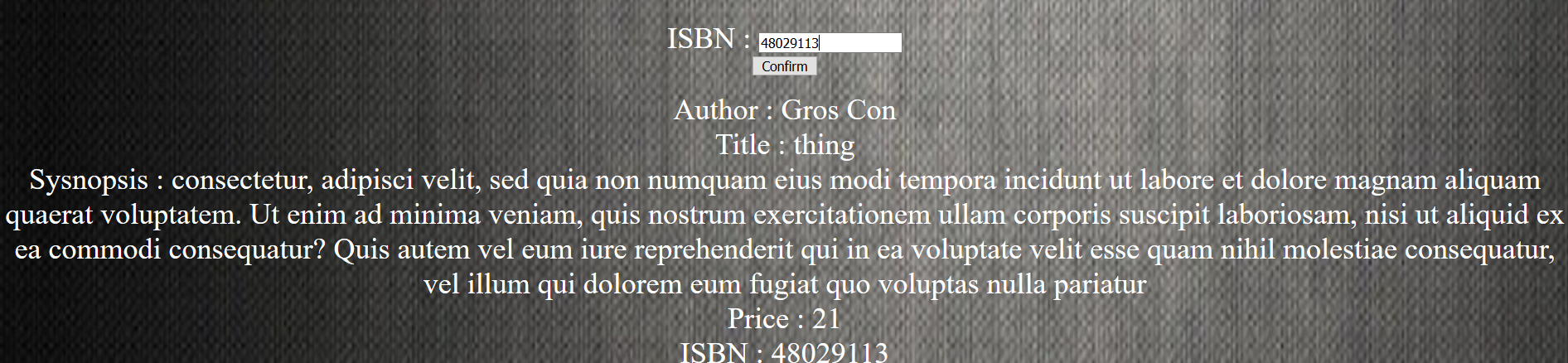
The meaning of having two tables with author and books separated is because one author can write several books. Whereas id, author, ISBN and title are unique. The advantage of doing so is for manipulating tables, it’ll be more convenient.

Let’s see the results in our website:



If I search “Tang Victor”, the website will show me all the books Tang Victor has written and their information.



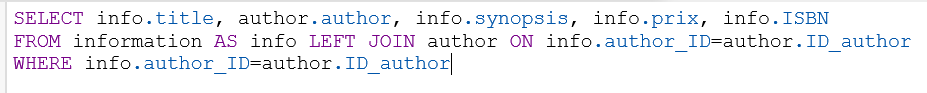


We of course have the same results for searching title or ISBN.

Let’s see the code behind these:

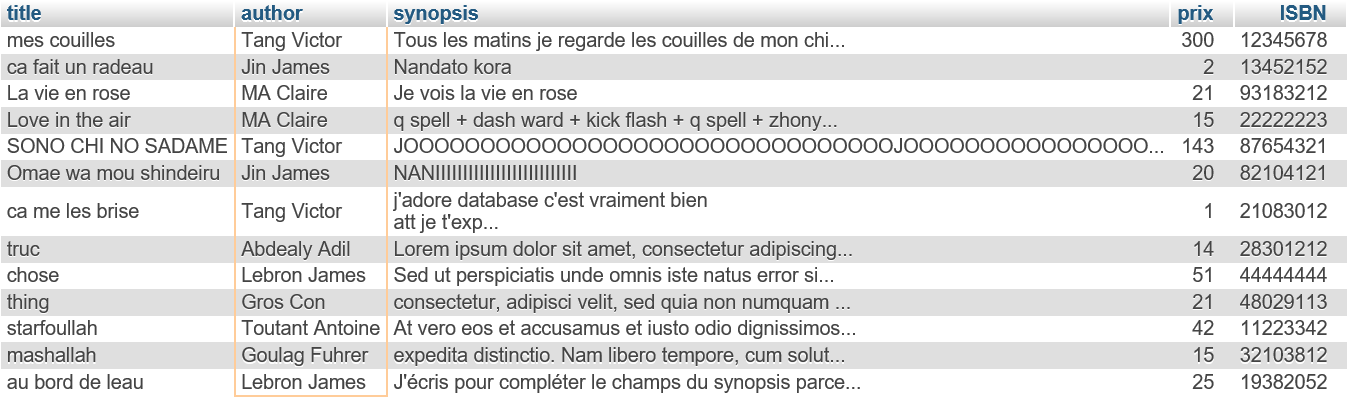
For showing the information by searching by ISBN:

In our book information table we don’t have the name of author, we only have the ID of the author. So in order to display the name of the other, I had to join the author table and book information tables together. We can make such a query directly in our database, using the SQL language:



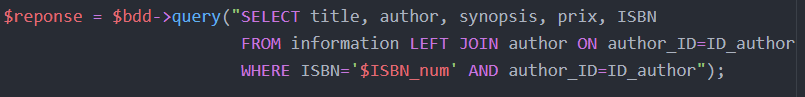
We **SELECT** all the information we need, so author’s name and the information about the book. Then these information come **FROM** the two tables information which will (**LEFT/RIGHT) JOIN** the table author **ON** the criteria the ID of authors should corresponds to each other. Then of course, we display the title, synopsis and everything **WHERE** the criteria “info.author\_ID=author.ID\_author” is respected.

This is what we get:



Our two tables had been successfully joined.

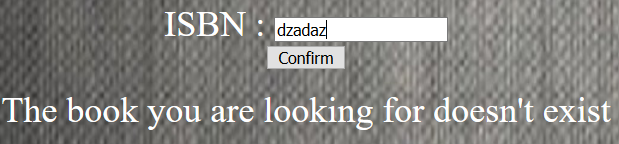
But this is only by using SQL language. We need now to implement this in our php language. This is the part of code which interests us:



We have something new here in our SQL query, it is: “**WHERE** ISBN=’$ISBN\_num’”. The previous SQL query I just showed allowed us to display all the books. But since we are searching by an unique ISBN number, we need to say in our query that the ISBN in our database must be the same as the ISBN number that the user input in our website.

Then the rest of the code is about displaying information and verifying if the ISBN number exits in our table.

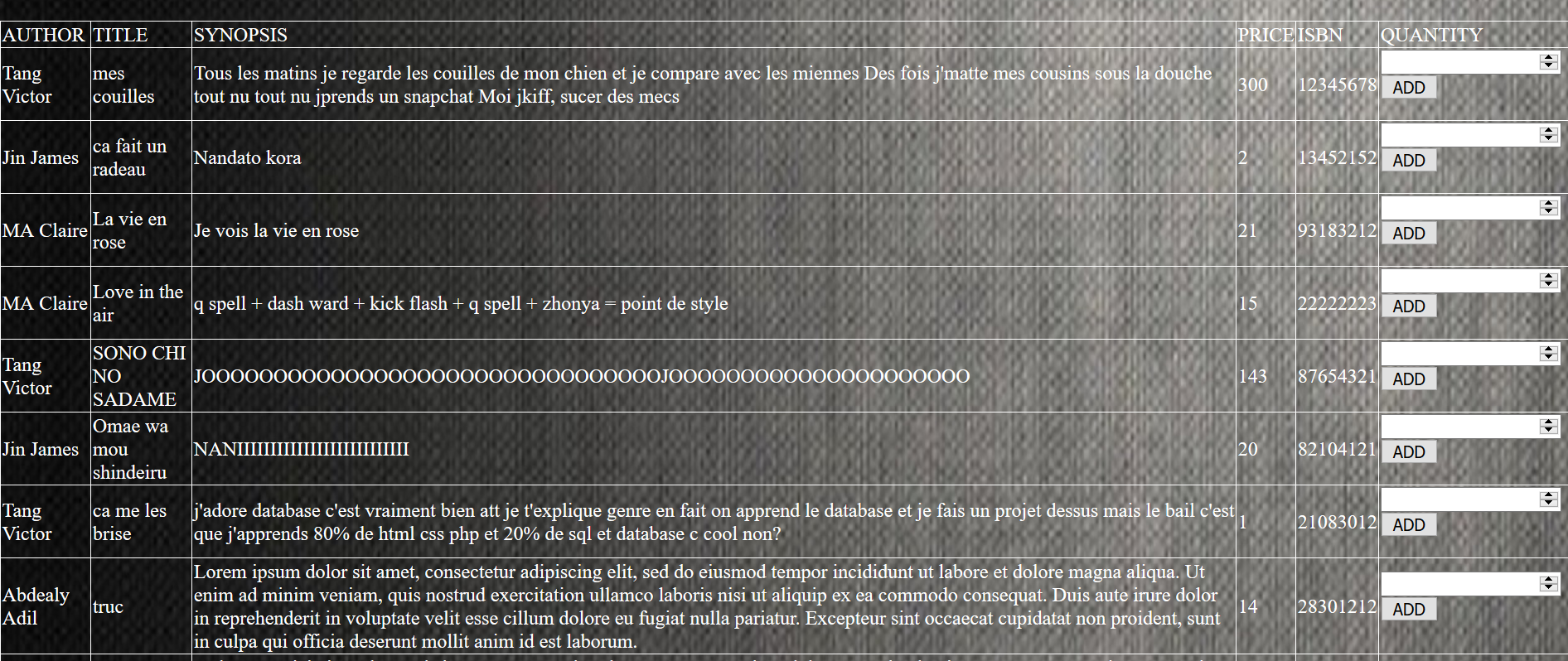
If it doesn’t exist, It’ll show:



For searching by author name and by title name is basically the same principle. But instead of having “**WHERE** ISBN=’$ISBN\_num’”, we will have “**WHERE** author=’$author\_name’” (for searching by author) and “**WHERE** title=’$title\_name’” (for title).

**4 Book page**

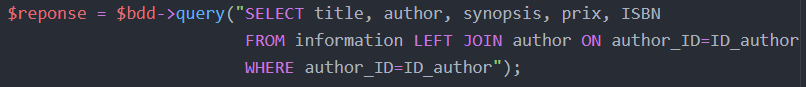
We obviously need to display all the books that exists in our database:



In our Book page, all the books are listed in a table. We have the author name, title, synopsis, price and the ISBN. At the end we have button “Add” which allows us to add a book into the basket of the user and of course the quantity of the desired books.

Let’s see the code behind this:

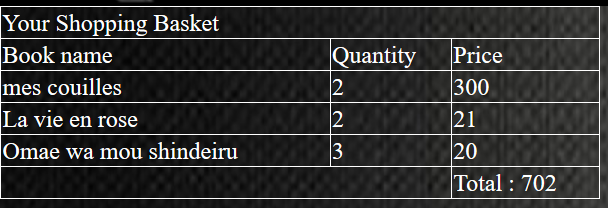
Actually, I already explained how we are going to process. I showed the SQL language to show all the information about our database, so we only need to implement this into the php language:



This code is basically the same as “searching a book by ISBN”. We just removed the “**WHERE** ISBN=’$ISBN’” since we want every book. The rest of the code is also not really interesting, it’s about collecting the information and displaying it into a <table>.

**5 Basket page**

After adding your desired books, the basket page will resume all of your purchases. Let’s see how it looks like:



As we can see, the total of the purchase is 702€. “Mes couilles” is a really expensive book.

The code behind this is full of php language. We could create a table “basket” in our database but we decided to do it through PHP language:

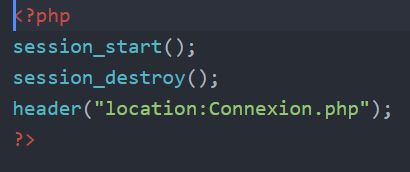


It’s not really interesting to explain such a code, but to summarize we create a table where we will input the data of the desired books.

**6 Logout**



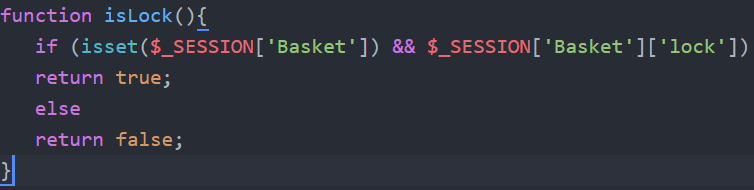
We don’t have a specific page to logout. You can logout from your account at any moment because our logout button is present in every page.



There’s no SQL language here, we just destroy the current session and redirect the user to the first page: “connexion.php” which is our login page.

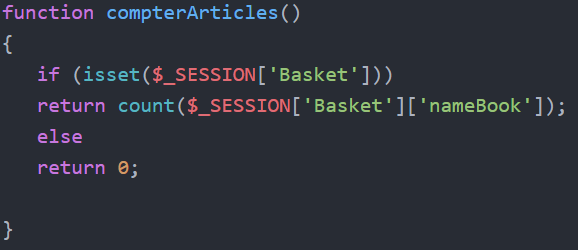
**Function Design**

Islock():



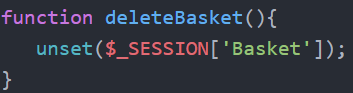
The basket is locked normally when the customer pays. The basket cannot be changed: he’s locked. Thus, this function is used to know if the basket is locked or not.

compterArticle():



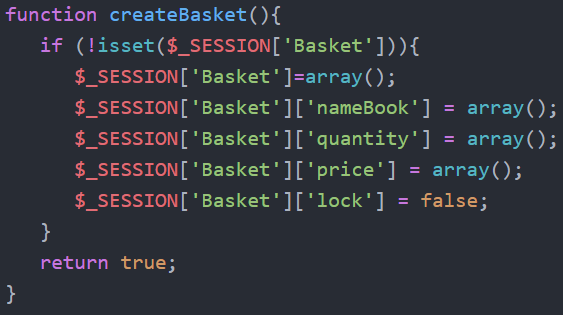
This function is used to count the articles which are present in the basket.

deleteBasket():



As the name says, this function is called for deleting the basket. So we use this function after the payment, so the basket will be empty again. Of course, it’ll be called if the user disconnects.

createBasket():



This function is used to create the basket of the customer in the session. So it’ll be created only if the user is already login. And when he logout the basket will be deleted. The Basket is composed by the nameBook, quantity, price and lock.

Addbook():



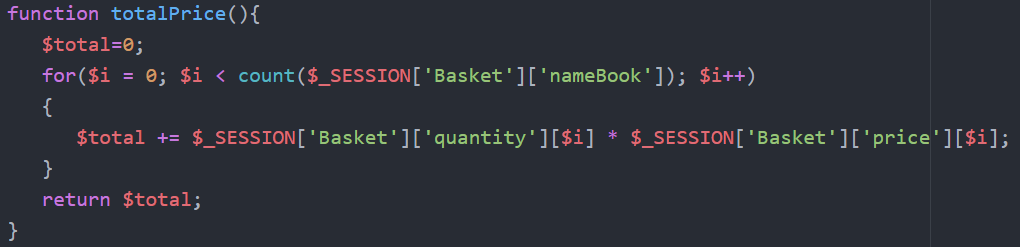
This function is used to add a book into the basket. So whenever the user will click on “Add” this function will be called. We first verify that the basket exists and that the basket isn’t locked. If everything’s good, we will check if the product added is already in the basket, if yes, we will only increment the quantity. If not, we add the whole product (name, quantity and price).

DeleteBook($nameBook):



This is for deleting a book in our basket. To do so, we will have to create a temporary basket declared by $tmp=array(). Of course to delete a book, the function takes a parameter which is the books which will be deleted.

totalPrice():



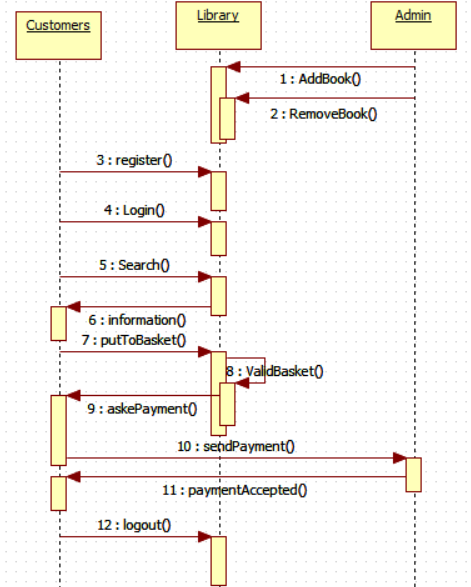
This function is used to return the total of the needed money. To do so we count the number of product in our basket and we focus on the price and quantity. And of course, for each line we will to quantity\*price until the end of the table.

**Software design**

**Use Case:**

****

**Sequence Diagram:**

****

**Conclusion**

To conclude, we would say that this project is the perfect project for finishing a semester. This project is a projection of the utilization of Database. Websites are everywhere now, and every sector have a website. Company, restaurants, governments, schools, institute etc, all structures must have a database, and a website.

So the library that we were assigned to do is a little projection of the reality. And we learned a lot from this.

If I can add one last thing to criticize this project, we had to learn the php language but we never had any class about it. But fortunately, when it comes to programming internet is the most wonderful tool for it.