E-Commerce Application Assignment

Practical information

In this assignment, we want you to create an e-commerce application on which users can buy products. Apart from pizza, you can choose anything you want as products (e.g. watches, shirts, shoes, food, …).

We suggest you work on the "eShopUlg" project where customers can buy University products (the appropriate data will be made available to you). An example website is that of the University of Limoges: https://boutique.unilim.fr/.

The example codes explained during the lectures should suffice to complete the assignment. You can (modify) and re-use them. Obviously, you are strongly encouraged to incorporate additional features.

- The assignment must be done in group of at most 3 students.
- The deadline is at 23:55, the 12th November 2018.
- The assignment will count for 30% of your final grade.

The e-commerce application can be coded in **PHP** and must contain seven main sections:

1. Registration page (for the eShopUlg example, you have to consider 3 categories (student, alumni or member of the school))
2. Login page
3. Product catalogue with for each product, its image and price, and the possibility to add to the basket, and also a search tool
4. Basket Validation Page: summary of basket including the image, product name/description, price per unit, quantity ordered, sub-total and total. This page should contain a validation button, which when clicked brings the user to the payment page
5. Payment page, with the summary of the order (product name, price, sub-total and total) and a pay button which brings to a blank page.
6. "About" information page
7. Admin page with stock management and the possibility to export XLS files with dates criteria and stock. This page can only be accessed by a super user with username and password "admin".

Concerning the database, you have to create a new database with the name "groupX" where "X" is the number of your group. Your database must be composed of, at least, the three following tables:

- **Product**: This table contains the data of the sold products. As mentioned earlier, you can choose eShopUlg or any product in which you are interested, for e.g. shirts, toys, CDs among others. This table must contain the product id, name, image, price, and quantity in stock.
- **Customer**: This table contains the data of the registered customers/users.
  This table must contain the customer id, username, password, forename, last name and mail. If you have chosen the eShopUlg example, don't forget the category (student, alumni or member of the school). Please make sure that the password field is large enough to store encrypted passwords.

- **An Order and/or OrderDetail table:** These tables (this table) keep track of customers' transactions, including date and time of purchase, product (ids), customer (ids) and quantity. Note that a customer can have multiple orders.

An example database will be shown in the lecture.

You can choose any name you want for the tables. Other tables can also be created if necessary. Moreover, additional fields can also be added to the existing tables.

Submit your final project

One of your files (e.g. php page) must contain the code to connect to your database. This page must be included in the other pages where a database connection is made. An example will be shown during the lecture.

You must submit your final project in a ZIP archive on LOLA. This archive must contain a directory with the name "groupX" where "X" is the number of your group. Inside this directory, you must have:

- The files (e.g. php, python, js) of your application;
- The images used;
- Your database: All the tables used in your project must be exported in SQL format (one ".sql" file containing the creation of your tables (CREATE TABLE) and some data (INSERT INTO) that can be used immediately (product, …)). If you do not know how to do it, please let us know.

The installation of your application must be straight forward. We should only have to copy the directory in your ZIP archive into our WAMP directory, and import your SQL database and your web application should run correctly.

Ask us if you have questions (on any aspects of project, e.g. database design, programming, …)

Minimum requirements

The minimum requirements for the project are as follow:

1. New users (customers) can register to the website. 3 marks

   A new user can register to your website by giving (at least) a username, a password and a name. These fields must be MANDATORY. A mandatory field is a field that the user must absolutely fill with some value. This kind of field cannot be empty.

   Hint: To make a field mandatory, take a look at the exercise (in the "Cart directory") that allow the user to add a quantity bought. In that exercise, the quantity field cannot be empty.

   When a user wants to register, check if the username entered is available or already stored inside the database.

   - If the username is already used, print a message warning the user that this username cannot be used. He must enter another one to be able to register.
   - On the other hand, if the username is not used (it is not yet stored inside the database), confirm the registration

2. A user can log in inside the website. 1 mark

   A user, which is already registered, can log in with its username and password. If the username and/or password supplied by the user are wrong, an error message should be displayed and the user will not be logged in. On the other hand, if they are correct, the

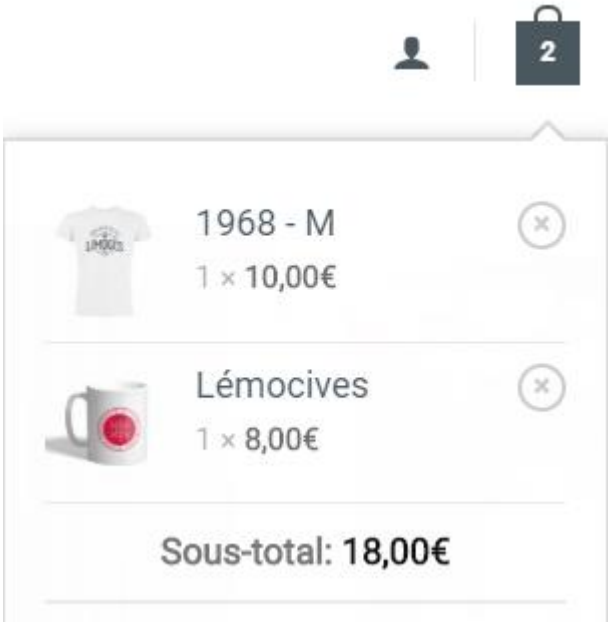
user is logged in and can access the other pages of the website. On all other pages of the website, he can see his forename and last name.

Hint: use a Session to store the user information and show them on all pages of the website.
Notes: We expect the password to be stored in its encrypted form inside the database (sha512).

3. Users have a basket containing their products. 1 mark

   A logged user can add products with a given quantity to his basket. The basket must display some data (e.g. total number of products, total price). You can be creative and display the basket in a different way than the one shown during the course.

   Notes: The basket must be visible on the different website's pages.



4. In his basket, the user can add different products and the quantity to be bought. Imagine that the user has selected two products: 2 marks
   - product 1, quantity 10, price per unit 2€
   - product 2, quantity 2, price per unit 6€

   Calculate the total price paid for each different types of product. In this example, you will have to print on the screen:
   - product 1, quantity 10, price per unit 2€ => **Price: 20€**
   - product 2, quantity 2, price per unit 6€ => **Price: 12€**

   The quantity entered should be >= 0, otherwise an error message should be displayed and the product should not be added to the basket.

   If the same product is added to the basket multiple times, then an aggregated quantity should be displayed. Suppose that the user has selected the two following products and put them inside his basket:
   - product 1, quantity **10**, price per unit 2€
   - product 2, quantity 2, price per unit 6€

   Now, he wants to add, one more time, "product 1" to his basket with the quantity 3. The expected result is as follow:

- product 1, quantity **13**, price per unit 2€
- product 2, quantity 2, price per unit 6€

Here are some steps that must be done:

- With a loop, check if the selected product is already present in the basket or not.
- If the product is present, calculate the new quantity (sum).
- If the product is not present, simply add it to the basket.

Also, your code should check whether the quantity purchased by a user is available in stock. If the quantity entered is bigger than the stock available, print an error message and do not add the product to the basket.

5. The user should see the total amount to be paid for all the selected products in its basket. Imagine that the user has selected two products: 2 marks
   - product 1, quantity 10, price per unit 2€
   - product 2, quantity 2, price per unit 6€

   Calculate the total amount to be paid for all the products contained inside the basket. In this example, you will have to print on the screen:
   - product 1, quantity 10, price per unit 2€
   - product 2, quantity 2, price per unit 6€

   TOTAL: 32€

6. Implement auto-complete function which should be as follows. On your "product catalogue" page, put a link to an html page containing a text field. Each time a user enters a character in the field, an event is generated and a request is sent to a different php page which returns suggestions for the product name. Example : https://www.w3schools.com/xml/ajax_php.asp. Note that this example uses a pre-defined array but in your project, you will use your product table. 7 marks

Additional requirements

You can gain additional grades for your project by completing at least one of the three additional requirements: 7 marks

A In the registration page, fields appear depending on the category of the customer. For alumni: employer and function. For students: faculty and registration year. For university employees: research project number (which is a string of size 10). You may use AJAX/JQuery to show/hide fields.

B At the basket validation page (before validating), possibility to change the quantity ordered or delete the product, as shown below

PRODUIT		PRIX	QUANTITÉ	TOTAL
	1968 - M	10,00€	- 1 +	10,00€
	Lémocives	8,00€	- 1 +	8,00€

C On the payment page, generate a QR code which can be scanned by your phone You can use this library: http://phpqrcode.sourceforge.net/index.php#demo

D Allow the administrator to view and export his product stock based on various criteria, e.g. quantity in stock < 10 or goods sold between a time period.

Evaluation

The total marks for this assignment is 30.

- Zero if one the 3 tables is missing ;
- Zero if one of the 7 sections is missing ;
- Zero if students do not show up for the presentation (see below) ;
- Minimum requirements are evaluated on 18 ;
- Additional requirements are evaluated only if the minimum requirements are completed.;
- Business Model and presentation are evaluated on 5.

Presentation

Each group will have to do a short presentation describing their work and showing a demo.
Presentation sessions are scheduled for the 19th November 2018 during the normal lecture hours in the usual lecture room.
Each group will have 15 minutes to present : 7 minutes for presentation and 8 minutes for questions and answers.
Presentations should include :

- A brief description of the website including the technical aspects ;
- A business plan for the product ;
- Demo of the minimum requirements ;
- Demo of the additional requirements.