NumericALL

# PHP / SQL

# Final practical assessment

28th November 2022

**Time allowed : 4h00**

## Marking

At the end of the allotted time, you must deliver all of your files in a Github repository, and a score out of 20 will be given based on to the following scale:

**Exercice 1 : 2 points**
**Exercice 2 : 4 points**
**Exercice 3 : 4 points**
**Exercice 4 : 8 points**
**Other : 2 points** ○ Indentation and readability ○ Code Comments ○ Variables name

***NB: it is not permitted to copy and use code obtained from other students of this training or previous students of this training - doing so is considered cheating and will be sanctioned.***

### ***Tips***

***If the task seems overwhelming, don't worry! Each step is detailed to move forward little by little.***

- Read the entire statement from the start, to know where you are going.
- Take the time to code, commenting on your code as soon as it is necessary, why not by copying the instructions in comment. The proofreader must be able to understand what you have done!
- Keep functions simple, which do only one thing, the better to navigate.
- Focus on PHP rather than CSS/HTML !
- The more your code will be indented and readable, the easier the rest will be!
- Each exercise is independent, be sure to separate them into different files or folders.

## Exercise 1: GitHub

You must upload your evaluation files on GitHub and share the link at the end of the evaluation.
For this purpose, the repository must be **private** !

If you can’t push to GitHub successfully, you will have to zip your evaluation and send it by email.

*Recommandations :* Try to do this first, before starting to code.
If you can’t figure it out, move on to Exercise 2.

## Exercise 2: Debugging - StarDiscover

An intern, for which you are responsible, must retrieve a list of ships from a database but also give the possibility of adding them using a form.

The main script consists of two parts:

- The list of ships and their information
- The form for adding a new ship

The intern begins in PHP programming. It does not indent or comment on its code. As an internship supervisor, you must help him correct and make his code work.
Good indentation of the code is also necessary.

/!\ You can ask the teacher to show you the final/working version.

**Implementation steps:**

1. Download the ‘Exercise 2’ folder on google drive.
2. Deploy the files index.php and database.php on your local server and import the SQL script spasceship.sql. This will create a new database containing a table “ships”.
3. Help your trainee to debug his code, correct his errors and make it work.

## Exercise 3:  Online Shop

We will work for an Online TShirt Shop. The owner wants to be able to add its products and display them. We will take care only of saving the products in the database through a form.

In this exercise, you will have to create a new database and a form.

This form will allow you to add a new t-shirt in the database.

**Step 1 :**

Create a new database called “online_shop” and create the ‘tshirt’ table matching this structure :

**TSHIRT**(id, name, price, color, size, description)

Available sizes are : XS/S/M/L/XL.

You’ll have to export the database and attach it to your evaluation folder.

**Step 2** :

Now, you need to be able to add a new shirt thanks to the form.

You have to follow those recommandations :

- The form will be processed in Ajax.
- Size must be a dropdown-list to choose from XS/S/M/L/XL.
- All the inputs are mandatory : validate the input coming from the form.
- Insert the tshirt in the database
- A success or error message will be displayed (in green or red).

# Exercise 4: Handle the Users

**Step 1 :**

You have an online shop to sell your products.
You want to be able to handle the users of your website.

As part of the site design, you must create a User() class which will have the following private properties:

- a nickname (string: between 5 and 15 characters)
- an email (string: must be valid email)
- a password (string: between 8 and 16 characters)
- a property 'admin' (boolean: false or  true)

By default, all users are not administrators. Set the 'admin' property to False (0).
Make the getters / setters to validate the data type above as well as the constructor allowing to instantiate the class.

Add one method:

- edit_password: Edit password is possible at any time. Edit password operation asks for the old password and the new one. It checks if the old password is correct before changing to the new one.

**Step 2 :**

You can choose between two user accounts : *AdminUser* or *RegularUser*

Create the matching classes using inheritance.

***AdminUser()*** the admin property that is defined at the creation of the user will be set to True (1).

***RegularUser*()** will have a 'cart' property. It has to be defined to empty (array) when creating the user.

A product is represented by a name and a price.
Add three methods:

- add_to_cart($product): Add the product to the cart. You must save both the name and the price and set the quantity to One (1). If the product is already in the cart, you have to increment quantity by One (1).
- remove_from_cart($product): Remove the product from the cart.
- display_cart(): Display the content : name, price and quantity for each product but also total price.

## Part 3 :

In a new file, instantiate the class so that you can display 2 different users (one *AdminUser* and one *RegularUser*).

For the RegularUser, try to add a product to the cart and display the content of the cart.