Distributed Programming (03NQVOC)
Distributed Programming I (03MQPOV)

Laboratory exercise n.7

Objective: writing dynamic pages in PHP driven by a database. A PHP manual is available to the URL http://php.net.

In this exercise the XAMPP suite will be used. It is already installed in the laboratory machines in the windows environment. After launching XAMPP, activate Apache and MySQL from the control panel. The folder in which the files will need to be placed is **c:\xampp\htdocs**. **Create a sub-folder** and work just inside that one to avoid destroying xampp content (like its own links, etc).

In order to write PHP pages the files in which the code is stored must be terminated with the ".php" extension, otherwise the pages will not be recognized by the Apache web server.

Moreover, to verify the proper functioning of the PHP pages, it is mandatory to access them by using the web server address (for example: http://localhost/myfolder/page.php) and not by using their local physical path.

To debug the proper working of PHP pages it is suggested to follow the same recommendations provided for Lab 6.

It is possible to analyze the DB content used by MySQL through a web browser interface, by using the URL http://localhost/phpmyadmin

For the exercises from 7.2.1 onwards, at first use phpmyadmin to create an "example" database in which must be imported the books.sql file provided along with the exercise. This file will create a table called "books" and it will insert some data (fields ID, Description, Price, Quantity). The student can add other tables to the database to store further information like users and password, the shopping cart, etc.

In PHP, use user='root' and pass='' to connect to the DB MySQL.

Exercise 7.1.1

Create a page A containing a form to send the following data to URL B:

- a name, composed of at most 10 alphabetical characters, in which the first one must be upper case and the other ones must be lower case.
- an age (a numerical value ranging from 0 to 199)

- a phone number, composed of a prefix (two or three digits, in which the first one must be zero) followed by the character “-” and then by a number (six or seven digits) that may contain another character “-” after the first three digits.

In the case a data does not respect its specification, an error must be signalled (client side, if javascript is enabled, or server side otherwise). To disable Javascript in the browser, in firefox: Tools -> Options -> Content -> Enable JavaScript.

Page B must show the introduced data, if correct, otherwise it must show an error message and propose a link to come back to page A.

Exercise 7.2.1

Create a page that queries the database and shows all the inserted values, by filling up a table composed of three columns:

- the first column must contain the product name
- the second column must contain the available quantity of the product
- the third column must contain the product price

Exercise 7.2.2

Repeat the previous exercise by just showing **10 records a time**. Insert a button that allows loading the next block of data. The actual page number must NOT be passed as a form parameter. Please avoid loading, at each page view, all the data present in the DB.

Exercise 7.2.3

Create a page A listing the products present in the database (by showing ID, Description and Price) and also containing a form to place a buy order, specifying the ID of the chosen product and the desired quantity.

Create then a page B that receives the form data and returns the total price for the selected product and the desired quantity.

Exercise 7.2.4

Create a page A listing the products present in the database (by showing ID, Description, Price and Quantity) and also containing a form to place a buy order, by specifying the ID of the chosen product ant the desired quantity.

Create then a page B that receives the form data, shows the total price for the order and modifies the DB “quantity” field by reducing it of the bought quantity (if possible). Otherwise, as example if the desired quantity exceeds the available one, an error message must be shown. The page must also contain a button or a link to come back to page A.

Exercise 7.2.5

Create a page A listing the products present in the database (by showing ID, Description, Price and Quantity) and also containing a form to insert a new product by specifying its name, quantity, and unitary cost.

Create then a page B that receives the form data and adds the new product to the DB, if data is correct, or that signals the missing or wrong data otherwise. The page must also contain a button or a link to come back to page A.

Exercise 7.2.6

Create a page A listing the products present in the database (by showing ID, Description, Price and Quantity) that contains, at the side of each product, also a textbox that can be used to write the desired quantity of that product. The page must also contain a button to place the buy order (by going to page B) and a button to reset the introduced data.

Create then a page B that receives the form data and shows a table containing a summary of the buy order (the list of all the products that the user wants to buy, along with their quantity and price) and the total cost. The page must also contain a button to confirm the buy order (by going to page C) and a button to come back to page A.

Create then a page C that receives data from page B, subtracts to the DB records the required quantities and prints a confirmation of the completed buy order. The page must also contain a button or a link to come back to page A.