Programming Assignment 2
PHP Scripting with Relational Database
Due on Thursday June 30 before midnight
Worth 8% of your final grade

Description

The goal of this project is to learn server-side web programming using PHP and a relational database system (MySQL). More specifically, you will develop an e-commerce web application where customers can buy books saved in the database.

This project must be done individually. No copying is permitted. Note: We will use a system for detecting software plagiarism. That is, your program will be compared with the programs of the other students in class as well as with the programs submitted in previous years. Note that, if you use a Search Engine to find similar programs on the web, we will find these programs too. So don't do it because you will get caught and you will get an F in the course (this is cheating). Don't look for code to use for your project on the web or from other students (current or past). Just do your project alone using the help given in this project description and from your instructor and GTA only. Finally, you should not post your code nor deploy your project on a public web site.

Domain

You have been hired to set-up a database for www.cheapbooks.com, which sales books on the web (much like www.amazon.com). A requirements analysis that was conducted has identified a number things about the operations and goals of CheapBooks. Each book sold by CheapBooks has a title, a price, a year of issue, a publisher, and a unique ISBN. It is written by one or more authors, but, of course, each author may have written multiple books. Although not available to customers, CheapBooks has information about each author, which may include name, address, and phone number. Each book is stocked in book warehouses and CheapBooks wants to keep track how many copies of a book each warehouse has, in order to check it's availability and complete the customer orders. Each warehouse has a Code, a name, an address, and a phone number. When a customer starts a session with CheapBooks through its web site, she is assigned an empty shopping basket so that she can make multiple purchases each time. The customer session is then to select books and insert them into the shopping basket. The shopping basket may contain multiple copies of multiple books. Note that, at any time you may have multiple customers buying books and a particular customer may have used multiple shopping baskets. At checkout, the total price of the customer's shopping basket is calculated and the customer is charged (assume that shipping cost is zero). At checkout, each warehouse involved in the purchase is received a single shipping order that contains the shipping information of the customer, and for each book in her shopping basket stocked in this warehouse, the number of copies bought. When the customer's shopping session ends, all information about the shopping basket is removed but the information about the warehouse shipping order is kept.

Platform

As in the previous projects, you will develop this project on your PC/laptop using XAMPP and you will test it using using your Mozilla Firefox web browser. Download the project2 files: project2.zip

from UTA Blackboard. Unarchive the files inside your web server document root directory. The project2 directory contains the file createDB.sql having examples of CREATE queries.

To create the database, start the Apache Web Server and the MySQL Database on your PC using the XAMPP manager console. Run phpMyAdmin on your browser, create a new database with name cheapbooks by clicking on New. Select this database, go to the SQL tab and write the appropriate CREATE table queries and push Go. You can run queries from CreateDB.sql for testing. You can test your setup on your web browser by using the URL address http://localhost/project2/customer.php

Requirements

Create your CheapBook database using phpMyAdmin based on the following schema:

Author (ssn, name, address, phone)	*info about a book author*
Book (ISBN, title, year, price, publisher)	*info about a book*
WrittenBy (ssn, ISBN)	*relates books with authors*
Warehouse (warehouseCode, name, address, phone)	*info about a warehouse*
Stocks (ISBN, warehouseCode, number)	*what books a warehouse stocks*
Customer (username, address, email, phone, password)	*info about a customer*
ShoppingBasket (basketID, username)	*relates customers with baskets*
Contains (ISBN, basketID, number)	*the content of a shopping basket*
ShippingOrder (ISBN, warehouseCode, username, number)	*books shipped from a warehouse to a customer*

In this schema, foreign keys have the same name as the primary keys they reference. More specifically, the foreign keys are:

WrittenBy.ssn → Author.ssn
WrittenBy.ISBN → Book.ISBN
Stocks.ISBN → Book.ISBN
Stocks.warehouseCode → Warehouse.warehouseCode
ShoppingBasket.username → Customer.username
Contains.ISBN → Book.ISBN
Contains.basketID → ShoppingBasket.basketID
ShippingOrder.ISBN → Book.ISBN
ShippingOrder.warehouseCode → Warehouse.warehouseCode
ShippingOrder.username → Customer.username

Then insert some data into the tables. Each table should contain at least four rows. Try to write your data in such way that you can get meaningful answers for various queries (such as "For each book, print the book ISBN, title, and price, along with the total number of copies of this book stocked in all warehouses"). You should declare all keys and foreign keys in your schema.

Project Requirements

Your script customer.php must be able to produce 4 web pages (can be from the same script or you can split it into 4 different php scripts):

1. Page1: a login form that has text windows for username and password, a "Login" button, and a

"New users must register here" button.

2. Page2: a "Logout" button, a textarea, a "SearchByAuthor" button, a"SearchByBookTitle", a "ShoppingBasket" buttom and a section to display the results of the search in the proper HTML format. Next to the "ShoppingBasket", there must be a counter Showing number of items in Shopping Basket.
3. Page3: a table showing all the items in the shopping basket with a "Buy" button. It should also show the total price of the basket.
4. Page4: a form to fill out user information along with a "Register" button

When customer.php is executed for the first time, it displays Page1:

•If the user enters a wrong username/password and pushes "Login", it should go back to Page1
•If the user enters a correct username/password and pushes "Login", it should go to Page2
•If the user pushes the "New users must register here" button, it should go to Page4

•On Page2, If the user pushes "SearchByAuthor", it should search the database for books having the same author name, it will reload the Page2 with search results.

•On Page2, If the user pushes "SearchByTitle", it should search the database for books having the same title, it will reload the Page2 with search results.

•The search results will show the BookName, ISBN number, Number of books available (stocks). It shouldn't show any books having its stocks = 0.

•On Page2, If the user pushes "ShoppingBasket", it should go to Page3, showing all the items in current shopping basket.

•On Page3, If the user pushes "Buy". Its going to save users current basket to the database, making appropriate queries to the tables ShoppingBasket, Contains and ShippingOrder. It will also update the stocks of book updating the Stocks table.

You need to use a PHP session to store the shopping basket (the list of chosen items throughout the session) as well as the results of the previous search. For each chosen item, ISBN, title, author name, price, and the number of stocks. You will also use the PHP session to create user session (for login and log out purposes).

Hints: Use md5 to encode passwords in PHP. Use uniqid to generate a unique id in PHP.

Documentation

The following are tutorials on PHP. Use them as a reference only.

•PHP Tutorial
•PHP
•PHP.net