CS 453 Electronic Commerce Technologies

Homework # 4 – PHP-based E-Store

Due: Monday, August 3, by 8pm that evening via electronic submission
Credit: 100 points

Instructions: You may work in teams of up to three people. You may utilize any publicly available resources regarding PHP. Don't forget the PHP material at http://iis.cs.virginia.edu/webweavers/ec.

Pledge: Electronic submission of your PHP code is your confirmation that every member of your team has complied with the honor requirements of this assignment.

Goal: Use PHP to create an e-store that can display inventory, display/validate/process an order form, collect/validate/process shipping and payment information, and confirm/deny the order. Choose a market niche (e.g., computer supplies, music, art) and generate an e-store that supports it.

Reward: Creativity is encouraged. Compared to other assignments, this homework has fewer specific directives to allow you more freedom in your design. The course staff will select the best website submitted and all members of that team will receive an extra credit bonus of 15 points.

Execution: Develop your code on a server of your own choosing. Submit all your code in a zip file via toolkit. Include in the zip file a one-line file named URL.txt that identifies the homepage URL of your e-store. To be clear, you are submitting all your code files so they can be examined as needed; we will execute your e-store from the URL you provide, which must point to an operational server that maintains your e-store. If your e-store is not running, then we can't see it or grade it.

1. *Database*. Create an electronic inventory containing a minimum of six items. The characteristics of each item must include an item name/description, a sales price, and a quantity on hand. The database may be maintained as a PHP file or as a mySQL database. You must be able to read, write, and update the inventory database. When submitting your program, be sure to set your file permissions such that both your code and database can be accessed.

2. *Timestamp*. Every webpage must display the store's name and a dynamically created timestamp. All timestamps must indicate when the page was dynamically created. An example timestamp is:

Tuesday, November 6 2007, 14:02:46

3. *Homepage*. Your store's homepage should include the name of the store, timestamp of creation, a graphic image appropriate to the store's business, a suitable welcome message, and one navigation button: "Proceed to Order Form".

4. *Order Form*. When the user clicks on "Proceed to Order Form", create a suitable page that displays the entire store's inventory. Typical information would include the item description, quantity available, price, and a textbox in which the user may enter the order quantity for each item. Example:

ACE Computer Supplies Order Form

Order Quantity	Quantity Available	Item Description	Price
☐	200	ACE zip disk	$ 30.00
☐	1500	ACE CD-ROM	$ 0.75
☐	50	ACE 1.44 MB floppy disk	$ 0.50

Return to Homepage Proceed to Checkout

All textboxes for Order Quantity should be initially null when this page is invoked from the homepage. This page allows a user to attempt to order one or more items by filling in a numeric quantity in one or more order quantity textboxes. For simplicity, when an order quantity textbox is left empty (null) or set to zero, that item is not being ordered; when the textbox contains anything other than null or zero, this is an attempt to order some quantity of that item. On the client side, enforce the rule that the "Proceed to Checkout" option can not be exercised until all textboxes are either null or contain only valid numeric entries.

Having filled in none, some, or all of the order quantity textboxes, the user has two navigation choices: "Return to Homepage" (which abandons the order) and "Proceed to Checkout".

5. *Inventory Verification*. When the user clicks on "Proceed to Checkout", first verify that the quantity of items ordered is actually available from inventory. If the user orders more units of an item than are available, create a response page that identifies which item(s) is/are not available and displays (again) the quantity actually available. In this case the only navigation option provided is "Return to Order Form". If all items ordered are available in the quantities requested, proceed to the checkout phase.

6. *Checkout*. Create a dynamic page that displays the user's order in tabular format, including quantity, item description, unit price, and total item price for the quantity ordered of this item. Do not list an item that was not ordered (i.e., if its quantity box was null or zero). Beneath the last line of the customer order, calculate and label the subtotal of the order, the Virginia state sales tax amount (subtotal times 5%), and the total cost of the order (subtotal plus sales tax). Example:

ACE Computer Supplies Checkout Form

Order Quantity	Item Description	Unit Price	Total Item Price
1	ACE zip disk	$ 30.00	$ 30.00
2	ACE 1.44 MB floppy disks	$ 0.50	$ 1.00
	Subtotal of order		$ 31.00
	Virginia sales tax		$ 1.55
	Total price of order		$ 32.55

Return to Order Form Proceed to Shipping

7. *Return to Order Form*. For simplicity, the only way to change an order is for the user to return to the Order Form page. When the user exercises this option, redisplay the full inventory page as shown before, but repopulate the quantity textboxes with the same information that was there when the user clicked on "Proceed to Checkout." (Hint: use cookies or session variables.)

8. *Shipping*. If all items ordered are available in the quantities requested, then create a Shipping page that shows the total price of the user's order and solicits the buyer's name, address, email, and credit card number. Example:

ACE Computer Supplies Shipping Form

The total of your order is: $ 32.55
Please enter your:

Name: John Q. Public

Address: 1543 East Main Street

City/State/Zip: Charlottesville, VA 22901

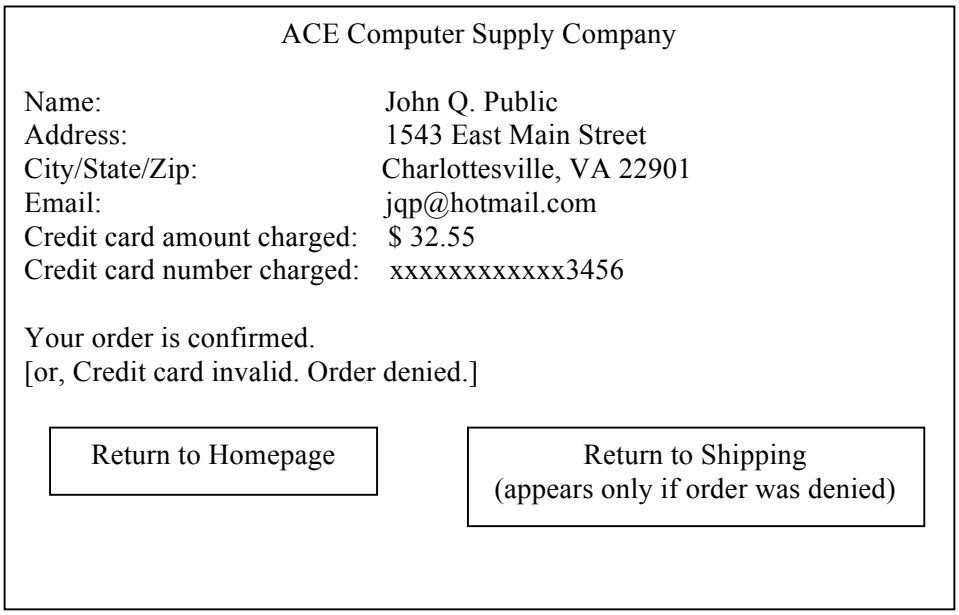
Email: jqp@hotmail.com

Credit card number: (16 digits) 1234567890123456

Return to Homepage Proceed to Payment

Client-side code verifies that the name, address, city/state/zip, and email textboxes are not null, and that the credit card number consists of exactly nine digits. (Note: we are using nine digits to simplify the verification step that follows.) If verification fails, alert the user to the type of error and allow him to correct it. Only when all verification checks pass will this page invoke the "Proceed to Payment" option. For simplicity, the only way to abandon the order is to return to the homepage.

9. *Payment*. When the user clicks on "Proceed to Payment", create a dynamic page that simulates checking the validity of the credit card. In reality, credit cards would be verified against an external database supplied by a credit card merchant. For our purposes, a credit card will be considered valid if the credit card number, taken as an integer, is a multiple of 103. This page checks the validity of the credit card number and produces one last page which confirms (by echoing) the shipping information and charge amount, and either confirms or denies the order. When displaying the credit card number, display only the last four digits and replace all other digits with x's. Example:



10. *Order Confirmation/Denial*. As shown in the example, credit card verification has two possible results. If the credit card is valid, the order is confirmed as shown, current inventory is reduced to reflect the items ordered, the navigation option is "Return to Homepage", and the user is sent an email that confirms that the order was processed. If the credit card is denied, then an appropriate error message is displayed, the order is denied, inventory is not updated, and the additional navigation button "Return to Shipping" is displayed. If the user clicks "Return to Shipping", redisplay the Shipping page and repopulate the boxes with the user's input.

11. *Database Update*. If the user now clicks on "Return to Homepage" and from there clicks on "Proceed to Order Form", the "Quantity Available" displayed must reflect the cumulative updates (reductions) due to all previous confirmed orders.

12. *Grading*.

Criteria 1-11 from above	*Value*
Database, timestamp, homepage	10
Order form, inventory verification	20
Checkout, return to order form	20
Shipping	20
Payment, order confirmation/denial	20
Database update (inventory reduction)	10
TOTAL	100

13. *Hints for a better project.*

If you want your project to be a competitor for the "best project" prize, extend the minimum requirements above in creative ways. Some ideas:

- add thumbnail pictures to the item description
- add a button that displays a user's past purchases
- creative graphic design and layout intended to create eye appeal
- allow users to provide comments about individual products, and make those user comments (attributed or anonymous) available to other potential purchasers for review
- recognize and acknowledge a returning customer
- personalize dynamically generated pages with information relevant to the customer

Again, the goal is to be creative and innovative, so these are merely suggestions.