**Project**

## E-Business Product Knowledgebase

In this assignment you’ll work build the front end to an e-business site that offers product information. The product area is for you to decide. While a complete online store would require quite a lot of additional functionality, you will only need to implement the basic ability to specify account information, view inventory and its information, and add new inventory via an administrative section.

An example of the minimum output requirements for this assignment can be found on the screen print attached however for your version the customer login does not need to be system-based.

## Design Requirements

Your e-business should have the capability to allow potential customers to view all of your products and if they wish to register, to create an account. While you will probably store all information using a database, you can also complete this project using a text file structure. Therefore, while the following requirements discussion is specific to using a relational database like MySQL, the basic requirements and principles presented will also hold true for those using text files. Text files are commonly referred to as a flat-file database when used to store structured data.

If you use the relational database approach, you will need to access the Zeus server and set up your MySQL database for use in your project. This database can initially be created at the IT&E Access Manager site (http://access.ite.gmu.edu) where some server-specific information is given supplemental to what is found in your textbook. For the file-based approach you may need to initially create the text files where data will be stored.

It is important that you take the time needed to model a normalized e-business database that encompasses all the components of your e-business system. Because site features will vary greatly from product area to product area, only general e-business requirements and process can be specified for the database portion of the project. Grading for this part of the assignment will differ from earlier assignments and will only involve a check off of certain components.

Once the database or text files are created, you will need to populate the tables/files with some initial product and category information.

## Design Overview and Approach

While the specific tables in your database or files will vary depending on what you are offering, the following list should get you thinking about all the pieces of information you may need to keep track of in your database:

* Product Information
* Customer Information
* Product Category Information
* Administrative Information
* Shipping and Handling Rate Information
* Product Manufacturer Information
* Product Review
* Product Feedback/Rating
* Search Engine Keyword List
* Administrator Information

Before touching MySQL or your text files, you should begin by diagramming the database on paper. Write down all the ‘things’ that you need to track. Examples of this would be the customer first name, customer last name, customer street address, product name, product retail price, product wholesale price, product weight, etc. Once you have a fairly comprehensive list, put these ‘things’ into different categories such as customers, products, etc. These categories will serve as the starting point for the tables/files that will make up your database. This however is only the starting point. If you simply make the tables/files based on these categories and make the fields the ‘things’ you have put in the categories, your database will not be normalized. A database that is not normalized could lead to inaccurate information later on in your database. To prevent this, you should follow normalization table-splitting or file-splitting steps. If you need more information on how to do this, your textbook and a simple search on the web on Normalization and Normal Forms will reveal a plethora of information.

Once you have normalized tables, you can implement these tables into MySQL. In addition to setting up the tables and columns in MySQL, don’t forget to set the attributes of the columns, tables, and relationships. If you are using a flat-file database/text file approach you will not need to do this step.

With this done, you should create a database diagram of all the tables/files, columns, relationships, etc that exist in your database and pertain to your project. The diagram should look similar to the diagrams found in your textbook. You should include these diagrams in your submission along with a discussion of the table/file design decisions you made. It should be noted that you have the flexibility as to how you want to make the diagrams.

Once you have designed your database, you’ll begin the design of your site with a way to present all of your product offerings. For this you will need a PHP file that will present menu choices to your visitors. While you can use any approach that you feel comfortable with you may want to use an approach that is modular. With this approach, you‘ll include each PHP file as needed (using the include() statement), based on menu choices made by the user. Grouping each operational component into a separate file will make it easier to debug or make changes to different aspects of your program. As such, the remaining requirements discussion will include this approach.

Start by designing a menu system for your e-business site. This system should contain the logic and control needed to allow the visitor to navigate your inventory and for the correct modules of code to be processed as needed. Your menu system should list all of the product categories by retrieving them from your database. In building your menu system, you should determine whether the visitor has logged into the site yet. If the user is logged in, you will show them links to edit their account information. If a user is not logged in, you will provide a means for them to log in or create an account as a potentially a new customer.

Once you have presented the user some menu choices, you should write the code to handle some of the menu choices that visitor might select. With the product categories, you can do this by determining whether a category identifier has been passed in the request, and if so, then you know that the user is browsing the product categories. If this is not the case, you will need to check whether they are requesting to create an account for your site as a potentially new customer. If this is so, you should allow them to do that by programming this option.

If the user has logged in, they may have selected the menu option to edit their information. After creating the section to add a new potential customer and while the customer information module is still fresh in your mind, you should go ahead and code the ability to edit the information the user entered. The programming of this section is similar to what you did in the new customer registration module. Now that the users have the ability to create an account and edit their information, you should move on to the next logical step, logging into the site. This should prompt a user for the user name and password they made up when they created an account. Similar logging in functionality should be added for the administrative portion of the site. You should program lower-case *admin* as the username and lower-case *admin* as the password for the administration portion.

With the ability to create, edit, and log in to an account out of the way, you will need to devise a way to list

the products that your company offers. The results should display all of the products in the database for the category your visitor selected. Finally, you'll want to be able to allow the user to select an individual product and view it. As such, you should program this feature also.

While not in the Coordinator's example site, you also need to implement a search engine into your project

and also an administrative portion where new products can be added. These new products should then

viewable in the customer portion of the site and also be searchable with the customer search engine.

There are no strict requirements for the search engine, however you may want to consider implementing

the following:

 Quick search - based on simple keyword(s)

 Advanced search - based on a combination of different attributes

 Category search - based on a predefined product category

Standard search engine flow usually corresponds to the following sequence:

Get required input data  Build SQL statement  Execute SQL statement

IF SUCCESS  Get results from database  Format result to output

IF FAILURE  Print error message

If your product tabe/file has many products, you should limit the number of results that are displayed per page.

For the main page of your site, you should use a file name that will automatically load when a user

specifies only the folder name in the URL. The file should be uploaded to a group member's Project directory. Your XHTML site should be well-formed and created using a strict XHTML DTD. You can

create the web page using any text editor (Microsoft Notepad, RogSoft Notepad+, Adobe Homesite, etc) that is not WYSIWYG capable (Adobe Dreamweaver, Microsoft Word, etc).

Once working like the output given, you can begin adding the CSS formatting. Finally, validate all the

pages and correct any ill-formed tags where necessary. You should use XHTML for the layout of all and

CSS for the formatting of each of the pages. Using an external .css file, add formatting of your choice to make the output more aesthetically pleasing.

**Submission**

Your assignment should be uploaded to a Project directory within one of your group member's

Zeus/Helios server space. Verify that the uploaded site works in the browser as this is what will be used for grading. Make sure that all group members can get to this site from their main IT207 page. It should not load in the *Assignment Content* frame of their home page. Make sure all links in the pages work. All members also need to submit the files to their individual appropriate Assignment drop box on Blackboard.

**Notables**

 Remember that the lab assignment must clearly display your group member names in the interface (if

any) and be commented with your names in all files.

 Try to be consistent in the naming of your files as this will help you later on.  Add appropriate comments to your code as we covered in lecture.

 Feel free to use your book and notes for general information on these technologies and concepts.

 For reference material on PHP, you may want to visit http://www.php.net/manual/en/ as this is a great

reference for being able to expand your abilities to solve problems beyond what we just cover in class.

 For reference material on XHTML, you may want to visit http://w3schools.com/xhtml/ which we

discussed in class.

 For reference material on CSS, you may want to visit http://w3schools.com/css/ which we discussed

in class.

 If need be, for reference material on HTML, you may want to visit http://w3schools.com/html/ which

we discussed in class.

 Just a reminder… this is group work so while you can work with one other student you should not

provide general assistance to other student groups or give solutions. The one exception is that general assistance can be