**SHEFFIELD HALLAM UNIVERSITY**

Faculty of Arts, Computing, Engineering and Sciences

Student Name: **PHINA ONYINYE ANYALECHI** Student Number: **26006046**

Course Title: **MSC. COMPUTER AND NETWORK ENGINEERING**

Module Title: **NETWORK APPLICATIONS**

Tutor: **MARCOS RODRIGUES** Module Code: **16-7203**

Assessment Aim: **TO DESIGN A WEB ENABLED DATABASE FOR A COMPANY THAT SELLS ELECTRONIC PRODUCTS ONLINE.**

Assessment Focus: **(LAMP) PLATFORM** Date Submitted: **APRIL, 2016**

**1.0 Introduction**

E-commerce simply put, is trading over the internet. It’s a virtual means of buying things in your comfort zone. The recent development in information and technology has drastically affected the way business is run today. E-stores are now trending because it gives people the opportunity to shop from any location. Today, virtually all companies, both small and big run an e-commerce site. A standard e-commerce site can be accomplished with a web enabled database. The website can then be referred to as a web based application, which consists of two pieces: Database and application (Valade, 2010).

**1.1 Statement of Purpose**

The purpose of this project is to design a web based application (e-commerce store) for a company that sells electronic products online. The website provides customers with a variety of electronic products grouped into categories which are made available after purchase (payment) has been made. The platform used in the development of this website is the LAMP programming. LAMP is an open source Web development platform that uses **Linux** as the operating system, **Apache** as the Web server, **MySQL** as the relational database management system (RDBMS) and **PHP** as the object-oriented scripting language (Gerner, et al., 2006). This report is based on the steps involved in designing a web based application. It covers four phases: the planning phase, the analysis phase, the design phase and implementation phase.

**1.2 Project Overview**

A web based application includes an administrator web interface and a customer web interface. The administrator interface is the interface that allows only administrators (admin) access to insert and remove database entries, query the database according to any chosen attribute, view statistics and customer preferences.

Whereas the customer web interface is the interface that allows customers to search the database under various attributes, place an order, inspect the status of ordered items, and review an item.

**1.3 Structure of the Web Based Application**

The structure of the web based application can be divided into two parts: the frontend and the backend as shown in **Figure 1.**

Backend

Frontend

Apache

PHP scripting

Http Requests

BROWSER

WEB SERVER

DATABASE

**Figure 1: Structure of the web based application.**

**1.3.1 The Frontend System**

The Frontend system is the user or customer web interface. It contains the application (programs) that performs the tasks requested by the user. The programs create the display which the customer sees in the browser, and they store information in the database and retrieve information from the database. The database is not a complete application without the frontend system, and the same applies to the frontend system. This means they cannot function alone. Making these two work together makes a website to be dynamic. The following functions can be performed in the frontend system:

* Search for items depending on category
* Order an item
* Add item to shopping cart
* Manage shopping cart
* Register
* Login and logout
* Pay for items
* Review an item

**1.3.2 The Backend System**

The Backend system is the administrative end of the webstore. The administrators (admin) has sole access when logged in. The database can be viewed, modified and can be deleted from this end. The admin performs two types of operations: e-store management and order management. E-store management involves keeping the e-store well stocked and up-to-date, while order management involves attending to the customer’s orders and getting them delivered. The following functions can be performed in the backend system:

* Login and logout
* Insert and remove database entries
* Query database according to any chosen attribute
* Perform management operations, like updating offers
* View orders and payments
* View customer reviews.

**Analysis**

**2.0 Background Research**

During the analysis of the webstore, various electronic webstores were visited to analyse their web applications. Webstores like Argos, Maplin, Currys PC world, etc. were helpful in the analysis of the customer interface. The Admin interface however, had to be analysed based on the operations it is expected to perform.

**2.1 Requirements Analysis**

System requirement is simply what a system is meant to do and the characteristics it is meant to have (Dennis, et al., 2012). The requirement analysis for the e-store involves eliciting, modelling and analysing various tools to meet the design specifications listed in the previous chapter. The system requirements are summarised in **Figure 2** using a proper use case diagram.

Search for products under categories

Place an order

View order and make changes

Pay for order

Review an item

Log on to user account or create new account

Insert new products

Update price and offers

Delete products from database

View orders and payments

View reviews

View available product stock

Database

Customer

Admin

Frontend

Backend

**Figure 2: Use case diagram of the system requirements**

Hereafter, each requirement was analysed to see the interrelations between them. The analysis was focused on capturing the business requirements for the webstore. Hence, certain questions were reviewed, such as;

* What is the technical feasibility of this site? This is the assessment of the technology and tools which will be used in the development of the webstore and the extent to which it can be deployed for customer usage. The webstore being designed is a small scale store, but has the capability of being expanded for a bigger market prospect.
* What if the admin is not a programmer, how will he access the database? The concept or structure of the webstore ensured that without the knowledge of programs used in the web design, the admin can work. By simply logging into the admin section (backend), admin can simply edit and modify the database.
* Of what advantage will the site be to the company? Considering customer preferences when visiting an e-store, the site incorporated into its design the bootstrap framework (this was downloaded from Google). This framework made the website to be responsive for users to access it even from their mobile phones.

These analysis aided in the development a proper design on how the system (webstore) will operate.

**Design**

**3.0 Design Considerations**

A successful analysis of the webstore led to the development of the design of the webstore. There are two system designs of the online store: the **presentation (user interface) system**, and the **data storage (SQL) system** design. A block diagram of the online store system design is shown in **Figure 4**.

**Online Store System**

**User Interface System**

**Data Storage System**

**Admin Interface**

**User Interface**

**Database**

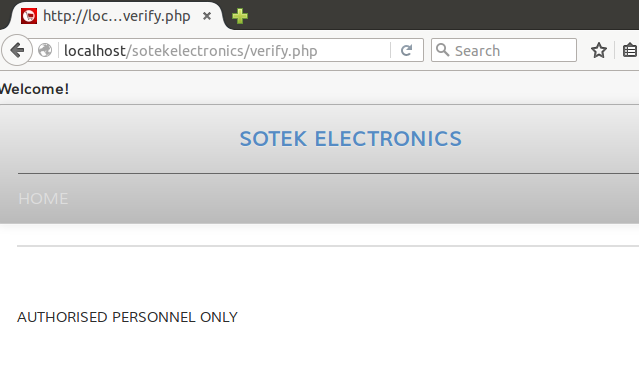
**Table A**

**Table B**

**FIGURE 4: Block diagram of the System design**

**3.1 The User Interface System Design**

The user interface system design focuses on maximising the user experience. It covers both the **Customer interface** and the **Admin interface**. The use of graphics in the design supports its usability and influences the way the user performs and interacts on the site. It also makes the webstore appealing to users. Considerations were given to the ability of the system interacting effectively with the user by sending appropriate feedbacks, as shown in **Figure 5,** where a customer tried to login to the admin interface.

**FIGURE 5: Example of an appropriate feedback**

The structure diagrams of the customer interface and the admin interface is shown in **Figure 6** and **Figure 7** respectively.

**Customer Homepage**

**Login/Register**

**Featured Products**

**User Account**

**View Product(s)**

**Search Categories**

**View Television & acc.**

**View Laptops**

**Contact Us**

**View Headphones**

**Add to Cart**

**View Home Cinema**

**Update Cart**

**Continue Shopping**

**Checkout**

**View Order(s)**

**Logout**

**Cancel Order(s)**

**Review Product(s)**

**Make Payment**

**FIGURE 6: Structure diagram of the customer interface**

**3.1.1 The Customer Interface**

Within the customer interface, which can be called the frontend, customers can view products (in general or according to categories). Then, add these products to a shopping cart which can be updated or cancelled. The shopping cart can be likened to a physical shopping cart, which stores all selected items from a store. If such customers choose to buy these products, then, they will need an account.

For existing customers, they will have to login to their accounts and access their orders, make changes, payments, review items and eventually logout. For new customers, they will have to register or create an account before they can make purchases or leave reviews. The need for this registration is for the customer contact to be recorded in the database. The contacts will eventually be seen by the admin, who will make delivery of the purchased items to their addresses.

For customers who choose to pay a visit to the local shops, they can find the company contact on the ‘Contact Us’ page. With a system like this, it will be easy for the customer to use and efficient for the company admin to manage. Shown in **Figure 7** is the admin interface.

**Admin Homepage**

**Logout**

**View Customer Order**

**Edit Product(s)**

**Edit Category**

**Admin Menu**

**Add/Delete Product(s)**

**Add/Delete Category**

**View Product Details**

**View Payments**

**FIGURE 7: Structure diagram of the admin interface**

**3.1.2 The Admin Interface**

Within the admin interface, which can be called the backend, the admin personnel can perform various tasks. The admin can insert and remove a product, insert and remove product categories. Query the database to view product details, keeping record of available stock. If stock happens to be limited, admin can update product information, such as the pricing and product stock. He can also view customer details, such that he can tell the orders made by which customer, payments made and the customers’ address to make delivery of purchased item.

Designing a good, user friendly and consistent user interface is ideal for every good webstore. User interface design is an art. According to (Dennis, et al., 2012), there are certain principles for effective user interface design:

* Layout – certain areas of the screen should be used for different purposes. Hence, the webstore was divided into: header, navigation bar, sidebar, the main body and the footer.
* Content awareness – simply put, the user or admin should easily identify with the page they are on.
* Aesthetics – involves careful use of space, colours and font to make the page inviting.
* User experience – it should be easy to use
* Consistency and minimum user effort – consistency makes the system predictable and easy to use, reducing user effort to perform an operation.

These principles were put into consideration for the user interface designs.

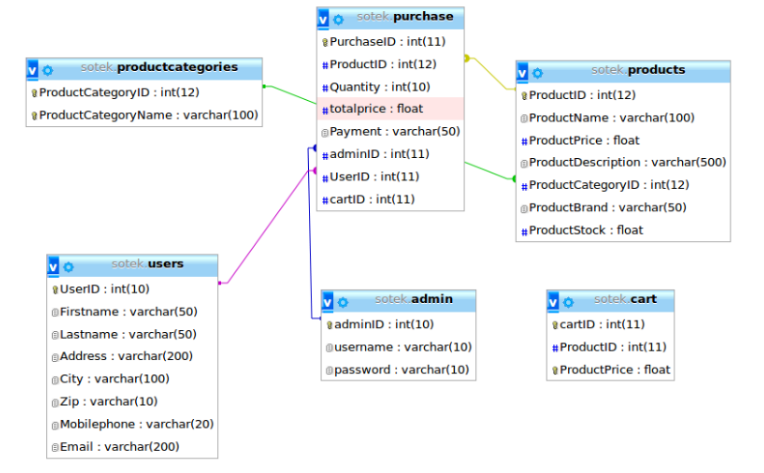
**3.2 Data Storage System Design**

The data storage system as the name implies is where the data is stored, which is the database. A database is a collection of grouped data (Suehring, et al., 2009). For a functional webstore, a database is important. It is also a tool that lets you store information, grab that information when needed, and organize the information you are storing (McLaughlin, 2012). It can be likened to your cabinet, where you store things, to retrieve them when required. A good database stores information for a long time, or even permanently (if not edited).

Every Database has a structure, and the good ones are relational. So this means that you not only store your data in a database, but also tell the database how each information is related to the other piece of stored information, and how it should be constructed. According to (Suehring, et al., 2009), database design can be broken into:

* One-to-one data which can be likened to a unique identifier such as a students’ registration number.
* One-to-many data and many-to-one data which can be portrayed as a teacher-to-students relationship or students-to-teacher relationship.
* Many-to-many data which can be represented by the relationship of programmers to software applications.

For the database of our web based application, MySQL which is a relational database management system (RDBMS) is used to create the relationship between the tables in the database system. With the use of a unique identifier (primary key) present in every table, each table is related by connecting these primary keys as a foreign key to other tables. This is shown in the entity relationship diagram (ERD) in **Figure 8.**

**FIGURE 8: Entity Relationship Diagram of the Database**

**Implementation and Testing**

**4.0 Overview of the Development Environment**

The proposed platform for the development of the web application is the LAMP platform. This is an open source web development platform that uses Linux as its operating system, Apache as its HTTP server, MySQL as its RDBMS, and PHP as its programming language. These components are the technologies used in the implementation of the system design.

**4.1 Implementation Technologies**

**4.1.1 Linux**

The Ubuntu 15.10 was installed as the Linux open source operating system. It was downloaded from its official website and installed on VMware Workstation. Then, a Virtual machine was created on the VMware Workstation. The installation process was followed with prompts until completed. The details of the username and password are:

* Phina Anyalechi
* para2509

**4.1.2 PHP**

PHP is a scripting language designed specifically for use on the web, make dynamic websites. It is an acronym for PHP: HyperText Preprocessor. It is an open source software which was easily installed into the Linux operating system.

**4.1.3 MySQL and PHP**

These two are a perfect pair and are frequently used together. MySQL provides the database and the PHP provides the application part of the Web database application (Valade, 2010). For our web enabled database PhpMyAdmin which is an open source web interface tool used to manage MySQL database was implemented. It was installed easily into the VMware Workstation. To check for successful installation, open the browser on the virtual machine, and type in **http:/127.0.0.1/phpmyadmin/** or **http:/localhost/phpmyadmin**. The browser should display the web tool asking for a username and password. The username and password for this web database is, ‘root’, ‘root’ respectively.

**4.1.4 Apache**

Apache is also an open source software. The installation was simple. Go to xterm on Ubuntu, and type in **sudo apt-get install apache2.** To check if it is working, open the browser on Ubuntu and type in **http:/localhost/** and a page will display with the words, ‘It works!’

**4.1.5 HTML**

This stands for HyperText Markup Language. It is a markup language used for creating webpages. It creates the texts and performs most text formatting which is displayed on the user interface. It creates static webpages, as they are not interactive without PHP or MySQL. The combination of these programming languages, created a dynamic and interactive web page.

**4.1.6 CSS**

This stand for Cascading Style Sheet. It was implemented in this web application development as it enhances the text fonts, background themes and basically the style of the web page. It puts colour and makes the web page look attractive to users. The CSS implemented in the styling of the HTML of our web application was downloaded from Google.

**4.1.7 Bootstrap Framework**

This framework enhances the frontend and makes the web page responsive. It was also downloaded from Google, for the enhancement of the webpage, making it look attractive.

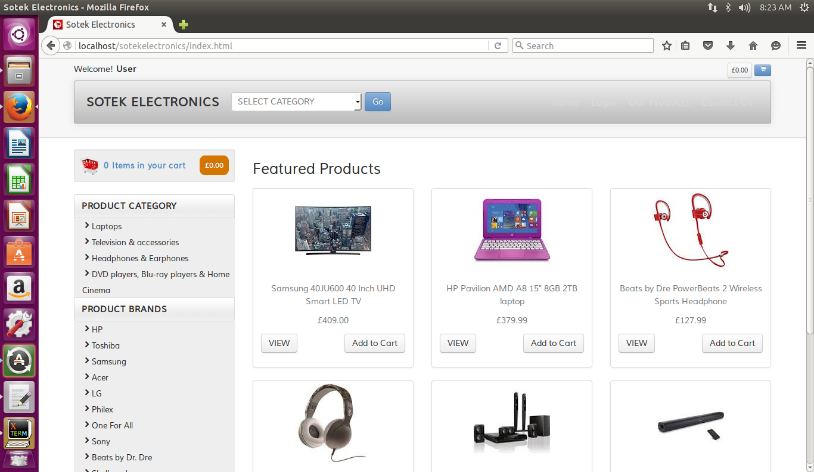
**4.2 Implementation and Testing**

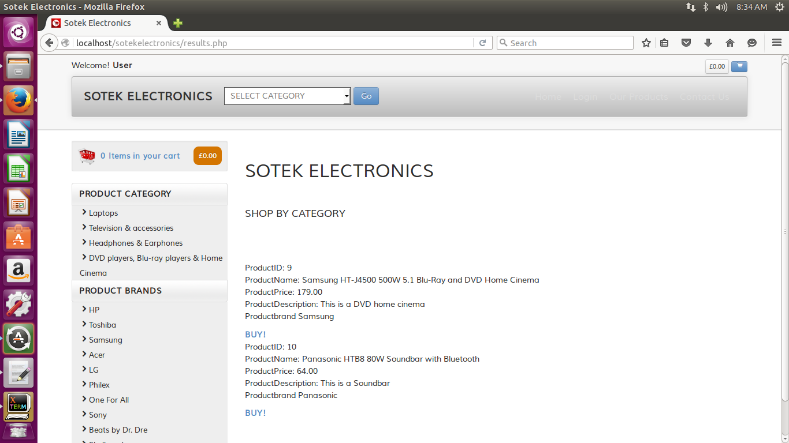
**4.2.1 Creating the Database**

The database was created using the PhpMyAdmin. Then tables were also created for different purposes and assigned primary keys to identify them. These primary keys were used to create the relationship as shown in **figure 8** between the tables.

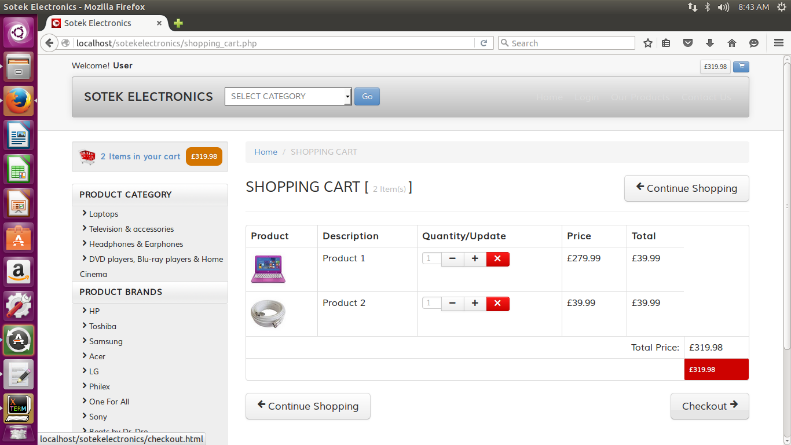
**4.2.2 Creating the User Interface**

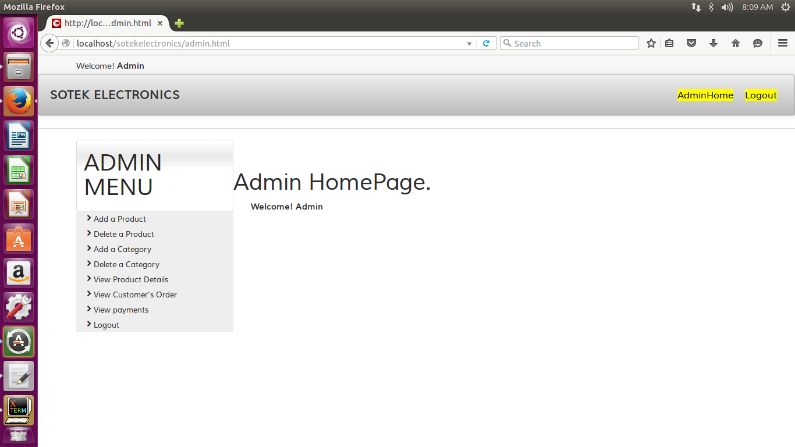
The user interface was created using PHP, HTML, the bootstrap framework and CSS. The source codes can be found in the appendix. Below are some screenshots of the user interface designs considering the principles according to (Dennis, et al., 2012).



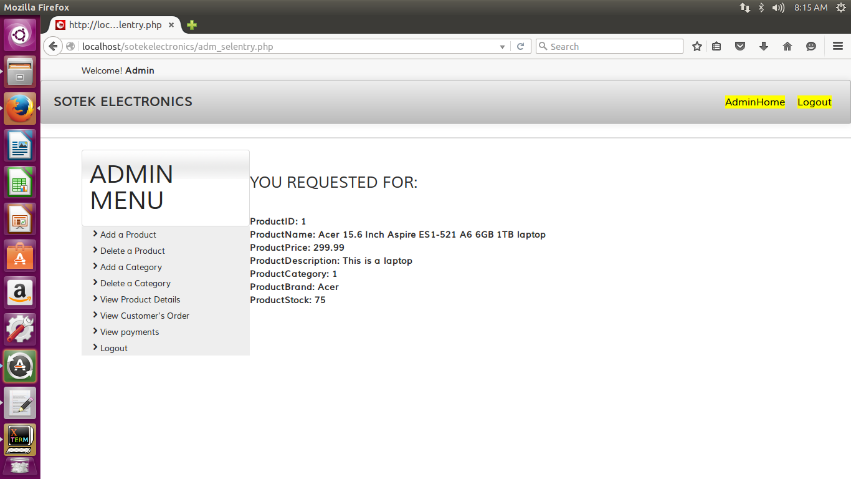
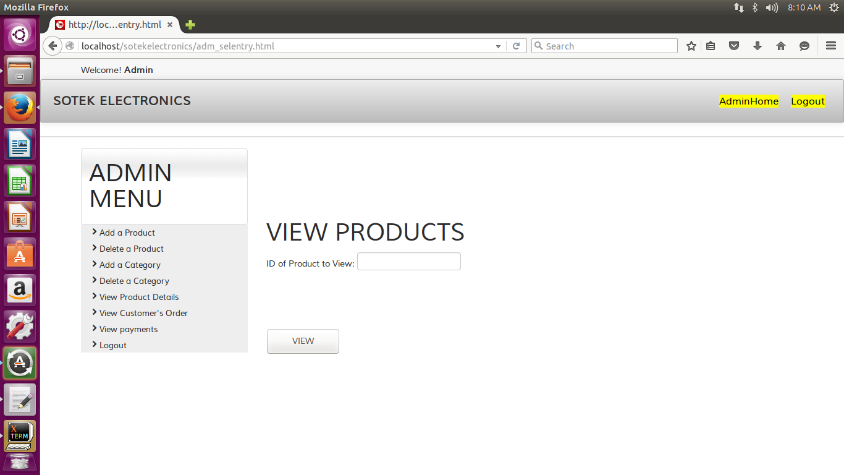
**FIGURE 9: Customer Homepage**

**FIGURE 10: Search Category Result**

****

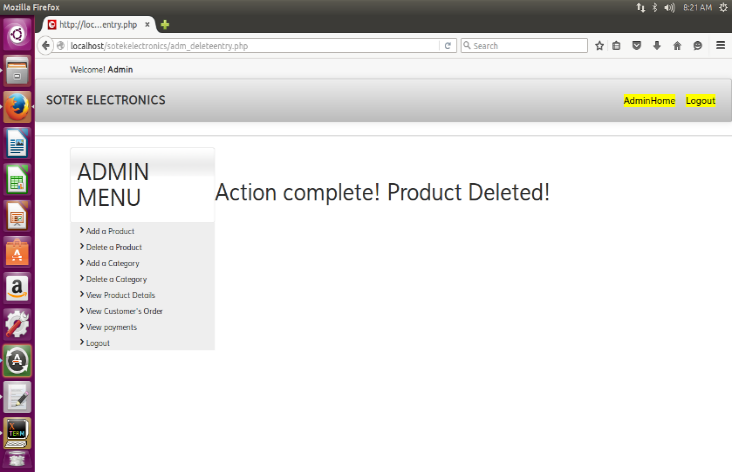
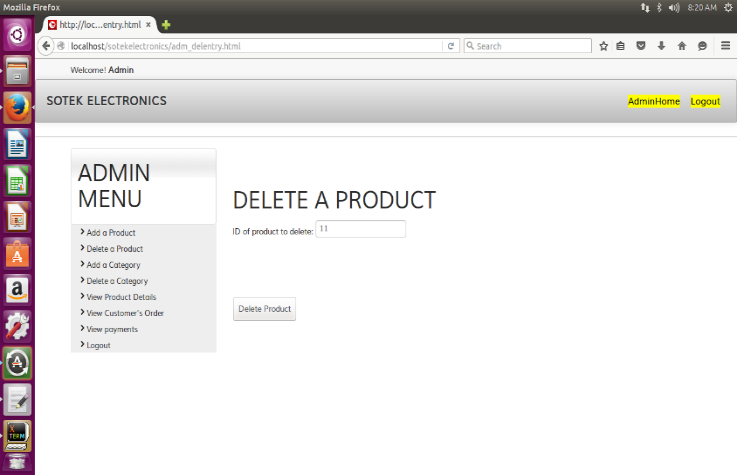
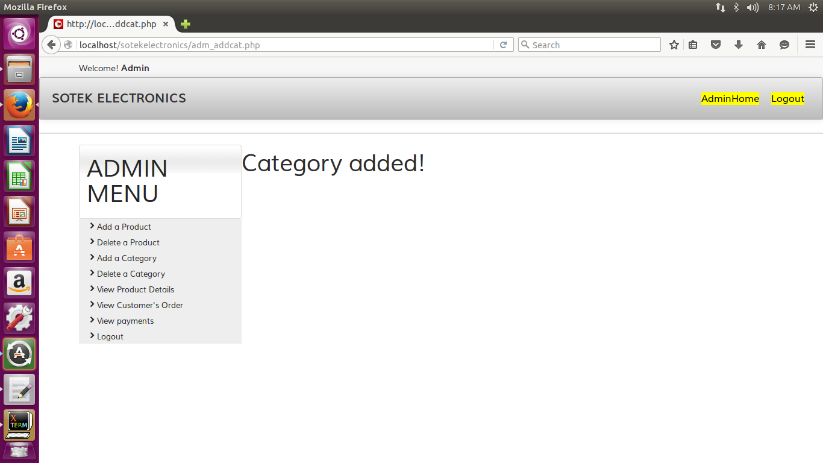
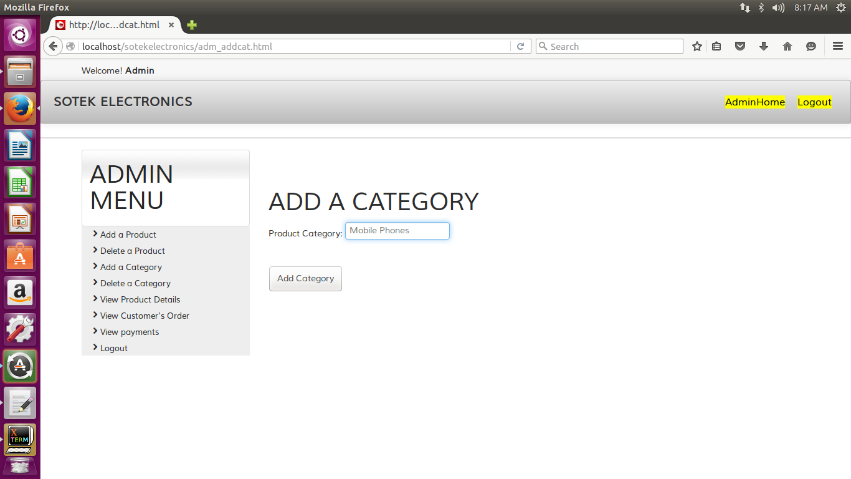
**FIGURE 11: Shopping Cart Page**

**FIGURE 12: Admin Homepage**



**FIGURE 13: View Products request and result page**

**FIGURE 14: Add category request and result page**

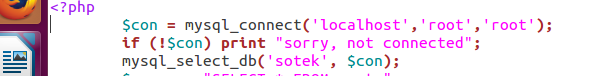


**FIGURE 15: Delete a Product request and result page**

**4.3 Implementation of PHP and MySQL Using PHPMyAdmin**

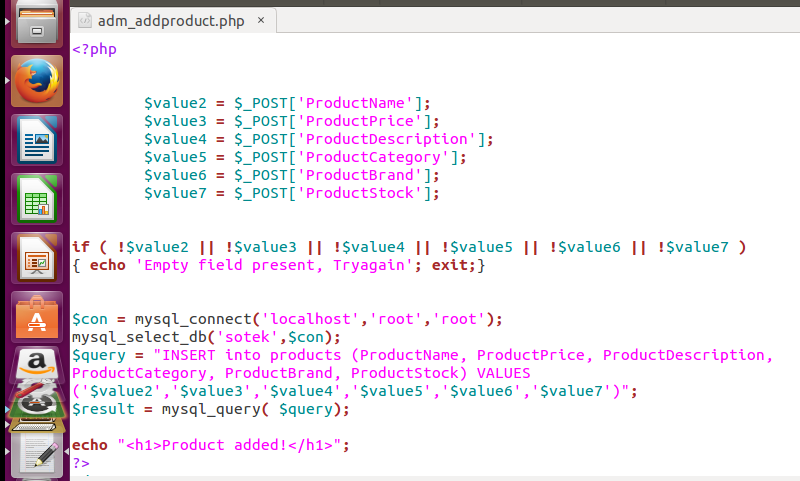
**4.3.1 Connecting to the Database**

The database containing all the data was stored in MySQL, and PHP language was used to write programs to perform application tasks. PHP scripts or commands are used for tasks such as displaying a webpage, accepting or rejecting a data in a HTML form, etc. The designed web enabled database had a lot of tasks to be performed, hence, different programs were written for the various tasks. In the figures below are some of the PHP programs and the operations they performed.

****

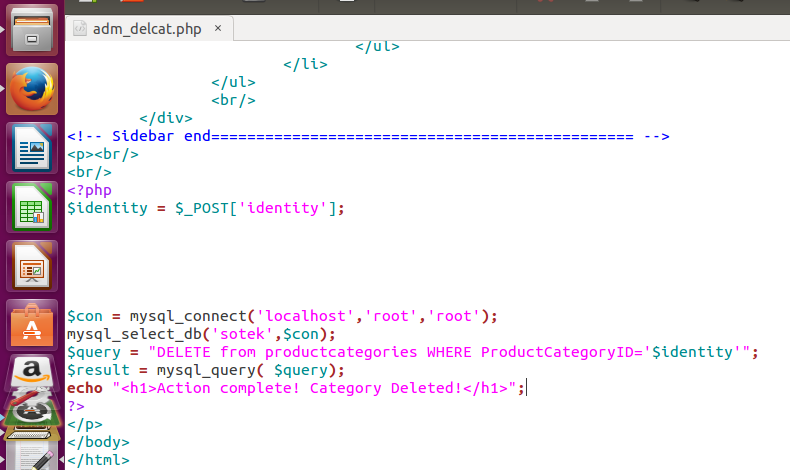
**FIGURE 16: Connection to the database**

This is the code used to open a connection to the MySQL Server. The first line contains (‘server’, ‘username’, ‘password’) to the Server. Then the third line connects user to the database. Hence, all applications requesting data from the database or sending data to it, needs to establish this connection. Hence, most web developers create such a file and store it as a connection file and ‘include’ it into other programs or codes. Thereby, eliminating redundancy.

****

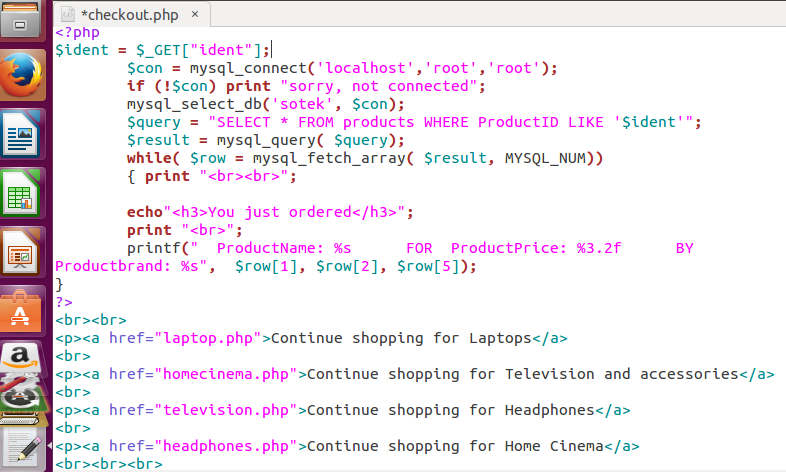
**FIGURE 17: Add Product Query**

This code is used to add products to the ‘products’ table. The $\_POST method is used to send data to the database (Meloni, 2008). Hence this code can also be to use in any adding data operation, such as add a category to category table, etc.

****

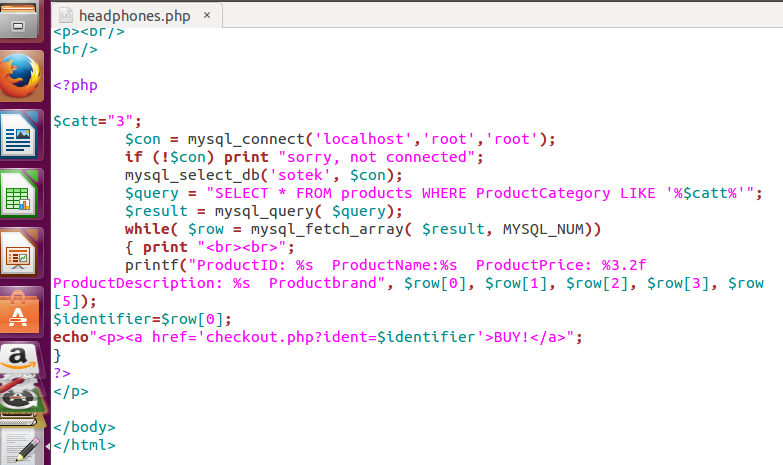
**FIGURE 18: Delete Category Query**

This query deletes from the database, and can also be used for any deleting operations. The reserved word being ‘DELETE’.

****

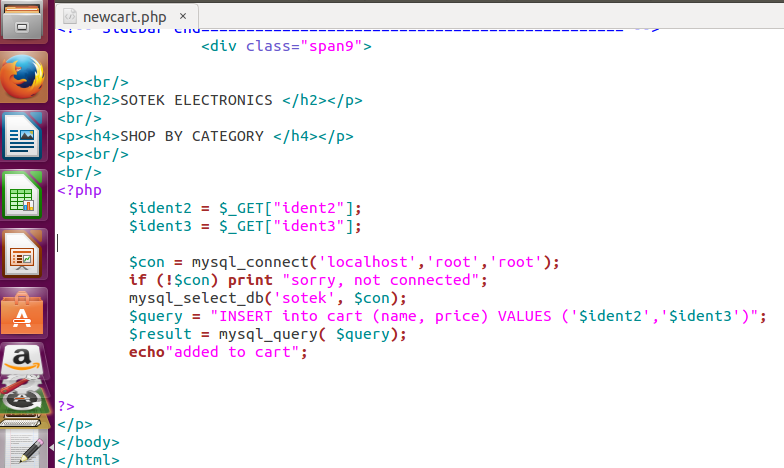
**FIGURE 19: Checkout Query**

This query was used for checking out a customer, after they have ordered a product from the store. The PHP query, after successful execution, sends an echo message telling customer what he just ordered.

****

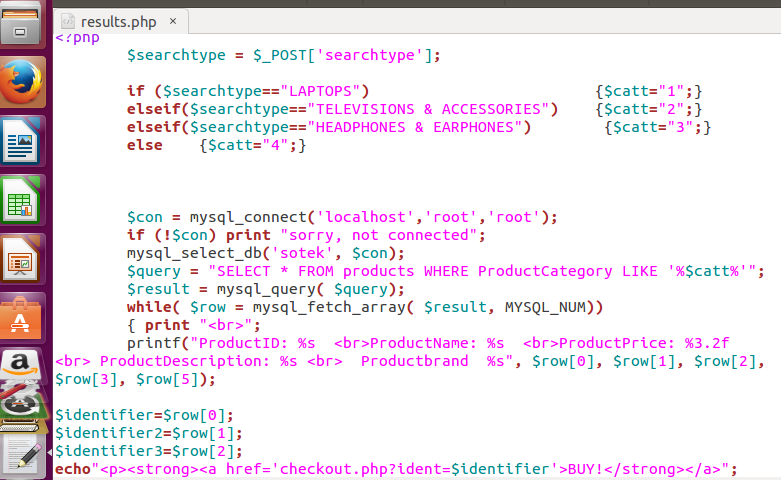
**FIGURE 20: Headphone and Earphone category Query**

This query display the results for all the products in ‘$catt=”3”’. Showing the customer the product ID, name, brand, price and description. Also giving the customer the option to ‘BUY!’. Choosing to BUY, will take customer to the ‘checkout.php’ page for further shopping experience.

****

**FIGURE 21: New Cart Query**

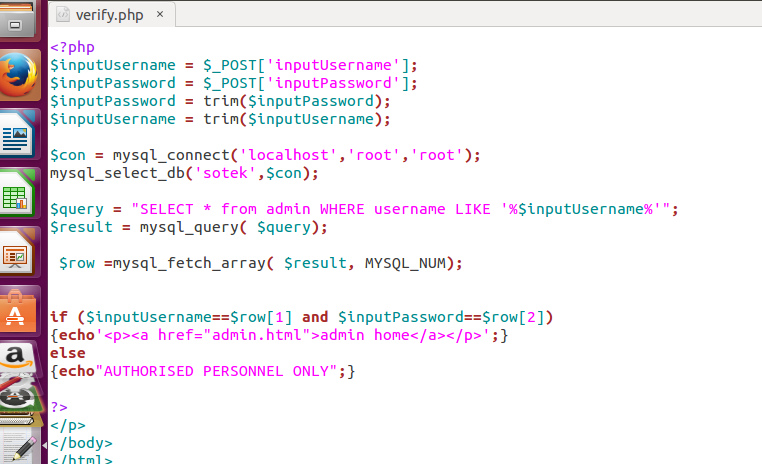
This query sends the product information to the cart table in the database, using the $\_GET method. Hence, the admin will easily access what products are being purchased.

****

**FIGURE 22: The Search Query**

This query searches the database for items or data according to the request of the query. In this case, the query is requesting the products that are in each category. The result for this query will depend on the value of the ‘$searchtype’. The results displayed will be the list of products for a category with the option for the customer to ‘BUY!’ such product.

**4.3.2 Securing the Admin Interface**

****

**FIGURE 23: Verification Query**

This was used to secure the admin page from customers. A wrong input of the username and password means it’s not accessed by an admin, hence a message will be sent saying, ‘AUTHORISED PERSONNEL ONLY’. This is one of the examples of a proper feedback mechanism, which was a consideration in the web application design.

**Evaluation**

The web enabled application was successfully implemented. For a standard web interface, using just the LAMP platform will not create an attractive website for customers. Technologies such as CSS and Bootstrap Framework which is available for free online, will create that enhancement which a standard web interface requires.

**Conclusion**

The four phases of this project were deemed successful. The planning, analysis, designing and implementation required research which was carried out to get these results. A lot of functionalities were omitted in this site due to insufficient programming knowledge. Some queries didn’t function properly, but the basic functionality of selling goods and recording them in the database was met. The opportunity of developing this webstore exposed me to the world of programming (which is interesting).

If I had more time, I would have met all of the requirements and probably exceeded it, as there are a lot of functions which a standard ecommerce site provides its customers and administrator.

# Bibliography

Dennis, A., Wixom, B. H. & Roth, R., 2012. *System Analysis and Design.* 5th ed. New Jersey: John Wiley & Sons, Inc..

Gerner, J., Naramore, E., Owens, M. L. & Warden, M., 2006. *Professional LAMP: Linux, Apache, MySQL and PHP5 Web Development.* 2nd ed. Indiana: Wiley Publishing, Inc.,.

McLaughlin, B., 2012. *PHP and MySQL: The Missing Manual.* 1st ed. Cambridge: O'Reilly Media, Inc.,.

Meloni, J. C., 2008. *Sams Teach Yourself PHP, MySQL and Apache All in One.* 4th ed. Indiana: Sams Publishing.

Suehring, S., Converse, T. & Park, J., 2009. *PHP 6 and MySQL 6 Bible.* 1st ed. Indiana: Wiley Publishing, Inc.,.

Valade, J., 2010. *PHP & MySQL For Dummies.* 4th ed. Indiana: Wiley Publishing, Inc.,.

**APPENDIX**

**Add product to Database**

<?php

$value2 = $\_POST['Firstname'];

$value3 = $\_POST['Lastname'];

$value4 = $\_POST['Address'];

$value5 = $\_POST['City'];

$value6 = $\_POST['Zip'];

$value7 = $\_POST['Mobilephone'];

$value8 = $\_POST['Email'];

if ( !$value2 || !$value3 || !$value4 || !$value5 || !$value6 || !$value7 || !$value8 )

{ echo 'Empty field present, Tryagain'; exit;}

$con = mysql\_connect('localhost','root','root');

mysql\_select\_db('sotek',$con);

$query = "INSERT into users (Firstname, Lastname, Address, City, Zip, Mobilephone, Email) VALUES ('$value2','$value3','$value4','$value5','$value6','$value7','$value8')";

$result = mysql\_query( $query);

echo "<h1>Entry added!</h1>";

?>

</p>

**Admin Login.**

<div class="span9"> <ul class="breadcrumb"> <li><a href="index.html">Home</a> <span class="divider">/</span></li> <li class="active">Login</li> </ul> <h3> Login</h3> <div class="span4"> <div class="well"> <h5>ADMIN LOGIN</h5><form name="Form" action="verify.php" method="post"><p> username: <input type="text" name="inputUsername" size="20" style="width: 1.75in; height: 0.3in"> </p> <p><br/>

<br/> </p> <p>password: <input type="password" name="inputPassword" size="20" style="width: 1.75in; height: 0.3in">

</p>

<p><br/>

<br/> <p><br/><input type="submit" value="log in" style="width: 1.24in; height: 0.43in"> </p></form>

</div>

</div>

</div>

</div>

</div></div>

</div>

**Select a Product**

<?php

$identity = $\_POST['identity'];

$con = mysql\_connect('localhost','root','root');

mysql\_select\_db('sotek',$con);

$query = "SELECT \* from products WHERE ProductID = '$identity'";

$result = mysql\_query( $query);

while( $row =mysql\_fetch\_array( $result, MYSQL\_NUM))

{ print '<br>';

printf("<strong>ProductID: %s <br>ProductName: %s <br> ProductPrice: %3.2f <br> ProductDescription: %s <br> ProductCategory: %s <br>ProductBrand: %s <br>ProductStock: %s</strong>", $row[0], $row[1], $row[2], $row[3],$row[4],$row[5],$row[6]);

}

?>

</p>

**Add a Category**

<?php

$value2 = $\_POST['ProductCategoryName'];

if ( !$value2)

{ echo 'Empty field present, Tryagain'; exit;}

$con = mysql\_connect('localhost','root','root');

mysql\_select\_db('sotek',$con);

$query = "INSERT into productcategories (ProductCategoryName) VALUES ('$value2')";

$result = mysql\_query( $query);

echo "<h1>Category added!</h1>";

?>

</p>

</body>

**Delete a Category**

<p><br/>

<br/>

<?php

$identity = $\_POST['identity'];

$con = mysql\_connect('localhost','root','root');

mysql\_select\_db('sotek',$con);

$query = "DELETE from productcategories WHERE ProductCategoryID='$identity'";

$result = mysql\_query( $query);

echo "<h1>Action complete! Category Deleted!</h1>";

?>

</p>

</body>

**Checkout**

<div class="span9">

<p><h1>Sotek Electronics Checkout</h1></p>

<p><br/>

<?php

$ident = $\_GET["ident"];

$con = mysql\_connect('localhost','root','root');

if (!$con) print "sorry, not connected";

mysql\_select\_db('sotek', $con);

$query = "SELECT \* FROM products WHERE ProductID LIKE '$ident'";

$result = mysql\_query( $query);

while( $row = mysql\_fetch\_array( $result, MYSQL\_NUM))

{ print "<br><br>";

echo"<h3>You just ordered</h3>";

print "<br>";

printf(" ProductName: %s FOR ProductPrice: %3.2f BY Productbrand: %s", $row[1], $row[2], $row[5]);

}

?>

<br><br>

<p><a href="laptop.php">Continue shopping for Laptops</a>

<br>

<p><a href="homecinema.php">Continue shopping for Television and accessories</a>

<br>

<p><a href="television.php">Continue shopping for Headphones</a>

<br>

<p><a href="headphones.php">Continue shopping for Home Cinema</a>

<br><br><br>

<p><a href="index.html">HOME</a>

</p>

</body>

</html>

**Delete Cart**

<?php

$ident5 = $\_GET["ident5"];

$con = mysql\_connect('localhost','root','root');

mysql\_select\_db('sotek',$con);

$query = "DELETE from cart WHERE name ='$ident5'";

$result = mysql\_query( $query);

echo "<h1>Entry deleted!</h1>";

?>

</p>

**Headphone Category**

<p><h4>HEADPHONES AND EARPHONES CATEGORY </h4></p>

<p><br/>

<br/>

<?php

$catt="3";

$con = mysql\_connect('localhost','root','root');

if (!$con) print "sorry, not connected";

mysql\_select\_db('sotek', $con);

$query = "SELECT \* FROM products WHERE ProductCategory LIKE '%$catt%'";

$result = mysql\_query( $query);

while( $row = mysql\_fetch\_array( $result, MYSQL\_NUM))

{ print "<br><br>";

printf("ProductID: %s ProductName:%s ProductPrice: %3.2f ProductDescription: %s Productbrand", $row[0], $row[1], $row[2], $row[3], $row[5]);

$identifier=$row[0];

echo"<p><a href='checkout.php?ident=$identifier'>BUY!</a>";

}

?>

</p>

</body>

**Home Cinema Category**

<p><h2>SOTEK ELECTRONICS </h2></p>

<br/>

<p><h4>DVD PLAYERS, BLU-RAY PLAYERS & HOME CINEMA CATEGORY </h4></p>

<p><br/>

<br/>

<?php

$catt="4";

$con = mysql\_connect('localhost','root','root');

if (!$con) print "sorry, not connected";

mysql\_select\_db('sotek', $con);

$query = "SELECT \* FROM products WHERE ProductCategory LIKE '%$catt%'";

$result = mysql\_query( $query);

while( $row = mysql\_fetch\_array( $result, MYSQL\_NUM))

{ print "<br><br>";

printf("ProductID: %s ProductName:%s ProductPrice: %3.2f ProductDescription: %s Productbrand %s", $row[0], $row[1], $row[2], $row[3], $row[5]);

$identifier=$row[0];

echo"<p><a href='checkout.php?ident=$identifier'>BUY!</a>";

}

?>

</p>

</body>

**Laptop Category**

<p><br/>

<p><h2>SOTEK ELECTRONICS </h2></p>

<br/>

<p><h4>LAPTOPS CATEGORY </h4></p>

<p><br/>

<br/>

<?php

$catt="1";

$con = mysql\_connect('localhost','root','root');

if (!$con) print "sorry, not connected";

mysql\_select\_db('sotek', $con);

$query = "SELECT \* FROM products WHERE ProductCategory LIKE '%$catt%'";

$result = mysql\_query( $query);

while( $row = mysql\_fetch\_array( $result, MYSQL\_NUM))

{ print "<br><br>";

printf("ProductID: %s ProductName:%s ProductPrice: %3.2f ProductDescription: %s Productbrand", $row[0], $row[1], $row[2], $row[3], $row[5]);

$identifier=$row[0];

echo"<p><a href='checkout.php?ident=$identifier'>BUY!</a>";

}

?>

**View/ modify Cart**

<?php

$con = mysql\_connect('localhost','root','root');

if (!$con) print "sorry, not connected";

mysql\_select\_db('sotek', $con);

$query = "SELECT \* FROM cart ";

$result = mysql\_query( $query);

while( $row = mysql\_fetch\_array( $result, MYSQL\_NUM))

{ print "<br>";

printf("name %s <br> price %3.2f", $row[0], $row[1]);

$identifier5=$row[0];

echo"<p><strong><a href='deletecart.php?ident5=$identifier5'>remove item</strong></a>";

}

?>

<p><strong><a href='checkout.html'>checkout</strong></a>

</p>

</body>

**Admin Verification**

<?php

$inputUsername = $\_POST['inputUsername'];

$inputPassword = $\_POST['inputPassword'];

$inputPassword = trim($inputPassword);

$inputUsername = trim($inputUsername);

$con = mysql\_connect('localhost','root','root');

mysql\_select\_db('sotek',$con);

$query = "SELECT \* from admin WHERE username LIKE '%$inputUsername%'";

$result = mysql\_query( $query);

$row =mysql\_fetch\_array( $result, MYSQL\_NUM);

if ($inputUsername==$row[1] and $inputPassword==$row[2])

{echo'<p><a href="admin.html">admin home</a></p>';}

else

{echo"AUTHORISED PERSONNEL ONLY";}

?>

</p>

</body>

**Add to Cart**

<p><h2>SOTEK ELECTRONICS </h2></p>

<br/>

<p><h4>SHOP BY CATEGORY </h4></p>

<p><br/>

<br/>

<?php

$ident2 = $\_GET["ident2"];

$ident3 = $\_GET["ident3"];

$con = mysql\_connect('localhost','root','root');

if (!$con) print "sorry, not connected";

mysql\_select\_db('sotek', $con);

$query = "INSERT into cart (name, price) VALUES ('$ident2','$ident3')";

$result = mysql\_query( $query);

echo"added to cart";

?>

</p>