

DESIGNING AN E-COMMERCE WEBSITE USING PHP

Project Report submitted in partial fulfillment of the requirement for
the degree of

Bachelor of Technology.

in

Computer Science & Engineering

under the Supervision of

Ms Nishta Ahuja



Jaypee University of Information and Technology Waknaghat, Solan
– 173234, Himachal Pradesh

Date: 24-12-2014

Name of the student

SANYOG SHARMA

Certificate

This is to certify that project report entitled “Desiging an E-commerce website using Php”, submitted by Sanyog Sharma in partial fulfillment for the award of degree of Bachelor of Technology in Computer Science & Engineering to Jaypee University of Information Technology, Wahnaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

Date: 24-12-2014

Supervisor’s Name

Nishta Ahuja

Designation

Acknowledgement

The project entitled as Designing an E-commerce website using Php. The main purpose of this project is to create a e-commerce website that is user friendly,cheap and easy to use. This project is the result of my endurance in best of ways for these couple of months.

My project guide,Ms Nishta Ahuja an Asstt. Professor for Department of Computer Science & IT at Jaypee University of Information Technology, has indeed guided me to proceed with this project in best of her capability and enlightened me with her knowledge and wisdom. She has given me the push that I needed to go forward and do more. So I take this opportunity to thank her for his lending help and cooperation. May she shower such wisdom and guidance to future individual.

Date: 24th Dec,2014

Name of the student: Sanyog Sharma(101266)

\.

CONTENT

1. Abstract.....	6
2. Chapter 1.....	7
1.1 Introduction.....	7

1.2 What PHP and MySQL Can Do.....	7
1.3 What is PHP like?.....	7
1.4 What does PHP stands for?.....	7
1.5 PHP comes in two flavours local and remote.....	8
1.6 HTML And CSS Run within a Browser.....	8
1.7 Javascript add complexity but not software.....	8
2. Advantages of PHP.....	11
3. Challenges.....	11
4. Motivation.....	12
5.Problem Statement.....	13
3. Chapter 2.....	14
2.1 PHP is not part of your browser.....	14
2.2 Write Anywhere, Run when there is PHP.....	16
2.3 Determination by extension.....	18
2.4 Javascript basic introduction.....	19
2.5 Function.....	19
2.6 JQuery.....	19
2.7 Creating Confrigation files and rendering the main page	20
2.8 Snapshots.....	26
3. CONCLUSION.....	28

List of Figures

S.No.	Title	Page No.
1.	Web browser and HTML Integration	8
2.	HTML And CSS Rendering	9
3.	How Javascript fits in	11
4.	How PHP works	15
5.	Snapshot of wamp server	17
6.	Snapshot of PHP admin	18
7.	Snapshot of admin Page	25
8.	Snapshot of registration Page	26
9.	Snapshot of Homepage	26

ABSTRACT

In this project, we are going to build an e-commerce system integrating latest technologies to real-world development scenarios as much as possible, by setting up a full-fledged Online Shoe Storefront for a company named Soccus in the footwear business. The main objective of doing this project is not only how to build a solid e-commerce infrastructure but also the actual business aspects of the system. The website should be cost efficient, easy to use and user can compare items based on his choice.

CHAPTER 1

1. INTRODUCTION

1.1 What PHP and MySQL Can Do?

PHP can handle payment processing on its own, and it can connect with services like PayPal and Google Checkout. PHP can store and load images from a database or a file system and give you the ability to log users in and out as well as control what they see throughout your application.

Add in MySQL, and you can store your users' names, addresses, billing data, and even their preferences regarding the color of their own personal landing page. MySQL can store just a few bits of data, a few thousand lines of data, or every page access by every user who ever logs into your application.

And, of course, PHP can easily connect to MySQL. PHP can do everything from grabbing a user name based on a user ID to storing the details about financial transactions to actually creating tables and updating their structures, and MySQL can back-end all that work and store that data. Ultimately, this is the stuff of web applications; it's what a web application is.

Obviously, web applications like this aren't simple. They have a lot of complexity, and that complexity has to be managed and ultimately tamed into a usable, sensible web application that you can maintain and your users can enjoy. That's what this book is about: building web applications, and doing it with an understanding of what you're doing, and why you're doing it.

1.3 What Is PHP Like?

PHP is a programming language. It's like JavaScript in that you spend most of your time dealing with values and making decisions about which path through your code should be followed at any given time. But it's like HTML in that you deal with output—tags that your users view through the lens of their web browsers. In fact, PHP in the context of web programming is a bit of a mutt; it does lots of things pretty well, rather than just doing one single thing.

1.4 What does PHP stand for?

PHP is an acronym. Originally, it stood for Personal Home Page Construction Kit, because lots of programmers used it to build their websites, going much further than what was possible with HTML, CSS, and JavaScript. But in the last few years, “personal home page” tends to sound more like something that happens on one of those really cheap hosting sites, rather than a highpowered programming language. So now, PHP stands for PHP: Hypertext Preprocessor. If that sounds geeky, it is. In fact, it's a bit of a programmer joke: PHP stands for something that actually contains PHP within itself. That makes it a recursive acronym, meaning that it references itself. You don't have to know what a recursive acronym is; that won't be on the quiz. Just be warned that PHP's recursive acronym won't be the last weird and slightly funny thing you'll run across in the PHP language.

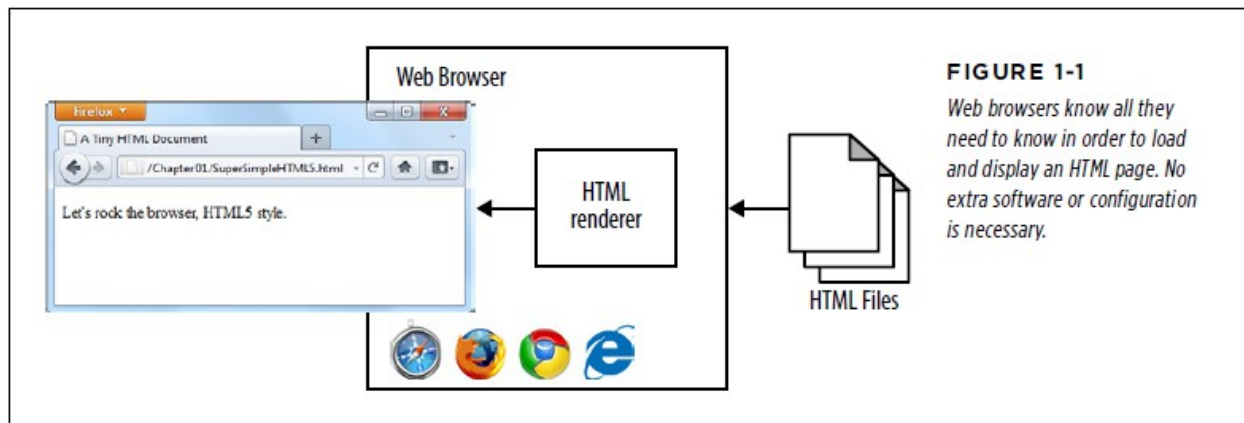
1.5 PHP Comes in Two Flavors: Local

and Remote

One of the most difficult things to get a handle on when it comes to PHP programming doesn't have much to do with programming at all. It's figuring out just how PHP runs, how it interacts with your web browser and web server, and why it's not possible to just double-click a PHP file on your hard drive and see the script in that file run.

1.6 HTML and CSS Run Within a Web Browser

First, it's worth thinking back to when you were a wee programmer, writing your first HTML page. You could save that page in a file, name that file with a .html extension, and boom—you had a web page. Double-click that file, and on most computers, you see that page open up in a web browser. That's because just as a .doc file is connected to the Microsoft Word program, a .html file is connected to a web browser (specifically, the browser you've chosen as the default on your computer). Figure 1-1 should give you an idea.



If you keep thinking back, you probably added some styling to your HTML pages. Using the style attribute and `<style></style>` tags in your HTML document, you could change fonts, add striping to your table rows, and generally spice up otherwise boring text.

Then, at some point, some well-meaning web designer slapped your hand and insisted that you start writing all your CSS in external style sheets, and referencing those files in the head of your HTML, like this:

```
<link rel="stylesheet" href="styles/mysite.css" type="text/css" />
```

You might even have a few style sheets for the benefit of people viewing your website on mobile devices or printing out a page:

```
<link rel="stylesheet" href="styles/mysite.css" type="text/css" media="all" />
```

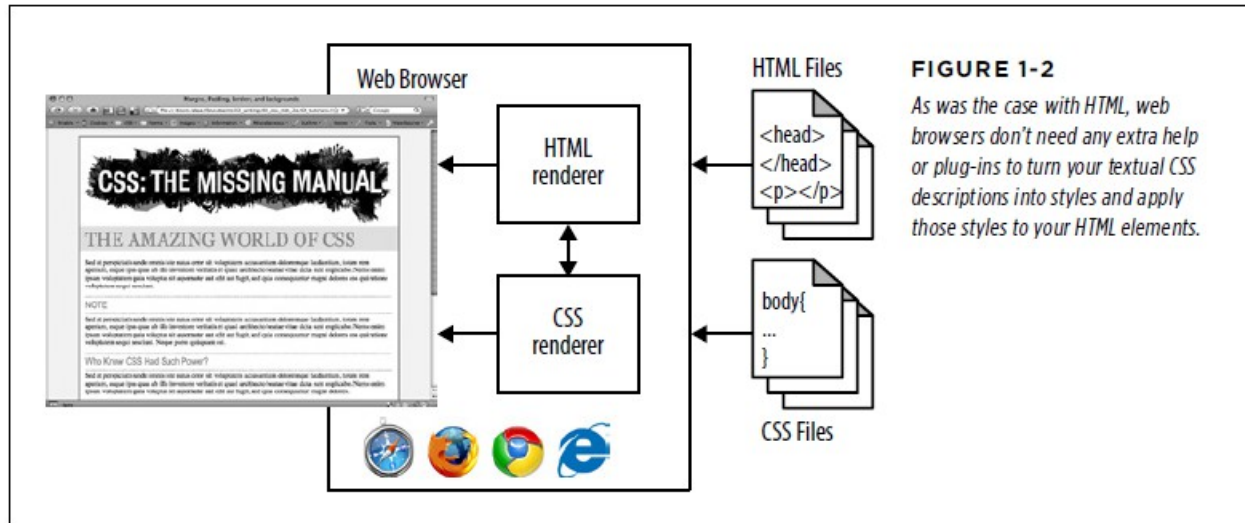
```
<link rel="stylesheet" href="styles/print.css" type="text/css" media="print"
```

```
/>
```

But you can still double-click that HTML file, and your browser knows what to do (see

Figure 1-2). That's because, once again, the web browser is completely capable of not just rendering HTML, but applying all those CSS styles to the page, too. Again, no extra software needed.

At this point, even though you're using only two technologies—HTML and CSS—you need only a single program to handle those technologies: the web browser.



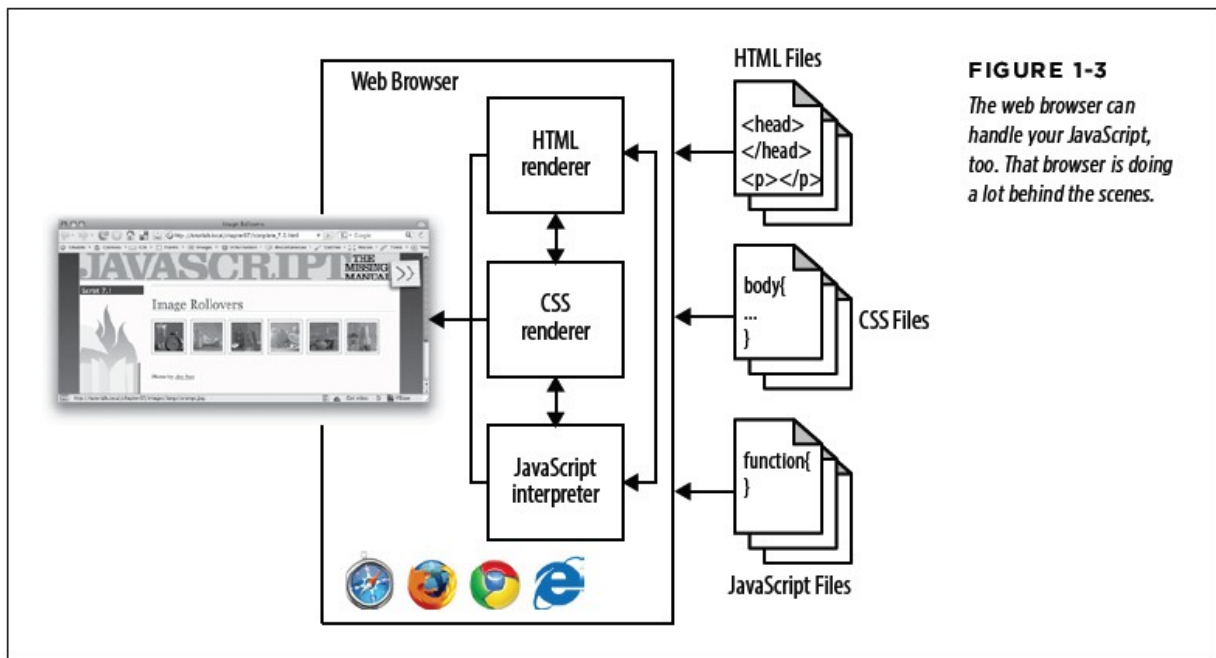
1.7 JavaScript Adds Complexity, but Not Software

Next up in the pantheon of web technologies that every designer and fledgling programmer needs to learn: JavaScript. Suddenly, you weren't limited to elements that never moved and text that never changed. Whether it was simple phone number validation, more advanced jQuery functions that turned boring gray boxes into animated buttons and `<div>` elements into tabs, or even the new HTML5 canvas object, within which you could build entire JavaScript-based 3D games, your pages suddenly had new life with JavaScript.

But just as with HTML and CSS, JavaScript is at heart a web technology, and even more specifically, a browser-based technology. In other words, support for JavaScript is part and parcel of your web browser. In fact, if a new version of JavaScript were to appear—something that rarely happens these days—you'd need to download a new

version of your browser to get that version of JavaScript. Just as you can't upgrade your HTML installation outside of your browser, you can't upgrade your JavaScript installation outside of your browser.

Figure 1-3 shows you how JavaScript fits in (hint: just as HTML and CSS do).



2. Advantages of Php

PHP also known as **Hypertext Preprocessor** was initially developed as a simple CGI application for interpreting form defined in HTML and processing them on the server side.

PHP (PHP current version 5.3) is a powerful package that provides an easy to learn programming language for generating HTML files. It's familiar, C-based syntax and the fact that's it's free to use make it a very attractive alternative to other server side languages like ASP.NET or Ruby.

I've decided to use PHP for this application because of the low web hosting prices for PHP websites and its ease of use and general reliability.

Another great feature of PHP is its vast support for numerous database systems. The following database systems are supported by PHP:

Adabas D	InterBase	PostgreSQL
dBase	FrontBase	Sesam
Empress	mSQL	Solid
FilePro	Direct MS-SQL	Sybase
Hyperwave	MySQL	Velocis
IBM DB2	ODBC	Unix dbm
Informix	Oracle	
Ingres	Ovrimos	

PHP offers support for multiple mail services and protocols including IMAP, SNMP, NNTP, POP3 and HTTP. This makes sending emails directly from the server side a very easy and straightforward task. PHP instructions can be interwoven with HTML code. The only requirement is that PHP instructions need to be placed between two special delimiters:

```
Code
<?php
    ...some code...
?>
```

3. Challenges:

1. In developing a E-commerce website we should take of several things like it should be cost efficient,user friendly,items should be displayed clearly. So in creating the work design we should take care of this factors.

2. PHP Is Not Part of Your Browser: Instead, you need PHP on a web server. It's the web server—not the web browser—that can interact with a PHP interpreter. Your browser can handle HTML on its own,but it has to make a request to a web server to deal with PHP scripts. That server can take your PHP scripts and run them, and then take the response and send it back to your browser. Your browser can then understand and handle the response.

3. There are lot of languages like Php,Css,Javascript,MySQL,Javaquery are used in the designing of E-commerce website so I have to dedicate myself to learn all these technologies.

4. Motivation:

1. To design an E-commecce website which is based on latest technologies yet it is very easy to use and host.

2. In designing an E-commerce I tried to create the basic design as simple as possible yet the basic structure can be used to host commercial E-commerce website by adding additional functionalities in the basic structure.

3. Motivation to take one of the most talked areas in IT world:E-commerce

5. Problem Statement:

Problem: Today, the revolution in technology is posing lots of challenges for businesses while also offering huge advantages. Businesses, more than ever, need to be efficient and cost effective reaching global customers but overcoming any complications in Communication, Research & Development, Procurement, Logistics & Order fulfillment, Sales & Marketing, Supply Chain etc. Hence the need of the time is for enterprises to be proactive in innovating, becoming more efficient, reducing costs, providing customers with better user experience value and service building customer relationships.

In this project, we are going to build an e-commerce system integrating latest technologies to real-world development scenarios as much as possible, by setting up a full-fledged Online Shoe Storefront for a fictitious company named Soccus in the footwear business.

The main objective of doing this project on not only how to build a solid ecommerce infrastructure but also the actual business aspects of the system.

Solution: E-commerce solution is a development model for all businesses from SMEs to Large enterprises selling products and services online. The model is envisaged to extend Internet advantage and enabling E-commerce capabilities in progressive stages to a full-fledged site for business.

CHAPTER-2

2.1 PHP Is Not Part of Your Browser

And here's where things change from the easy, browser-centric view of the world. When you download a web browser, you get HTML, CSS, and JavaScript, but you do not get PHP. PHP scripts—which you'll soon be writing—have to be interpreted by the PHP interpreter program, called `php`. And, you can't just add a PHP interpreter to your browser. It doesn't know what to do with scripts and isn't built to interpret PHP. Instead, you need PHP on a web server. It's the web server—not the web browser—that can interact with a PHP interpreter. Your browser can handle HTML on its own, but it has to make a request to a web server to deal with PHP scripts. That server can take your PHP scripts and run them, and then take the response and send it back to your browser. Your browser can then understand and handle the response. So, Figure 1-4 adds a couple of new wrinkles: the PHP interpreter, the magical thing that takes the PHP scripts you'll be writing and does something useful with them; and a web server to communicate with that interpreter. These both live outside of your web browser. In this scenario, the browser now makes a request to the server and then takes the response and shows it to you.

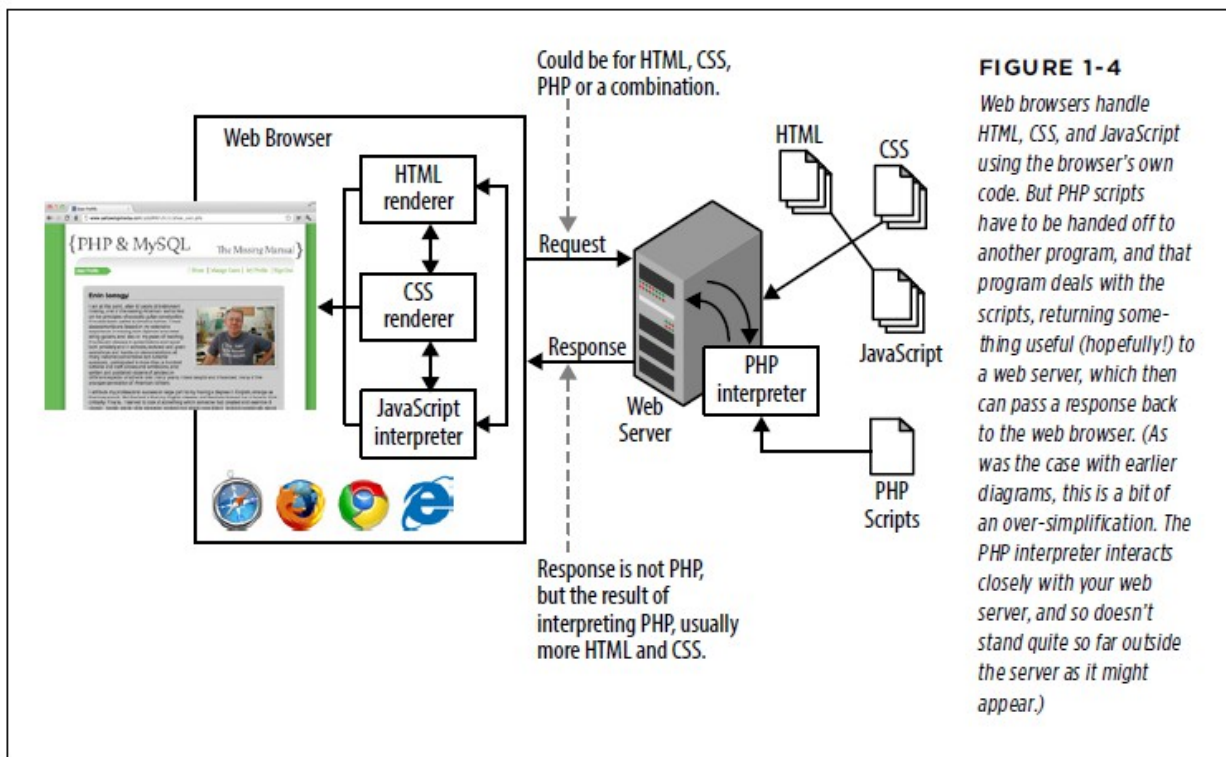


FIGURE 1-4

Web browsers handle HTML, CSS, and JavaScript using the browser's own code. But PHP scripts have to be handed off to another program, and that program deals with the scripts, returning something useful (hopefully!) to a web server, which then can pass a response back to the web browser. (As was the case with earlier diagrams, this is a bit of an over-simplification. The PHP interpreter interacts closely with your web server, and so doesn't stand quite so far outside the server as it might appear.)

Here's the basic process:

1. **A web browser makes a request for some page.** That page might be a URL on a remote web server, or a local file on your computer. Right away, there's potential for trouble here. If the browser requests a local HTML, CSS, or JavaScript file, there's no problem. That's because, as you now know, browsers can handle those file types. But if it requests a PHP file without going through a web server you're not going to get a response that the browser can handle on its own.
2. Assuming that the request goes to a web server, the web server returns HTML (and CSS and JavaScript) or, in the case of PHP, passes the PHP request on to the PHP interpreter.
3. The PHP interpreter does what it's supposed to: it interprets, or runs, the PHP. The result of that should be something that a browser **can** understand, like HTML. It passes this result, or response, back to the web server.
4. The web server gives the browser back something that the browser can understand: the HTML result of interpreting a PHP script, or CSS, or JavaScript, or a combination of all of the above.

Understanding this difference in how PHP works, as opposed to HTML, CSS, and JavaScript, is important because it determines the approach you'll take to writing PHP scripts and getting those scripts to run.

2.2 Write Anywhere, Run Where There's PHP

The cool thing about HTML, CSS, and JavaScript is that because they're built in to browsers and you can download browsers so easily, those technologies become instantly available. It's tough to even find a computer without a browser preinstalled. So, you turn on your computer for the first time, and boom, you can start creating web pages immediately. Double-click the HTML file, your browser fires up, and you're good to go.

But PHP isn't part of that browser. It's not always preinstalled. If you write a PHP script and then double-click it, you'll probably see a code editor launch, but not something that will actually run that script. Even worse, if your browser does open up your PHP script, it's not a web server. It doesn't have a PHP interpreter. It will just show you your code, rather than run it, and what good is that to anyone?

This long prelude is just a big warning: although it's easy enough to start writing PHP scripts, you can't just open them in Dreamweaver or Firefox and expect them to run. You'll end up frustrated and annoyed, and that's no good for anyone.

The bottom line is this: You can write PHP on your own local computer, but you've got two choices for actually running that PHP:

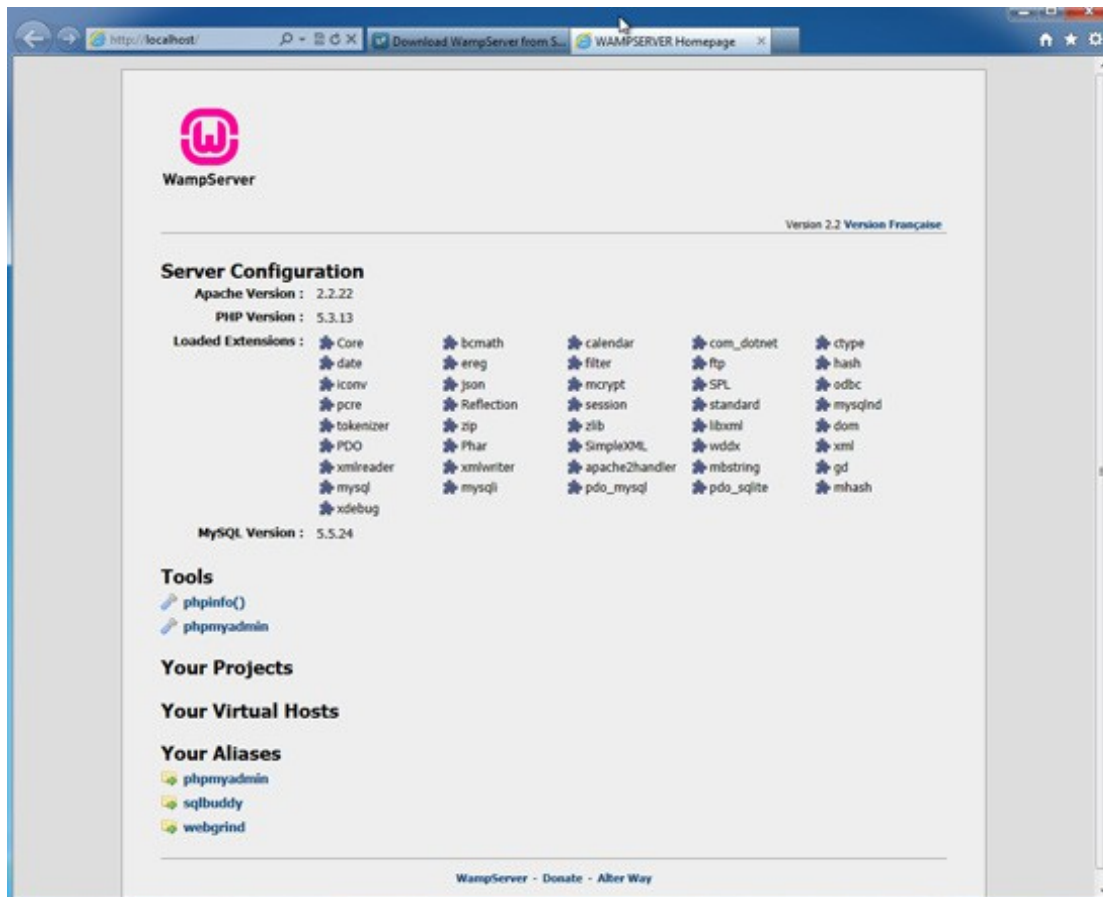
1. You can go through the lengthy process detailed in the next section and

install PHP on your local computer. This process will take some time, and you'll have to monkey around a bit with your computer at a system and network level. You'll also need a local web server to handle the PHP interpreting part of the gig. This way, you'll not only have a browser that can handle HTML, CSS, and JavaScript, but a complete setup that can take on PHP without a problem, too—right on your own computer.

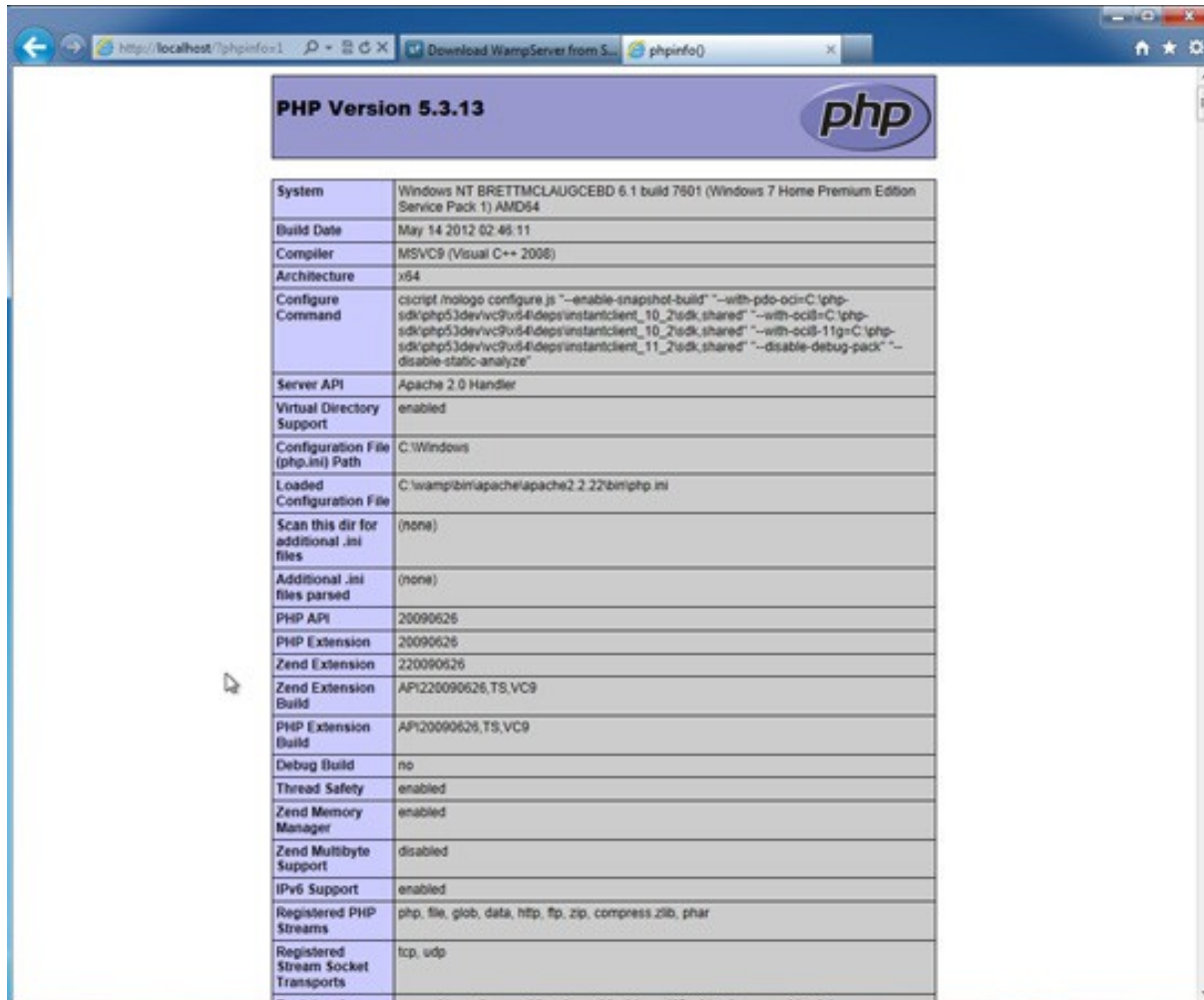
2. You can write your scripts locally and always upload them to an Internet Service Provider (ISP) or web hosting company. Every ISP and web hosting company supports PHP, and you usually don't have to do anything more than name your scripts with a .php extension. This option involves less initial setup, but it means that every time you edit your script, you need to upload it again to your ISP. It also means that double-clicking your PHP script won't do anything more than, at best, open your editor. You can't test your scripts on your own

Both choices are equally good, and which one you choose depends largely on your circumstances. Even though it might seem perfectly natural to jump right into uploading your scripts, you aren't always going to have a network connection. (The sound you just heard was the cheering of all the programmers who have an hour-long commute into work on their local metro or subway!) For those unwired situations, it's nice to be able to keep developing on your own computer without the need to access your hosting provider. Note only that, installing PHP on your own computer is great for understanding what the PHP interpreter actually does.

For running PHP on my system I have used WAMP which stands for Windows, Apache, MySQL, PHP.



Having a web server running on your local computer isn't necessary for developing HTML, CSS, or most JavaScript applications. But because a browser can't interpret PHP, a local web server is essential if you want to write PHP scripts on that computer and run them without uploading them to a server somewhere



PHP Version 5.3.13	
System	Windows NT BRETTMCLAUGCEBD 6.1 build 7601 (Windows 7 Home Premium Edition Service Pack 1) AMD64
Build Date	May 14 2012 02:45:11
Compiler	MSVC9 (Visual C++ 2008)
Architecture	x64
Configure Command	cdscript /nologo configure.js "--enable-snapshot-build" "--with-pdo-oci=C:\php-sdk\php53dev\vc9\vs9-4depr\instantclient_10_2\odk_shared" "--with-oci=C:\php-sdk\php53dev\vc9\vs9-4depr\instantclient_10_2\odk_shared" "--with-oci-11g=C:\php-sdk\php53dev\vc9\vs9-4depr\instantclient_11_2\odk_shared" "--disable-static-analyze"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\Windows
Loaded Configuration File	C:\wamp\bin\apache\apache2.2.22\bin\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20090626
PHP Extension	20090626
Zend Extension	220090626
Zend Extension Build	API220090626,TS,VC9
PHP Extension Build	API20090626,TS,VC9
Debug Build	no
Thread Safety	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
Registered PHP Streams	php, file, glob, data, http, ftp, zip, compress, zlib, phar
Registered Stream Socket Transports	tcp, udp

And the big win: PHP is running! Actually, your browser made a request to your local web server, your local web server executed some PHP, and then it responded to your browser with the response from that PHP command.

2.3 Determination by Extension

PHP scripts are identified by the extension `.php`. Accordingly, web servers that supports

PHP see a file with a `.php` extension and hand that file off to the PHP interpreter for processing. The interpreter does its thing and hands the result of the interpreted script back to the web server, which in turn passes that response along to a user's web browser.

2.4 JavaScript Basic Introduction

JavaScript or JS is a widely used scripting language that's mostly used in combination with HTML. It uses a simple, easy to learn C-style syntax but unlike PHP doesn't provide support for traditional Object Oriented Programming concepts. This can be a challenge for

developers who've worked only with classic OOP languages like C.

JavaScript has been in use since the early 90's but only recently has it become a truly popular programming language with the rise of technologies such as AJAX and HTML5.

JavaScript code can be placed in external .js files or in the HTML document within the <head> or <body> tags.

An important thing to note about JavaScript is that it can only access HTML elements that exist in the DOM. The DOM or Document

Object Model is the way that an HTML document is represented inside the browser. When an internet browser opens a HTML document

it will generate a DOM for it. It will then traverse this DOM and render the web page to the screen. If JavaScript code is set to run before

the DOM is generated by the browser then this code can access only HTML elements that are present in the DOM.

Basically what this means is that JavaScript code can only access HTML elements that are declared before it. This is the reason that some developers place their code at the end of the document so that it can access all of the elements on the page.

JavaScript is a loosely typed language, meaning that there's no need to declare variable types, the language does this automatically based on the value you assign to a variable.

2.5 Functions

A function is a block of code that executes only when you tell it to execute. It can be when an event occurs, like when a user clicks a

button, or from a call within your script, or from a call within another function.

Functions can be placed both in the <head> and in the <body> section of a document.

2.6 jQuery

jQuery is a JavaScript library designed to simplify the client-side scripting of HTML. It's free to use, open source and has seen a huge

increase in popularity over the years. The reason why it has become so popular is that it allows programmers to create complex

animations and effects with relative ease and without requiring vast knowledge in programming or JavaScript.

The heart of jQuery is its powerful selector engine. This allows us to easily access and modify any HTML tag on the page. For example the following instruction will target every <div> on the page that has a class of .red

2.7 Creating configuration files and rendering the main page

Before I could get started on coding the main store pages in PHP, I needed to set some configuration variables so that I could later access them from any page on the site. I also needed to create a couple of function for displaying the page head (this is common for all the pages of the site) and checking if the current user is logged in.

All of these things are located in the functions.php file. The first lines of this file contain general configuration settings such as the site title and database login details.

Code

```
//===== Global Variables=====
$site_title = 'Magazin Online';
$site_url = 'http://localhost/alarmebuzau/';
$document_path = $site_url;
//===== Database Connection=====
$dbUser = "AA";
$dbPass = "aa";
$dbServer = "localhost";
```

Following these is the function that renders the document head for a page. Some key elements to note are the inclusion of jQuery library <script type="text/javascript" language="javascript" src="jquery.js"></script> and the site title <title>Alarme Buzau</title> . Both of these are generated with PHP based on the configuration variables described above. The PHP code that renders the head is as follows:

Code

```
function display_head(){
global $site_title, $site_url, $document_path;
echo '<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional..<html
xmlns="http://www.w3.org/1999/xhtml" lang="ro">'. "\n"
.<head>'. "\n"
.<title>'. $site_title. '</title>'. "\n"
.<link href="'. $document_path . 'css/style.css" media="screen" rel="stylesheet"
type="text/css" />'. "\n"
.<link rel="shortcut icon" href="'. $document_path . 'images/favicon.ico"/>'. "\n"
.<script type="text/javascript" language="javascript" src="'. $document_path . 'script/js/jquery-
1.7.2.min.js"></script>'
```

```

.<script type="text/javascript" language="javascript" src="".
$document_path .script/js/jquery.roundabout.min.js"></.<script type="text/javascript" src="".
$document_path .script/jwplayer/jwplayer.js"></script>'. "\n"
.</head>' . "\n";
}

```

The next and final function in functions.php is used to check if the user is logged in. Every time a customer logs in, PHP will set a value for the following variable `$_SESSION['user_email']`. This variable is stored in PHP's session which means it will be only available for the current customer. Each customer has his own session.

The function checks if the above mentioned variable has been set and if so it will return true; otherwise it will return false indicating that the customer is not logged in.

Code

```

function isLoggedIn(){
if(isset($_SESSION['user_email']) && $_SESSION['user_email'] != ""){
return true;
}
else{
return false;
}
}
}

```

Having taken care of the configuration details, the next thing I coded was the landing page.

The landing page or home page is the first page that a customer sees when he enters the site's address in a browser. The general layout

used on this page can be seen in all other pages of the site, thus maintaining a consistent theme.

The only difference between two

pages will be the content area. The basic template for these pages is presented below.

Code

```

<?php
include 'script/functions.php';
display_head();
?>
<body>
<?php include 'header.php'; ?>
<div id="carousel-container">
<!--carousel goes here-->
</div>
<div class="wrapper">
<!--content goes here-->
<?php include 'footer.php'; ?>
</div>
<script type="text/javascript">
<!--javascript goes here-->

```

```

</script>
</body>
</html>

```

The first two lines are written in PHP. They're responsible for accessing functions.php so that the variables and functions declared there are available in the current page. Then the display_head() function is called that will display the page head.

The next part is responsible for including the page header include 'header.php' . This header is the top menu that appears on all pages.

The file that's included contains both HTML and PHP code that is directly inserted in the current page.

The last important part to note here is the inclusion of the footer which is done in similar way as the header.

The carousel is created using the Roundabout JavaScript library. In order to make it run two things are required: HTML markup and JavaScript code.

For the former all that was needed was a simple unordered list in which each list item is an image.

Code

```

<div id="carousel-container">
<ul id="carousel">
<li></li>
...
</ul>
</div>

```

And on the JavaScript side, we include the library in the document head and write the necessary code so that it's properly initialized.

Code

```

<script type="text/javascript" language="javascript"
src="<?php echo $document_path ?>script/js/jquery.roundabout.min.js"></script>
<script type="text/javascript">
$(document).ready(function() {
var carousel = $('#carousel').hide();
//initialize the carousel
carousel.roundabout({autoplay:true, autoplayDuration:6000});
//display the carousel
carousel.show();
});
</script>

```

Using jQuery like this `$(document).ready` allows us to run the code after the page has finished loading. This way I was able to avoid problems with the DOM that can arise when you run JavaScript code before the DOM is completely loaded.

To display the products on the main page I wrote PHP code that goes through the categories and products tables. The steps needed to do this are as follows:

1. Connect to the database

Code

```
$conn = oci_connect($dbUser, $dbPass, $dbServer);
```

2. Select from the database only the parent categories

Code

```
$sql = oci_parse($conn, 'select * from categories where parent is null');
oci_execute($sql);
```

3. Each of these parent categories corresponds to one of the four columns on the page so the code will render a column for each row that's returned by the database query.

Code

```
for($i=0; $i<4; $i++){
$row = oci_fetch_assoc($sql);
1<
/li>
```

For each of the main categories go through the database and search for subcategories.

```
1$
sql2 = oci_parse($conn, "select * from categories where parent = " . $row["NAME"] . "");
oci_execute($sql2);
```

4. Render the subcategory and search for subcategories for it.

Code

```
$sql3 = oci_parse($conn, "select * from categories where parent = " . $row2["NAME"] . "");
oci_execute($sql3);
```

5. The menu is now rendered. When a user hovers the mouse over one of the 4 main column titles he will be shown a menu with subcategories. To enable this functionality, I used JavaScript

Code

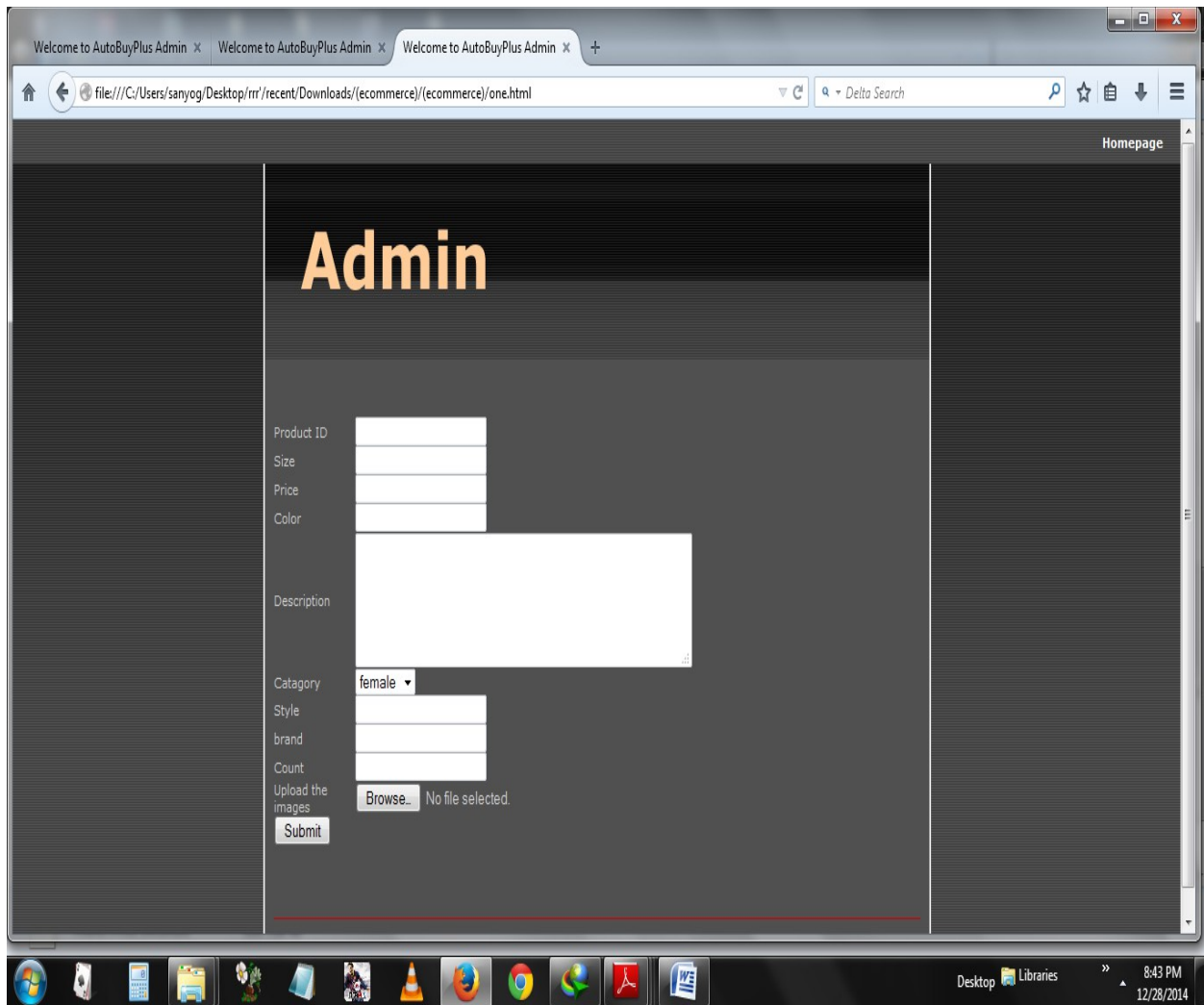
```
$(".menu li").hover(function(){
$(this).addClass("hover");
$('ul:first',this).css('visibility', 'visible');
}, function(){
```

```
$(this).removeClass("hover");
$('ul:first',this).css('visibility', 'hidden');
});
```

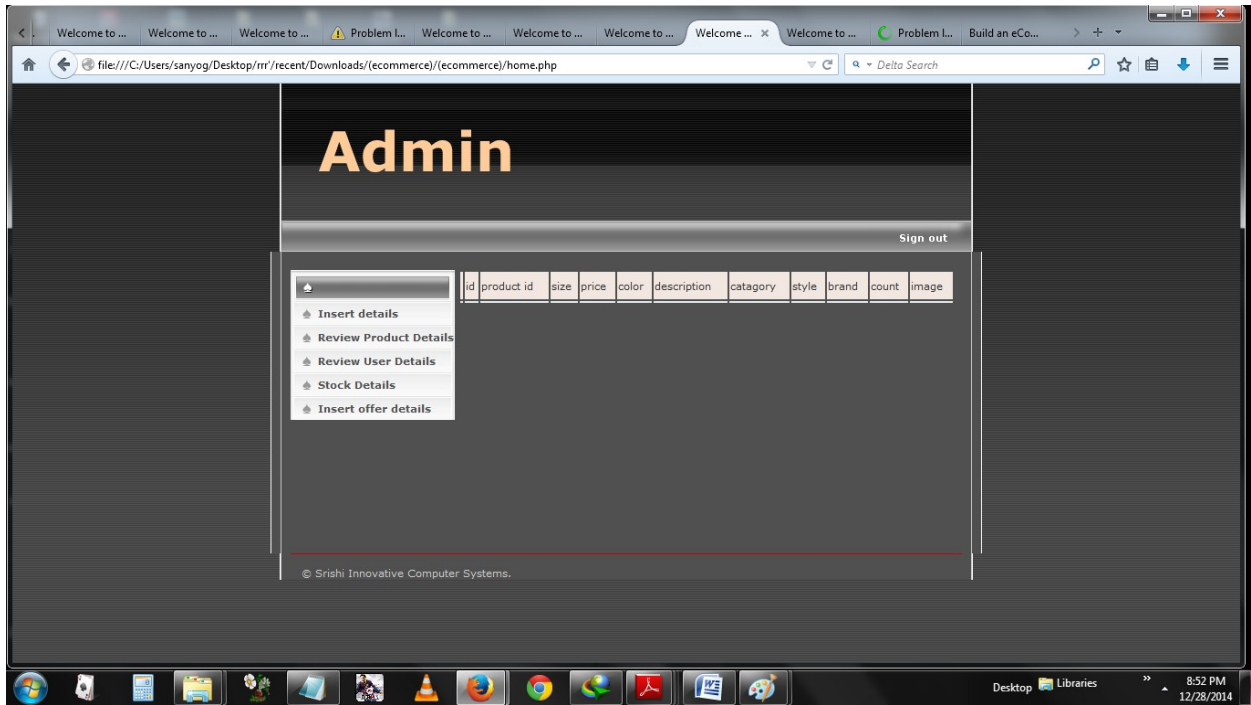
6. Finally, the code will display the products in each category.

Code

```
$sql4 = oci_parse($conn, "select * from products where category_pk = " . $cat . " ");
oci_execute($sql4);
<div class="prodDescription" style="margin:15px 0 0 0;">
<a href="<?php echo $document_path ?>product.php?prod_id=<?php echo $row4["ID"] ?
>"><?php echo $row4["TITLE"];
<a href="<?php echo $document_path ?>product.php?prod_id=<?php echo $row4["ID"] ?>">
" width="170" height="170"
alt="<?php echo $row4
</a>
<span class="details">
<?php echo $row4["DESC1"]; ?><br/>
<?php echo $row4["DESC2"]; ?><br/>
<?php echo $row4["DESC3"]; ?><br/>
<?php echo $row4["DESC4"]; ?><br/>
<?php echo $row4["DESC5"]; ?><br/>
<?php echo $row4["DESC6"]; ?><br/>
</span>
<?php if(isset($row4["SALE"])){?>
<span>Pret: <?php echo $row4["PRICE"]; ?>&nbsp; Lei (cu TVA)</span>
<span>Reducere: <?php echo $row4["SALE"]; ?> <br/></span>
<span>Pret: <?php echo $row4["PRICESALE"]; ?>(cu TVA)<br/></span>
<?php
} else{
?>
<br/>
<span>Pret: <?php echo $row4["PRICE"]; ?>(cu TVA)<br/></span>
<?php } ?>
</div>
```

Snapshot of My Ecommerce website



REFERENCES

1. The Missing Manual :PHP AND MYSQL(2ND EDITION)
2. The Missing Manual: JAVASCRIPT AND JQUERY(2ND EDITION)
- 3.The Missing Manual: HTML AND CSS(2ND EDITION)
4. A Comparative Study of Recommendation Algorithms in ECommerce Applications By Zan Huang
5. Vanletin-radulescue-me paper on how to design an E-commece website using Php.
- 6.w3schools.com

7. Analysis of Recommendation Algorithms for ECommerce by Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl