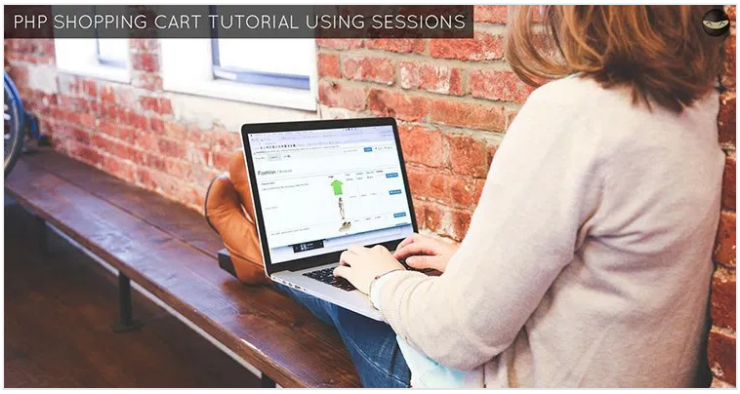
PHP and MySQL Shopping Cart Tutorial – Using SESSIONS To Store Cart Data

Last Update: July 14, 2019 • Date Posted: April 6, 2013 • by Mike Dalisay ➜ Get FREE Updates Here



Previously, we learned how to build a Shopping Cart with PHP & MySQL [https://www.codeofaninja.com/2015/08/simple-php-mysql-shopping-cart-tutorial.html] where we used a database to store cart items. Today, we will learn another version of it. We will use PHP session variables [https://php.net/manual/en/reserved.variables.session.] to store cart items.

This tutorial have the following contents:

Before we start, we want to let you know that your feedback is important to us!

If you have a positive feedback about our work, please let us know. If there's a section in this tutorial that is confusing or hard to understand, we consider it as a problem. Please let us know as well.

Write your positive feedback or detailed description of the problem in the comments section below. Before you write a comment, please read this guide [https://www.codeofaninja.com/how-to-write-a-great-comment-on-codeofaninja-com] and our code of conduct [https://www.codeofaninja.com/code-of-conduct] . Thank you!

1.0 OVERVIEW

1.1 Introduction

If you want to build your own online shopping cart from scratch, we have good news for you!

This post can help you get it done because we will build a simple shopping cart script today.

We will use PHP, MySQL and PHP sessions [https://php.net/manual/en/reserved.variables.session.
to complete this task.

A lot of people use a ready-made software for this.

But for coders like us, it is important to learn and experience how to do it. We can create more features like making the system more

PHP Contact Form Module

PHP PayPal Integration Module

MORE SCRIPTS

More code examples that can be useful for you!

Shopping Cart Tutorial using COOKIES

Extra Tutorials

WHAT STUDENTS SAY

"Wow, I love you guys! The best web programming tutorial I've ever seen. So comprehensive, yet easy to follow. I love how you combine all necessary elements in such a neat structure."

Olaug Nessa

secured, add some unique functionality and more.

Your imagination can be the only limit.

1.2 Is this code for you?

The source codes in this page is NOT for you if:

- You are already an expert in PHP & MySQL programming.
- You have a lot of time to code a shopping cart system from scratch.
- You are not that interested in learning PHP & MySQL programming.

But, this SOURCE CODE is FOR YOU if:

- You want to SAVE huge amount of development time.
- You want to develop your own shopping cart system from scratch.
- You determined to learn how to make a web application in PHP & MySQL.

But if you are an expert in PHP & MySQL programming and would like to take a look at our code, please do so! We'd love to hear your response and great insights! The comments section below is always open for anyone with questions and suggestions.

1.3 How to use this tutorial?

This tutorial is already working. You can proceed to the instructions below.

But we are recording a video demo about how to use this tutorial.

"The fact that you've put it all together saves so much time and its worth buying the code. Makes me feel good supporting a developer like yourself. Keep up the good

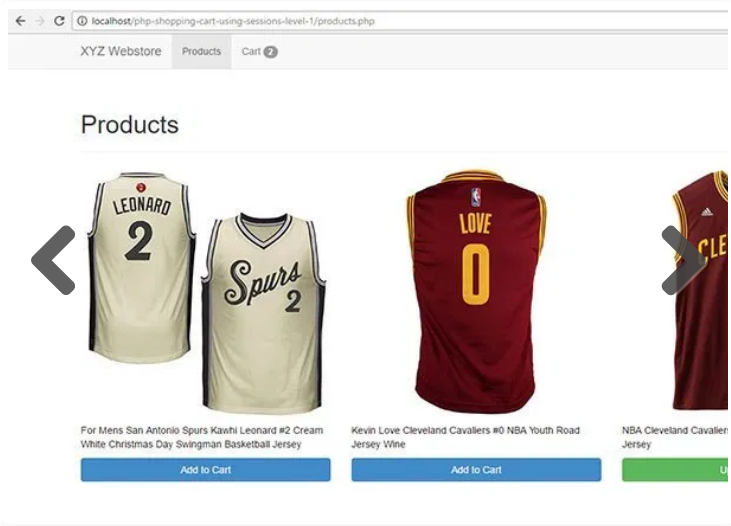
It is coming soon! Please subscribe here [https://www.codeofaninja.com/subscribe] so you will be updated.

2.0 TUTORIAL OUTPUT PREVIEW

Below are some screenshots of our script's output. You can click an image to view the larger version of it. Use the left and right arrow to navigate through the screenshots.

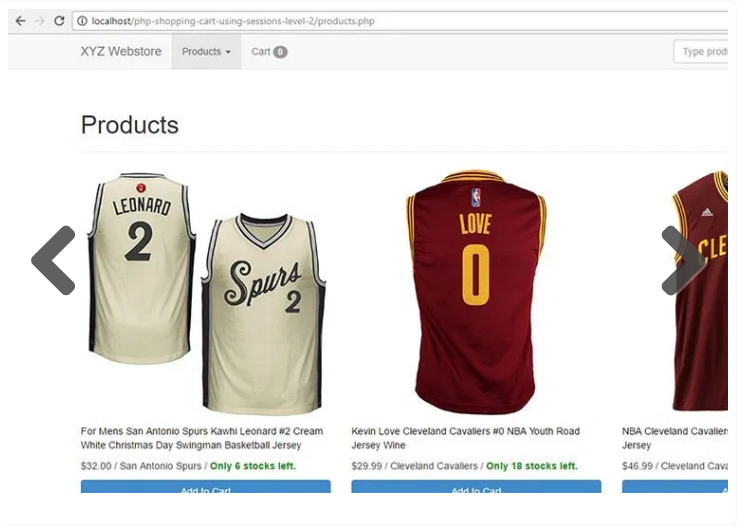
Please note that the following images are just output previews. New features might be added already the time you are reading this.

2.1 LEVEL 1 Source Code Output



LEVEL 1 LIVE DEMO HERE [https://www.codeofaninja.com/demos/shopping-cart-using-sessions-level-1/products.php]

2.2 LEVEL 2 Source Code Output



The LEVEL 2 source code output proves that you can add and customize more features. It will be easier and faster if you will learn by following our tutorial below.

Downloading our source codes is your huge advantage as well.

If you need more features like product variations, admin features and user login, see our PHP Shopping Cart Module [https://www.codeofaninja.com/2016/07/php-online-shopping-cart-source-code.html] and PHP Shopping Cart System [https://www.codeofaninja.com/2016/04/php-shopping-cart-source-code-download.html] .

For now, let's proceed to the step by step tutorial of our LEVEL 1 source code. Enjoy!

3.0 FILE STRUCTURE

The following folders and files are included in the final source code of this tutorial. It will have

more meaning if you will see the code inside the folders and files as we go through this tutorial.

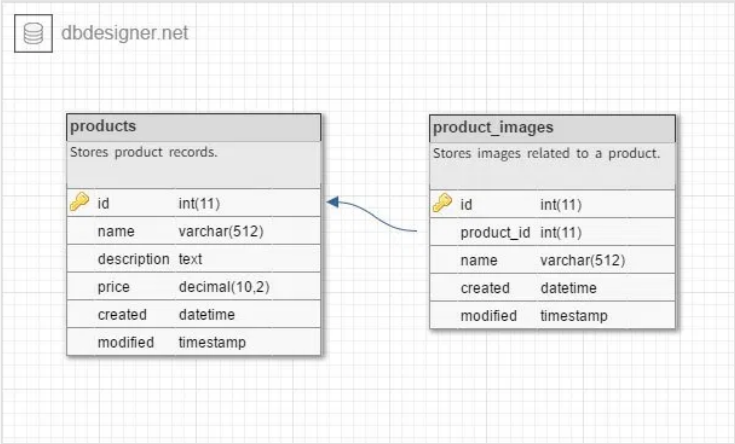
The branch with backslash represents a folder. Everything else represents a file. It can be a PHP file, SQL file, text file, CSS file or JavaScript file.

```
├── config/
├──────── database.php
├── dev/
├──────── shop_cart_sessions_1.sql
├──────── readme.txt
├── images/
├── libs/
├──────── css/
├──────────────── bootstrap/
├──────── js/
├──────────────── jquery.js
├── objects/
├──────── product_image.php
├──────── product.php
├── uploads/
├──────── images/
├── .htaccess
├── add_to_cart.php
├── cart.php
├── checkout.php
├── layout_footer.php
├── layout_header.php
├── navigation.php
├── paging.php
├── place_order.php
├── product.php
├── products.php
├── read_products_template.php
├── remove_from_cart.php
├── update_quantity.php
```

4.0 PREPARE THE DATABASE

4.1 Database Design

Our database name will be called "shop_cart_sessions_1", and we will have two (2) tables. The image below is a visual representation of our database tables and how they are related.



[https://i2.wp.com/www.codeofaninja.com/wp-content/uploads/2013/04/database-design.jpg?ssl=1]

4.2 Create a database

Make sure your Apache and MySQL servers are running.

- Open your PhpMyAdmin (http://localhost/phpmyadmin [http://localhost/phpmyadmin])
- Create a new database.
- Put "shop_cart_sessions_1" as database name.
- Click "Create" button.

4.3 Create "products" table

In this section, we will create the "products" table (using PhpMyAdmin) on the database we just created. This table will hold the product records.

Here's how to run an SQL statement using PhpMyAdmin.

- Click "shop_cart_sessions_1" database.
- Click "SQL" tab.
- Copy the SQL statement below and paste it in the text area.
- Click the "Go" button.

```
CREATE TABLE IF NOT EXISTS `products`
  `id` int(11) NOT NULL AUTO_INCREMEN
  `name` varchar(512) NOT NULL,
  `description` text NOT NULL,
  `price` decimal(10,2) NOT NULL,
  `created` datetime NOT NULL,
  `modified` timestamp NOT NULL DEFAU
  PRIMARY KEY (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=lati
```

4.4 Create "categories" table

This table will hold images related to product.

Run the following SQL statement using your PhpMyAdmin.

```
CREATE TABLE IF NOT EXISTS `product_i
  `id` int(11) NOT NULL AUTO_INCREMEN
  `product_id` int(11) NOT NULL,
  `name` varchar(512) NOT NULL,
  `created` datetime NOT NULL,
  `modified` timestamp NOT NULL DEFAU
  PRIMARY KEY (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=utf8
```

4.5 Download sample data and images

The **products** and **product_images** table will not fully work without the sample data and related

image files.

To make things easier, I decided to create a **ZIP file** with **shop_cart_sessions_1.sql** and **28 image files** inside (1.30 MB).

Use the following button to download the ZIP file.

> DOWNLOAD SQL FILE AND IMAGES [https://drive.google.com/file/d/0B-AInNrVeucKenZzZjRWNUVrYVE/view]

4.6 Extract and import data

Once downloaded, please extract the files.

Import the SQL file using PhpMyAdmin.

Put the image files in "php-shopping-cart-using-sessions-level-1/uploads/images/" directory. That directory does not exist yet. We need to create it now.

- Create "php-shopping-cart-using-sessions-level-1" folder and open it. This is our project's main folder.
- Create "uploads" folder and open it.
- Create "images" folder and open it.
- Copy and paste the images on this directory.

4.7 Database connection file

This file will be used to get connection to the database.

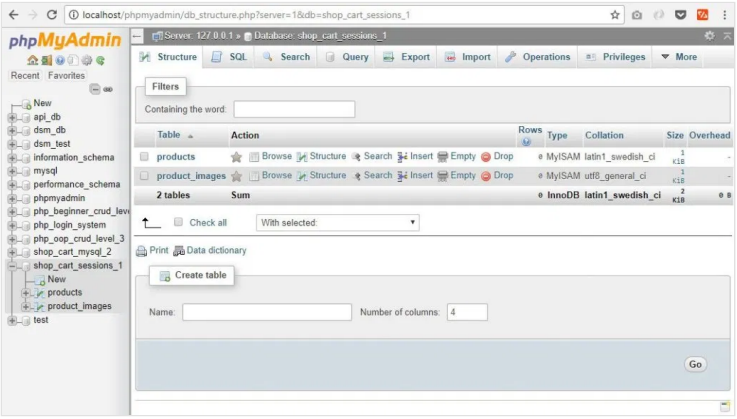
- Open "php-shopping-cart-using-sessions-level-1" folder.
- Create "config" folder and open it.
- Create "database.php" file and open it.

- Place the following code.

```
<?php
// used to get mysql database connect
class Database{

    // specify your own database cred
    private $host = "localhost";
    private $db_name = "shop_cart_ses
    private $username = "root";
    private $password = "";
    public $conn;

    // get the database connection
    public function getConnection(){

        $this->conn = null;

        try{
            $this->conn = new PDO("my
        }catch(PDOException $exceptio
            echo "Connection error: "
        }

        return $this->conn;
    }

}
?>
```

4.8 Output

Our PhpMyAdmin should look like the image below. A database with two tables.



We don't have an actual program output yet because we only set up the database. Let's

continue our tutorial below to achieve more outputs.

5.0 CREATE THE LAYOUT FILES

5.1 Create header layout file

This “layout_header.php” file will be included at the beginning of the PHP files that will need it. This way, we won’t have to write the same header codes every time.

We use the Bootstrap framework [https://getbootstrap.com/docs/3.3/] to make our project look good. If you’re not yet familiar with you, please learn our Bootstrap tutorial here [https://www.codeofaninja.com/2014/05/bootstrap-tutorial-beginners-step-step.html] .

Bootstrap CSS asset will be included inside the head tags.

- Open "php-shopping-cart-using-sessions-level-1" folder.
- Create "layout_header.php" file.
- Place the following code.

```
<?php
$_SESSION['cart']=isset($_SESSION['ca
?>
<!DOCTYPE html>
<html lang="en">
<head>

    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible
    <meta name="viewport" content="wi

    <title><?php echo isset($page_tit
```

```
    <!-- Latest compiled and minified
    <link rel="stylesheet" href="http

    <!-- our custom CSS -->
    <link rel="stylesheet" href="libs

</head>
<body>

    <?php include 'navigation.php'; ?

    <!-- container -->
    <div class="container">
        <div class="row">

        <div class="col-md-12">
            <div class="page-header">
                <h1><?php echo isset(
            </div>
        </div>
```

5.2 Create footer layout file

This "layout_footer.php" will be included at the end of the PHP files that will needs it. This way, we won't have to write the same footer codes every time.

The assets used in this file are:

- jQuery [https://jquery.com/] – needed by Bootstrap JavaScript.
- Bootstrap JavaScript [https://getbootstrap.com/docs/3.3/javascript/] – to make cool UI components work.

Let's go on and create the footer layout file.

- Open "php-shopping-cart-using-sessions-level-1" folder.
- Create "layout_footer.php" file.
- Place the following code.

```
        </div>
        <!-- /row -->
```

```
    </div>
    <!-- /container -->

<!-- jQuery (necessary for Bootstrap'
<script src="https://code.jquery.com/

<!-- Latest compiled and minified Boo
<script src="https://maxcdn.bootstrap

<!-- custom script will be here -->

</body>
</html>
```

5.3 Create navigation layout file

This file will render the "Products" and "Cart" links that the user can click.

- Open "php-shopping-cart-using-sessions-level-1" folder.
- Create "navigation.php" file.
- Place the following code.

```
<!-- navbar -->
<div class="navbar navbar-default navl
    <div class="container">

        <div class="navbar-header">
            <button type="button" cla
            <span class="sr-only">Tog
            <span class="icon-bar"></
            <span class="icon-bar"></
            <span class="icon-bar"></
            </button>
            <a class="navbar-brand" h
        </div>

        <div class="navbar-collapse c
            <ul class="nav navbar-nav

                <!-- highlight if $pa
                <li <?php echo $page_
                    <a href="products
                </li>

                <li <?php echo $page_
                    <a href="cart.php
                        <?php
                        // count prod
```

```
                        $cart_count=c
                        ?>
                        Cart <span cl
                    </a>
                </li>
            </ul>

        </div><!--/.nav-collapse -->

    </div>
</div>
<!-- /navbar -->
```

5.4 Create custom CSS file

This custom.css file is where our custom styles are located.

- Open "php-shopping-cart-using-sessions-level-1" folder.
- Open "libs" folder.
- Open "css" folder.
- Create "custom.css" file.
- Place the following code.

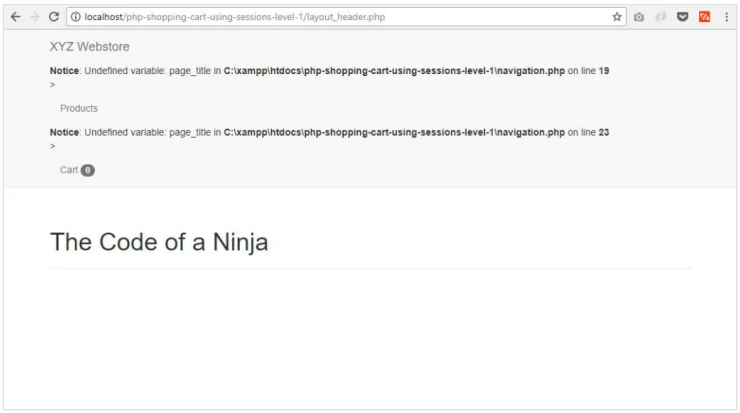
```
.text-align-center{ text-align:center
.f-w-b{ font-weight:bold; }
.display-none{ display:none; }

.w-5-pct{ width:5%; }
.w-10-pct{ width:10%; }
.w-15-pct{ width:15%; }
.w-20-pct{ width:20%; }
.w-25-pct{ width:25%; }
.w-30-pct{ width:30%; }
.w-35-pct{ width:35%; }
.w-40-pct{ width:40%; }
.w-45-pct{ width:45%; }
.w-50-pct{ width:50%; }
.w-55-pct{ width:55%; }
.w-60-pct{ width:60%; }
.w-65-pct{ width:65%; }
.w-70-pct{ width:70%; }
.w-75-pct{ width:75%; }
.w-80-pct{ width:80%; }
.w-85-pct{ width:85%; }
.w-90-pct{ width:90%; }
.w-95-pct{ width:95%; }
.w-100-pct{ width:100%; }
```

```
.m-t-0px{ margin-top:0px; }
.m-b-10px{ margin-bottom:10px; }
.m-b-20px{ margin-bottom:20px; }
.m-b-30px{ margin-bottom:30px; }
.m-b-40px{ margin-bottom:40px; }

.stock-text {
    font-weight: bold;
    color: #008a00;
}

.stock-text-red{
    font-weight:bold;
    color:#b12704;
}

.product-detail {
    font-weight: bold;
    margin: 0 0 5px 0;
}

.blueimp-gallery>.prev, .blueimp-gall

.update-quantity-form {
    width: 150px;
    float: left;
    margin: 0 10px 0 0;
}

.cart-row {
    border-bottom: thin solid #f1f1f1
    overflow: hidden;
    width: 100%;
    padding: 20px 0 20px 0;
}

.product-link{
    color:#000000;
}

.product-link:hover{
    color:#000000;
    text-decoration:none;
}

.product-img-thumb {
    margin: 0 0 10px 0;
    width: 100%;
    cursor: pointer;
}
```

5.5 Output

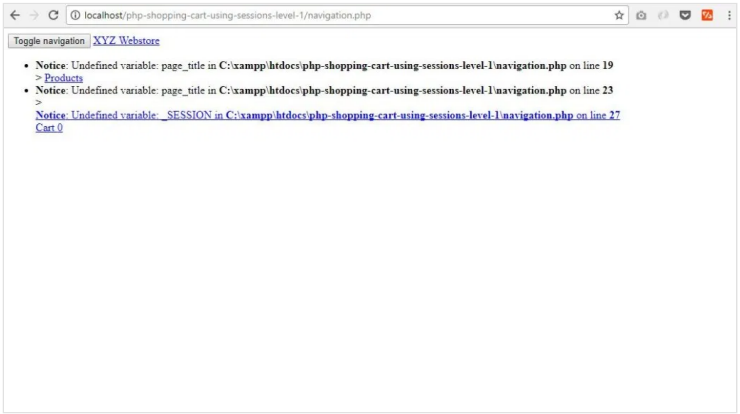
The files we created in this section is meant to be used within another PHP file. If we will try to run the files, we won't see anything meaningful yet.

If you will run layout_header.php file, it will look like this.



The custom.css looks like this.

The navigation.php looks like this.



The footer.php is blank. Let's continue on the next section to see something meaningful.

6.0 DISPLAY PRODUCTS

6.1 Create products.php

Now we are going to start displaying products from the database. Create products.php with the following basic code.

```
<?php
// start session
session_start();

// set page title
$page_title="Products";

// page header html
include 'layout_header.php';

// contents will be here

// layout footer code
include 'layout_footer.php';
?>
```

6.2 Include PHP Classes

Put the following code after "session_start();" code of the previous section.

```
// connect to database
include 'config/database.php';

// include objects
include_once "objects/product.php";
include_once "objects/product_image.p

// class instances will be here
```

6.3 Create "product" object file

Create "objects" folder. Inside it, create product.php file with the following code.

```
<?php
// 'product' object
class Product{

    // database connection and table
```

```
    private $conn;
    private $table_name="products";

    // object properties
    public $id;
    public $name;
    public $price;
    public $description;
    public $category_id;
    public $category_name;
    public $timestamp;

    // constructor
    public function __construct($db){
        $this->conn = $db;
    }
}
```

6.4 Create "product image" object file

Create product_image.php file inside "objects" folder.

```
<?php
// 'product image' object
class ProductImage{

    // database connection and table
    private $conn;
    private $table_name = "product_im

    // object properties
    public $id;
    public $product_id;
    public $name;
    public $timestamp;

    // constructor
    public function __construct($db){
        $this->conn = $db;
    }
}
```

6.5 Connect to the database

Open products.php file. Replace `// class instances will be here` comment with the following code.

```
// get database connection
```

```
$database = new Database();
$db = $database->getConnection();

// initialize objects
$product = new Product($db);
$product_image = new ProductImage($db
```

6.6 Initialize action and pagination

Put the following code after the code on the previous section.

```
// to prevent undefined index notice
$action = isset($_GET['action']) ? $_

// for pagination purposes
$page = isset($_GET['page']) ? $_GET[
$records_per_page = 6; // set records
$from_record_num = ($records_per_page
```

6.7 Display messages based on action

We'll display messages basedon given action.

Put the following code after `include 'layout_header.php';` code.

```
echo "<div class='col-md-12'>";
    if($action=='added'){
        echo "<div class='alert alert
            echo "Product was added t
        echo "</div>";
    }

    if($action=='exists'){
        echo "<div class='alert alert
            echo "Product already exi
        echo "</div>";
    }
echo "</div>";
```

6.8 Request data from the database

Request data from the database. Put the following code after the code on the previous section.

```
```

```
// read all products in the database
$stmt=$product->read($from_record_num

// count number of retrieved products
$num = $stmt->rowCount();

// if products retrieved were more th
if($num>0){
    // needed for paging
    $page_url="products.php?";
    $total_rows=$product->count();

    // show products
    include_once "read_products_templ
}

// tell the user if there's no produc
else{
    echo "<div class='col-md-12'>";
        echo "<div class='alert alert
    echo "</div>";
}
```

6.9 Add "read" and "count" methods

The previous section will not work without the following code inside "objects/product.php" object file.

```
// read all products
function read($from_record_num, $reco

    // select all products query
    $query = "SELECT
                id, name, description
            FROM
                " . $this->table_name
            ORDER BY
                created DESC
            LIMIT
                ?, ?";

    // prepare query statement
    $stmt = $this->conn->prepare( $qu

    // bind limit clause variables
    $stmt->bindParam(1, $from_record_
    $stmt->bindParam(2, $records_per_

    // execute query
    $stmt->execute();

    // return values
    return $stmt;
```

```
}

// used for paging products
public function count(){

    // query to count all product rec
    $query = "SELECT count(*) FROM "

    // prepare query statement
    $stmt = $this->conn->prepare( $qu

    // execute query
    $stmt->execute();

    // get row value
    $rows = $stmt->fetch(PDO::FETCH_N

    // return count
    return $rows[0];
}
```

6.10 Template to display products

The products.php won't work without "read_products_template.php", so create that file and put the following code.

```
<?php
if(!isset($_SESSION['cart'])){
    $_SESSION['cart']=array();
}

while ($row = $stmt->fetch(PDO::FETCH_
    extract($row);

    // creating box
    echo "<div class='col-md-4 m-b-20

        // product id for javascript
        echo "<div class='product-id

        echo "<a href='product.php?id
            // select and show first
            $product_image->product_i
            $stmt_product_image=$prod

            while ($row_product_image
                echo "<div class='m-b
                    echo "<img src='u
                echo "</div>";
            }

            // product name
            echo "<div class='product
```

```
        echo "</a>";

        // add to cart button
        echo "<div class='m-b-10px'>"
            if(array_key_exists($id,
                echo "<a href='cart.p
                    echo "Update Cart
                echo "</a>";
            }else{
                echo "<a href='add_to
            }
        echo "</div>";

    echo "</div>";
}

include_once "paging.php";
?>
```

6.11 Add "readFirst()" method

Add "readFirst()" method in "objects/product_image.php" file. The previous section will not work without it.

```
// read the first product image relat
function readFirst(){

    // select query
    $query = "SELECT id, product_id,
            FROM " . $this->table_nam
            WHERE product_id = ?
            ORDER BY name DESC
            LIMIT 0, 1";

    // prepare query statement
    $stmt = $this->conn->prepare( $qu

    // sanitize
    $this->id=htmlspecialchars(strip_

    // bind prodcut id variable
    $stmt->bindParam(1, $this->produc

    // execute query
    $stmt->execute();

    // return values
    return $stmt;
}
```

6.12 Make "add to cart" button work

Open layout_footer.php file. Replace `<!-- custom script will be here -->` comment with the following code.

```
<script>
$(document).ready(function(){
    // add to cart button listener
    $('.add-to-cart-form').on('submit

        // info is in the table / sin
        var id = $(this).find('.produ
        var quantity = $(this).find('

        // redirect to add_to_cart.ph
        window.location.href = "add_t
        return false;
    });
});
</script>
```

6.13 Create pagination file

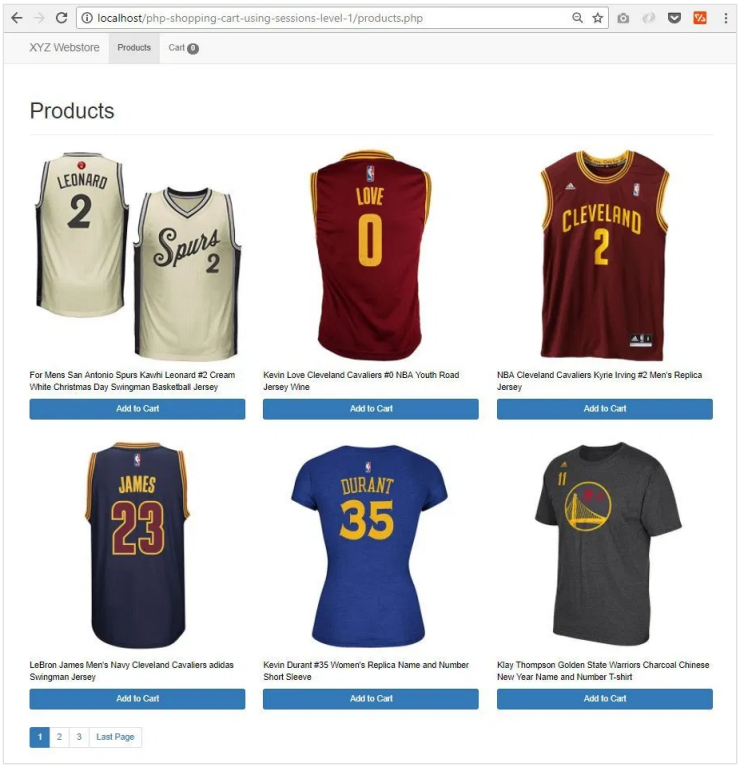
The read_products_template.php file won't work without the paging.php file. Create paging.php with the following code.

```
<?php
echo "<div class='col-md-12'>";

    echo "<ul class='pagination m-b-2

    // button for first page
    if($page>1){
        echo "<li><a href='{$page_url
            echo "First Page";
        echo "</a></li>";
    }

    $total_pages = ceil($total_rows /

    // range of links to show
    $range = 2;

    // display links to 'range of pag
    $initial_num = $page - $range;
    $condition_limit_num = ($page + $

    for ($x=$initial_num; $x<$conditi

        // be sure '$x is greater tha
        if (($x > 0) && ($x <= $total
```

```
            // current page
            if ($x == $page) {
                echo "<li class='acti
            }

            // not current page
            else {
                echo "<li><a href='{$
            }
        }
    }

    // button for last page
    if($page<$total_pages){
        echo "<li>";
            echo "<a href='" . $page_
                echo "Last Page";
            echo "</a>";
        echo "</li>";
    }

    echo "</ul>";
echo "</div>";
?>
```

6.14 Output

Run your products.php file on the browser http://localhost/php-shopping-cart-using-sessions-level-1/products.php [http://localhost/php-shopping-cart-using-sessions-level-1/products.php] . You should see an output like the image below.



7.0 HOW TO ADD TO CART?

7.1 Create add_to_cart.php

Create add_to_cart.php file because when 'Add to cart' button was clicked, that file with the following code inside will be executed.

```
<?php
// start session
session_start();

// get the product id
$id = isset($_GET['id']) ? $_GET['id']
$quantity = isset($_GET['quantity'])
$page = isset($_GET['page']) ? $_GET[

// make quantity a minimum of 1
$quantity=$quantity<=0 ? 1 : $quantit

// add new item on array
$cart_item=array(
    'quantity'=>$quantity
);
```

```
/*
 * check if the 'cart' session array
 * if it is NOT, create the 'cart' se
 */
if(!isset($_SESSION['cart'])){
    $_SESSION['cart'] = array();
}

// check if the item is in the array,
if(array_key_exists($id, $_SESSION['c
    // redirect to product list and t
    header('Location: products.php?ac
}

// else, add the item to the array
else{
    $_SESSION['cart'][$id]=$cart_item

    // redirect to product list and t
    header('Location: products.php?ac
}
?>
```

7.2 Create cart.php

Create cart.php with the following basic code.

```
<?php
// start session
session_start();

// connect to database
include 'config/database.php';

// include objects
include_once "objects/product.php";
include_once "objects/product_image.p

// get database connection
$database = new Database();
$db = $database->getConnection();

// initialize objects
$product = new Product($db);
$product_image = new ProductImage($db

// set page title
$page_title="Cart";

// include page header html
include 'layout_header.php';

// contents will be here

// layout footer
```

```
include 'layout_footer.php';
?>
```

7.3 Display message based on action

We'll display message on cart.php based on given action.

Put the following code after `include 'layout_header.php';` of the previous section.

```
$action = isset($_GET['action']) ? $_

echo "<div class='col-md-12'>";
    if($action=='removed'){
        echo "<div class='alert alert
            echo "Product was removed
        echo "</div>";
    }

    else if($action=='quantity_update
        echo "<div class='alert alert
            echo "Product quantity wa
        echo "</div>";
    }
echo "</div>";
```

7.4 Display cart items

Put the following code after the code of the previous section.

```
if(count($_SESSION['cart'])>0){

    // get the product ids
    $ids = array();
    foreach($_SESSION['cart'] as $id=
        array_push($ids, $id);
    }

    $stmt=$product->readByIds($ids);

    $total=0;
    $item_count=0;

    while ($row = $stmt->fetch(PDO::F
        extract($row);

        $quantity=$_SESSION['cart'][$
```

```
        $sub_total=$price*$quantity;

        // ================
        echo "<div class='cart-row'>"
            echo "<div class='col-md-

                echo "<div class='pro

                // update quantity
                echo "<form class='up
                    echo "<div class=
                    echo "<div class=
                        echo "<input
                            echo "<sp
                                echo
                            echo "</s
                    echo "</div>";
                echo "</form>";

                // delete from cart
                echo "<a href='remove
                    echo "Delete";
                echo "</a>";
            echo "</div>";

            echo "<div class='col-md-
                echo "<h4>&#36;" . nu
            echo "</div>";
        echo "</div>";
        // ================

        $item_count += $quantity;
        $total+=$sub_total;
    }

    echo "<div class='col-md-8'></div
    echo "<div class='col-md-4'>";
        echo "<div class='cart-row'>"
            echo "<h4 class='m-b-10px
            echo "<h4>&#36;" . number
            echo "<a href='checkout.p
                echo "<span class='gl
            echo "</a>";
        echo "</div>";
    echo "</div>";

}

// no products were added to cart
else{
    echo "<div class='col-md-12'>";
        echo "<div class='alert alert
            echo "No products found i
        echo "</div>";
    echo "</div>";
}
```

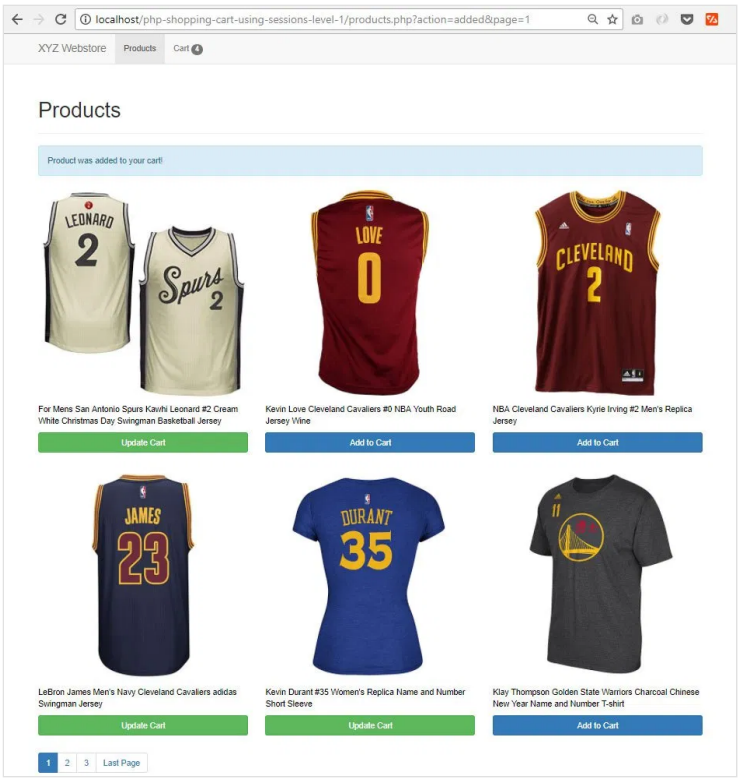
7.5 Read products by IDs

The previous section will not work without the following "readByIds()" method inside "objects/product.php" file.

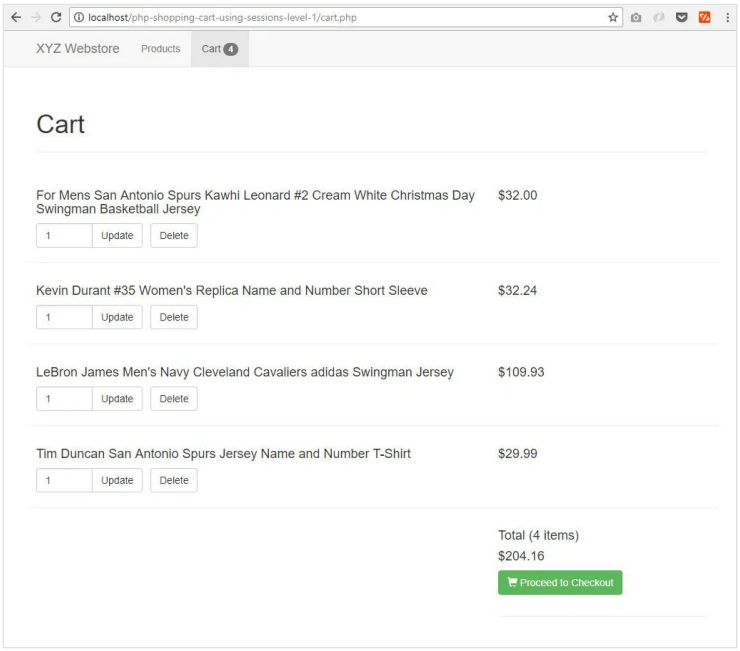
```
// read all product based on product
// reference http://stackoverflow.com
public function readByIds($ids){

    $ids_arr = str_repeat('?,', count

    // query to select products
    $query = "SELECT id, name, price

    // prepare query statement
    $stmt = $this->conn->prepare($que

    // execute query
    $stmt->execute($ids);

    // return values from database
    return $stmt;
}
```

7.6 Output

When user click the "Add to cart" button.



Go to the cart page by clicking the "Cart" option on the navigation bar.



8.0 HOW TO UPDATE CART?

8.1 Update cart quantity with JavaScript

We have the 'update' button on cart.php file. When that button was clicked, a javascript code is triggered.

Place the following code inside `$(document).ready(function(){` of `layout_footer.php` file.

```
// update quantity button listener
$('.update-quantity-form').on('submit

    // get basic information for upda
    var id = $(this).find('.product-i
    var quantity = $(this).find('.car

    // redirect to update_quantity.ph
    window.location.href = "update_qu
    return false;
});
```

8.2 PHP script to update cart

The previous section will not work without this file.

Create `update_quantity.php` file. Place the following code and save it.

```
<?php
session_start();

// get the product id
$id = isset($_GET['id']) ? $_GET['id']
$quantity = isset($_GET['quantity'])

// make quantity a minimum of 1
$quantity=$quantity<=0 ? 1 : $quantit

// remove the item from the array
unset($_SESSION['cart'][$id]);

// add the item with updated quantity
$_SESSION['cart'][$id]=array(
    'quantity'=>$quantity
);

// redirect to product list and tell
header('Location: cart.php?action=qua
?>
```

8.3 How to remove product on cart?

We have the 'remove' button on cart.php file. When that button was clicked, it will trigger remove_from_cart.php file.

Create remove_from_cart.php file. Place the following code and save it.

```
<?php
// start session
session_start();

// get the product id
$id = isset($_GET['id']) ? $_GET['id']
$name = isset($_GET['name']) ? $_GET[

// remove the item from the array
unset($_SESSION['cart'][$id]);
```

```
// redirect to product list and tell
header('Location: cart.php?action=rem
?>
```

8.4 Create the checkout page

The checkout page looks like the cart page but the items cannot be updated or removed. It just like the summary of orders. Create checkout.php with the following code.

```
<?php
// start session
session_start();

// connect to database
include 'config/database.php';

// include objects
include_once "objects/product.php";
include_once "objects/product_image.p

// get database connection
$database = new Database();
$db = $database->getConnection();

// initialize objects
$product = new Product($db);
$product_image = new ProductImage($db

// set page title
$page_title="Checkout";

// include page header html
include 'layout_header.php';

if(count($_SESSION['cart'])>0){

    // get the product ids
    $ids = array();
    foreach($_SESSION['cart'] as $id=
        array_push($ids, $id);
    }

    $stmt=$product->readByIds($ids);

    $total=0;
    $item_count=0;

    while ($row = $stmt->fetch(PDO::F
        extract($row);
```

```
        $quantity=$_SESSION['cart'][$
        $sub_total=$price*$quantity;

        //echo "<div class='product-i
        //echo "<div class='product-n

        // =================
        echo "<div class='cart-row'>"
            echo "<div class='col-md-

                echo "<div class='pro
                echo $quantity>1 ? "<

            echo "</div>";

            echo "<div class='col-md-
                echo "<h4>&#36;" . nu
            echo "</div>";
        echo "</div>";
        // =================

        $item_count += $quantity;
        $total+=$sub_total;
    }

    // echo "<div class='col-md-8'></
    echo "<div class='col-md-12 text-
        echo "<div class='cart-row'>"
            if($item_count>1){
                echo "<h4 class='m-b-
            }else{
                echo "<h4 class='m-b-
            }
            echo "<h4>&#36;" . number
            echo "<a href='place_orde
                echo "<span class='gl
            echo "</a>";
        echo "</div>";
    echo "</div>";

}

else{
    echo "<div class='col-md-12'>";
        echo "<div class='alert alert
            echo "No products found i
        echo "</div>";
    echo "</div>";
}

include 'layout_footer.php';
?>
```

8.5 Create place_order.php

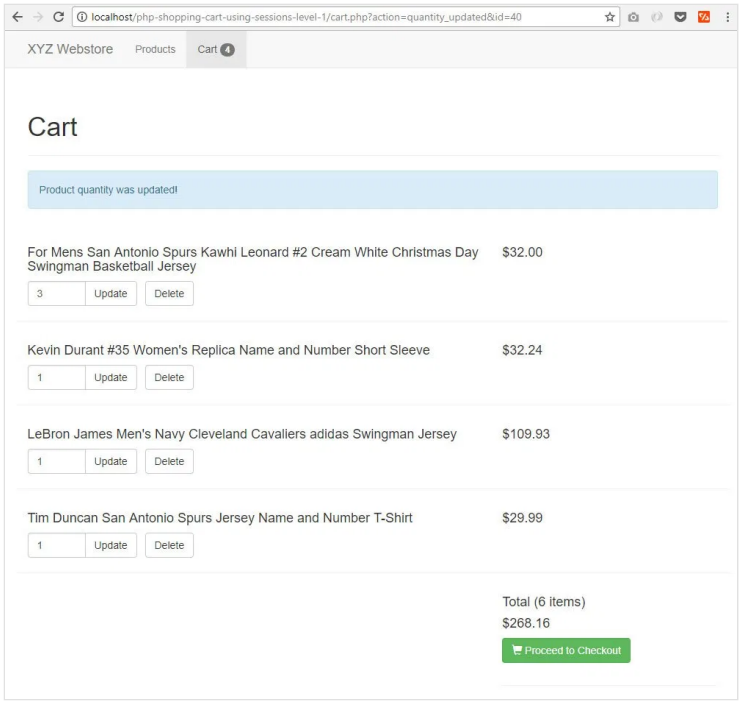
We'll use this file to show a "thank you" message and remove all items in the cart.

Create place_order.php file. Place the following code.

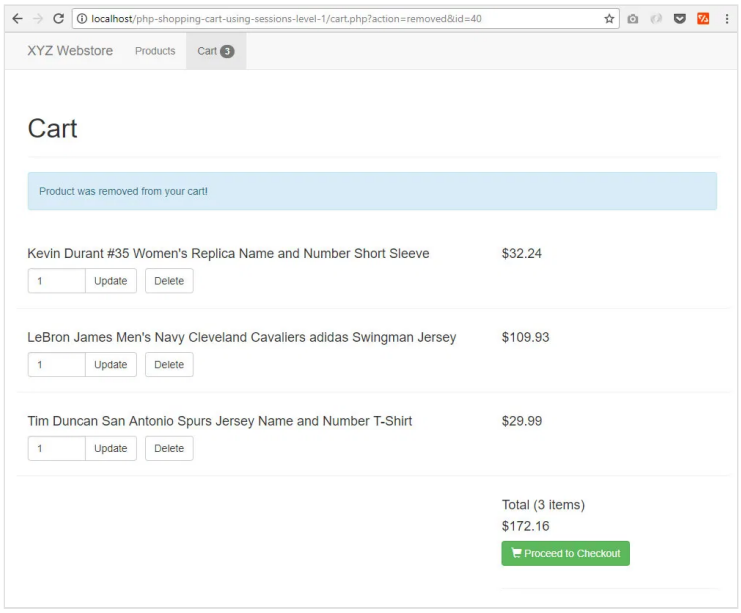
```
<?php
// start session
session_start();

// remove items from the cart
session_destroy();

// set page title
$page_title="Thank You!";

// include page header HTML
include_once 'layout_header.php';

echo "<div class='col-md-12'>";

    // tell the user order has been p
    echo "<div class='alert alert-suc
        echo "<strong>Your order has
    echo "</div>";

echo "</div>";

// include page footer HTML
include_once 'layout_footer.php';
?>
```

8.6 Output

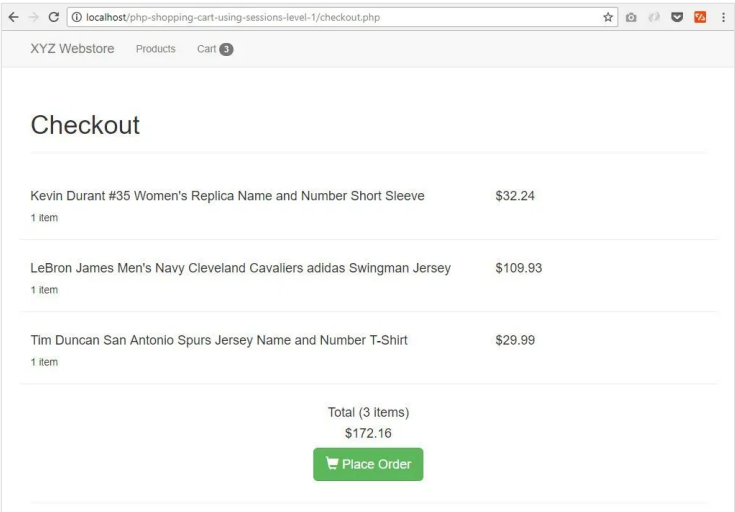
When user click the "Update" button in the cart page.



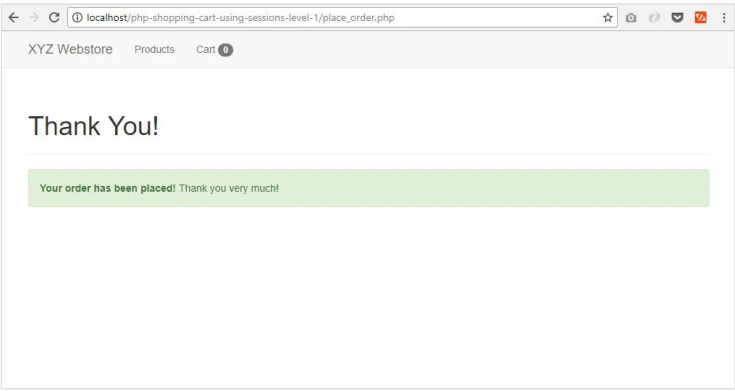
If user click the "Delete" button.



The checkout page.



When user click the "Place Order" button.



9.0 HOW TO MAKE THE PRODUCT PAGE?

9.1 Create product.php

Create product.php with the following basic code.

```
<?php
// start session
session_start();

// include classes
include_once "config/database.php";
include_once "objects/product.php";
include_once "objects/product_image.p

// get database connection
```

```
$database = new Database();
$db = $database->getConnection();

// initialize objects
$product = new Product($db);
$product_image = new ProductImage($db

// include page header HTML
include_once 'layout_header.php';

// content will be here

// include page footer HTML
include_once 'layout_footer.php';
?>
```

9.2 Read product details

Put the following code after "$product_image = new ProductImage($db);" code of the previous section.

```
// get ID of the product to be edited
$id = isset($_GET['id']) ? $_GET['id']

// set the id as product id property
$product->id = $id;

// to read single record product
$product->readOne();

// set page title
$page_title = $product->name;

// product thumbnail will be here
```

9.3 Read one product method

The previous section will not work without the "readOne()" method. Add the following method inside "objects/product.php" file.

```
// used when filling up the update pr
function readOne(){

    // query to select single record
    $query = "SELECT
                name, description, pr
            FROM
                " . $this->table_name
```

```
            WHERE
                id = ?
            LIMIT
                0,1";

    // prepare query statement
    $stmt = $this->conn->prepare( $qu

    // sanitize
    $this->id=htmlspecialchars(strip_

    // bind product id value
    $stmt->bindParam(1, $this->id);

    // execute query
    $stmt->execute();

    // get row values
    $row = $stmt->fetch(PDO::FETCH_AS

    // assign retrieved row value to
    $this->name = $row['name'];
    $this->description = $row['descri
    $this->price = $row['price'];
}
```

9.4 Display product thumbnails

When these product thumbnails were hovered, it displayes a larger version of the image. It is Amazon-style.

Open product.php file. Replace `// product thumbnail will be here` comment with the following code.

```
// set product id
$product_image->product_id=$id;

// read all related product image
$stmt_product_image = $product_image-

// count all relatd product image
$num_product_image = $stmt_product_im

echo "<div class='col-md-1'>";
    // if count is more than zero
    if($num_product_image>0){
        // loop through all product i
        while ($row = $stmt_product_i
            // image name and source
            $product_image_name = $ro
```

```
            $source="uploads/images/{
            echo "<img src='{$source}
        }
    }else{ echo "No images."; }
echo "</div>";

// product image will be here
```

9.5 Read images related to product

The previous section section will not work without the "readByProductId()" method inside "objects/product_image.php" file.

```
// read all product image related to
function readByProductId(){

    // select query
    $query = "SELECT id, product_id,
            FROM " . $this->table_nam
            WHERE product_id = ?
            ORDER BY name ASC";

    // prepare query statement
    $stmt = $this->conn->prepare( $qu

    // sanitize
    $this->product_id=htmlspecialchar

    // bind prodcut id variable
    $stmt->bindParam(1, $this->produc

    // execute query
    $stmt->execute();

    // return values
    return $stmt;
}
```

9.6 Display product image

Only one product image are displayed at a time. This part displays the larger product image based on the hovered product thumbnail.

Open product.php file. Replace `// product image will be here` comment with the following code.

```
echo "<div class='col-md-4' id='produ

    // read all related product image
    $stmt_product_image = $product_im
    $num_product_image = $stmt_produc

    // if count is more than zero
    if($num_product_image>0){
        // loop through all product i
        $x=0;
        while ($row = $stmt_product_i
            // image name and source
            $product_image_name = $ro
            $source="uploads/images/{
            $show_product_img=$x==0 ?
            echo "<a href='{$source}'
                echo "<img src='{$sou
            echo "</a>";
            $x++;
        }
    }else{ echo "No images."; }
echo "</div>";

// product details will be here
```

9.7 Make image hover work

Put the following jQuery code inside "$(document).ready(function(){" of layout_footer.php file.

```
// change product image on hover
$(document).on('mouseenter', '.produc
    var data_img_id = $(this).attr('d
    $('.product-img').hide();
    $('#product-img-'+data_img_id).sh
});
```

9.8 Display product details

This part display product price, description and category.

Open product.php file. Replace `// product details will be here` comment with the following code.

```
echo "<div class='col-md-5'>";
```

```
    echo "<div class='product-detail'
    echo "<h4 class='m-b-10px price-d

    echo "<div class='product-detail'
    echo "<div class='m-b-10px'>";
        // make html
        $page_description = htmlspeci

        // show to user
        echo $page_description;
    echo "</div>";

    echo "<div class='product-detail'
    echo "<div class='m-b-10px'>{$pro

echo "</div>";
```

9.9 Render 'Cart' button

Now we will display 'Add to cart' button if the product is not yet added to cart. Else, we will display 'update cart' button.

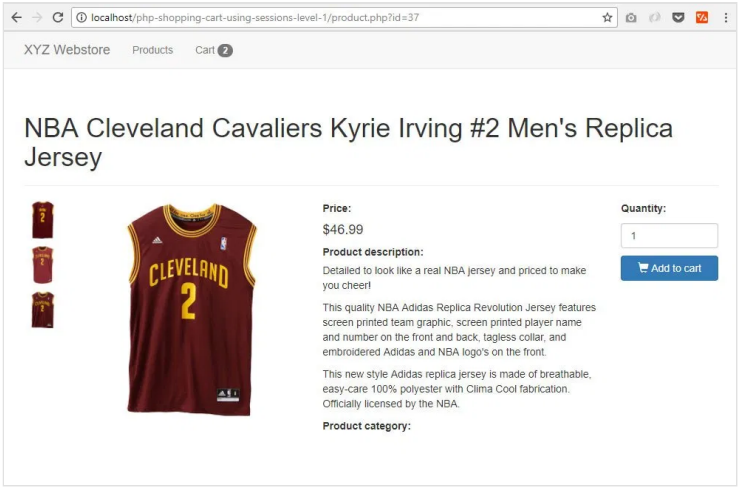
Place the following code after the previous section's code.

```
echo "<div class='col-md-2'>";

    // if product was already added i
    if(array_key_exists($id, $_SESSIO
        echo "<div class='m-b-10px'>T
        echo "<a href='cart.php' clas
            echo "Update Cart";
        echo "</a>";

    }

    // if product was not added to th
    else{

        echo "<form class='add-to-car
            // product id
            echo "<div class='product

            echo "<div class='m-b-10p
            echo "<input type='number

            // enable add to cart but
            echo "<button style='widt
                echo "<span class='gl
            echo "</button>";

        echo "</form>";
```

```
    }
echo "</div>";
```

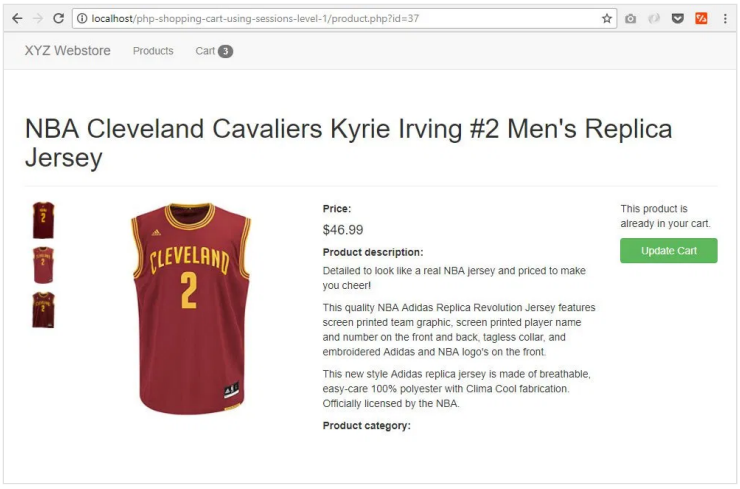
9.10 Output

When user click on any product image in products.php page, he will land to a product page that looks like the image below.



If user hovers on any of those thumbnail or small images, the big image will change as well. The "Add to cart" button is working as well.

Here's the output when the product is already added to cart.



If user click the "Update Cart" button, he will land on the cart page where he can update the cart quantity.

10.0 WHAT PEOPLE SAY ABOUT THIS CODE?

I'm so glad that this code delights other people. The following are some of them from the comments section!

★★★★★ "Hey Mike, my name is Leonardo from Argentina. I've been reading your blog since like 4 months from now, and I really must say: your tutorials are very good, they has helped me in many of my works... Well, thank you very much man. I really admire your work." ~ Leonardo [https://www.codeofaninja.com/wp-content/uploads/2015/03/leonardo-codeofaninja-review.jpg]

★★★★★ "Man, your tut's are awesome. Im so glad ive found your blog. Big respect!" ~ Milos [https://www.codeofaninja.com/wp-content/uploads/2015/03/milos-codeofaninja-review.jpg]

★★★★★ "I bought your level-2 source code and it was so good, very big help for me. It was worth it. Thank you very much!" ~ Ashley Deanna Plata [https://www.codeofaninja.com/wp-content/uploads/2015/03/ashley-deanna-plata-codeofaninja-review.jpg]

★★★★★ "Hello, This is a great script and I have paid for your work (it Worth it)." ~ Louis Blais [https://www.codeofaninja.com/wp-content/uploads/2015/03/louis-blais-codeofaninja-review.jpg]

★★★★★ "Words can't express how grateful I am for the work and the articles you post, had some troubles with doing somethings but your articles as per usual hit the hammer right on the head. They are a great way for expanding upon later too!" ~ Jeremy Smith [https://www.codeofaninja.com/wp-content/uploads/2015/03/jeremy-smith-codeofaninja-review.jpg]

11.0 HOW TO RUN THE SOURCE CODE?

We highly recommend for you to follow and study our well-detailed, step-by-step tutorial above first. Nothing beats experience when it comes to learning.

But we believe you will learn faster if you'll see the final source code as well. We consider it as your additional guide.

Imagine the value or skill upgrade it can bring you. The additional income you can get from your work, projects or business. The precious time you save. Isn't that what you want?

By now, you need to download our source codes. To do it, use any download buttons in the next few sections below.

Once you downloaded the source codes, here's how you can run it.

1. Extract the files to your server directory.
2. Go to your PhpMyAdmin, create a database with a name "shop_cart_sessions_1".

3. Import the "shop_cart_sessions_1.sql" file located in the "README" folder.
4. You might need to change database credentials in /config/database.php
5. Run "products.php", this is the main PHP file. We do not have index.php

12.0 DOWNLOAD THE LEVEL 1 SOURCE CODE

FEATURE	LEVEL 1
Learn to code a simple cart function	YES
List all products from MySQL database	YES
Pagination on products list page	YES
Add to cart action button	YES
Remove from cart action button	Yes
Update cart quantity	YES
Checkout Page	YES
Place order / Thank you page	YES
Amazon-style product details page	YES
Change image on hover of thumbnail	YES
Show message about a product added to cart	YES
Show message about a product removed from cart	YES
Navigation bar highlights which page is selected	YES
Cart link shows count of products added in the cart	YES

Show message if no products found in database	YES
Show message if no product found in cart	YES
Bootstrap enabled UI	YES
Cart page that lists all products added to cart	YES
Auto-compute total cost of all products added to cart	YES
PDO extension used	YES
Step by step tutorial	YES
Free source code updates	YES
Free support for 6 months	YES

$19.99 – DOWNLOAD LEVEL 1 SOURCE CODE NOW [#]

13.0 DOWNLOAD THE LEVEL 2 SOURCE CODE

FEATURE	LEVEL 2
All features of LEVEL 1 source code	YES
Navigation bar has drop down of product categories	YES
Highlight selected category in drop down	YES
Categories are retrieved from the database	YES
Show products by category	YES
List products under a category with pagination	YES

Search product	YES
Search results with pagination	YES
Search box located on upper right corner of navigation bar	YES
Search box requires search term before clicking the search button	YES
Add to cart action button	YES
Quantity text box beside the add to cart button	YES
Quantity text box required to be a number	YES
Quantity text box required to have a minimum value of 1, negative value not allowed	YES
Remember the page number where the user clicked the "Add to cart" button	YES
Quantity drop down options based on available stock	YES
Well formatted money value	YES
Check out button with cart icon	YES
Product image viwable in lightbox	YES
Shows number of stock left	YES
Stock decreases once checked out	YES
Order saved in orders and order_items table in the database	YES
Emtpy cart button	YES
Emtpy cart confirmation pop up	YES
Bootstrap enabled UI	YES

Cart page that lists all products added to cart	YES
Quantity text box beside update quantity button	YES
Show price, category and stocks left in product list page	YES
Auto-compute total cost of all products added to cart	YES
Used PDO bindParam() to prevent SQL injection in MySQL queries	YES
Used PHP htmlspecialchars() to prevent XSS attacks	YES
SQL file is in the "dev" folder	YES
PDO extension used	YES
Free code updates	YES
Free support for 6 months	YES

$39.99 – DOWNLOAD LEVEL 2 SOURCE CODE NOW [#]

14.0 PHP SHOPPING CART MODULE

You can download our "PHP Shopping Cart & Ordering Module" source code. It has several features you need to learn more about how to handle the users, shopping cart, and ordering using the PHP & MySQL technology. CLICK HERE TO LEARN MORE [https://www.codeofaninja.com/2016/07/php-online-shopping-cart-source-code.html]

15.0 PHP SHOPPING CART SYSTEM

You can download our "PHP Shopping Cart System" source code as well. Many of you requested this type of source code and not it is here!

You needed a shopping cart system with user management (merchant and customer), product management, order management, security and more features based on our source codes here in codeofaninja.com. CLICK HERE TO LEARN MORE [https://www.codeofaninja.com/shop-cart-code] .

16.0 WHAT'S NEXT?

You have two options:

Option #1:
We just learned how to code an online shopping cart from scratch using PHP SESSIONS. But did you know that we can create the almost the same functions using another PHP mechanism called COOKIES?

If you're excited to learn this new concept, let us go to the next tutorial: PHP Shopping Cart Tutorial Using COOKIES [https://www.codeofaninja.com/2014/09/php-shopping-cart-tutorial-using-cookies.html]

Option #2:
This next tutorial is the start of our JavaScript programming journey. Go to our next tutorial:

How To Create a Simple REST API in PHP – Step By Step Guide! [https://www.codeofaninja.com/2017/02/create-simple-rest-api-in-php.html]

17.0 RELATED TUTORIALS

GETTING STARTED

Learn the basics of web programming.

How to Run a PHP Script? [https://www.codeofaninja.com/2013/06/how-to-run-a-php-script.html]

Bootstrap Tutorial for Beginners [https://www.codeofaninja.com/2014/05/bootst tutorial-beginners-step-step.html]

jQuery Tutorial for Beginners [https://www.codeofaninja.com/2013/10/jquery step-by-step-tutorial-for-beginners.html]

jQuery UI Tutorial for Beginners [https://www.codeofaninja.com/2013/10/jquery ui-tutorial-for-beginners.html]

PHP PROGRAMMING TUTORIALS

Start something awesome with PHP!

PHP CRUD Tutorial for Beginners [https://www.codeofaninja.com/2011/12/php-and-mysql-crud-tutorial.html]

PHP OOP CRUD Tutorial [https://www.codeofaninja.com/2014/06/php-object-oriented-crud-example-oop.html]

PHP Login Script with Session [https://www.codeofaninja.com/2013/03/php-login-script.html]

PHP Shopping Cart Tutorial [https://www.codeofaninja.com/2015/08/simple

php-mysql-shopping-cart-tutorial.html]

PHP Shopping Cart Tutorial 2.0 [https://www.codeofaninja.com/2013/04/shopp cart-in-php.html]

REST API TUTORIALS

Welcome to the new world of web development!

PHP REST API Tutorial [https://www.codeofaninja.com/2017/02/create simple-rest-api-in-php.html]

PHP REST API Authentication Example [https://www.codeofaninja.com/2018/09/rest-api-authentication-example-php-jwt-tutorial.html]

AJAX CRUD Tutorial [https://www.codeofaninja.com/2015/06/php-crud-with-ajax-and-oop.html]

PHP WEB APP SOURCE CODES

Scripts that will help you build a web app.

PHP SHOPPING CART SYSTEM [https://www.codeofaninja.com/2016/04/php-shopping-cart-source-code-download.html]

Download By Modules:

PHP Login & Registration Module [https://www.codeofaninja.com/2016/05/php-login-system-tutorial.html]

PHP Shopping Cart Module [https://www.codeofaninja.com/2016/07/php-online-shopping-cart-source-code.html]

PHP Product Catalog Module [https://www.codeofaninja.com/2016/07/php-product-catalog-script.html]

PHP Content Management Module [https://www.codeofaninja.com/2016/07/php-web-page-content-management-module.html]

PHP Contact Form Module [https://www.codeofaninja.com/2016/07/php-contact-form-messages-module.html]

PHP PayPal Integration Module [https://www.codeofaninja.com/2016/04/paypal-integration-in-php.html]

MORE SCRIPTS

More code examples that can be useful for you!

Shopping Cart Tutorial using COOKIES [https://www.codeofaninja.com/2014/09/php-shopping-cart-tutorial-using-cookies.html]

Extra Tutorials [https://www.codeofaninja.com/extras]

18.0 SOME NOTES

#1 Found An Issue?

If you found a problem with this code, please write a comment below. Please be descriptive about your issue. Please provide the error messages, screenshots (or screencast) and your test URL. Thanks!

Before you write a comment, remember to read this guide [https://www.codeofaninja.com/how-to-write-a-great-comment-on-codeofaninja-com] and our code of conduct [https://www.codeofaninja.com/code-of-conduct] .

#2 Become a true Ninja!

We constantly add new tutorials and improve our existing tutorials and source codes. Be one of the first to know an update by subscribing to our FREE newsletter. CLICK HERE TO

SUBSCRIBE FOR FREE!
[https://www.codeofaninja.com/subscribe]

#3 Thank You!

Thank you for studying our tutorial about PHP Shopping Cart Tutorial using SESSIONS!

If you think our work is helpful, please share it to your friends!

Share this entry

Before you write a comment, please read this guide and our code of conduct.