**Web Scripting Coursework**

**API**

**Index.php**

<?php

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

require 'products.php';

/\*\*

\* A Resoruce List for Products to include HTTP Verbs

\* Switch Statement used for HTTP Verbs (GET, POST, DELETE, and PATCH)

\* Server Request\_Method is an array which contains information of paths and locations

\* Routing for Products Starts Here

\*/

// Remove white space (trim) and check if variable is set and not empty(isset)

$pathParts = trim(isset($\_SERVER['PATH\_INFO'])? $\_SERVER['PATH\_INFO'] : '/' , '/');

$pathParts = explode('/', $pathParts); // that gets the path

switch ($pathParts[0]) {

case 'products':

switch ($\_SERVER['REQUEST\_METHOD'])

{

// Gets List of all products or single product using GET Method

case 'GET':

if (empty($pathParts[1])){

if (!empty($\_GET['category'])) {

echo getProducts($\_GET['category']);

} else {

echo getProducts();

}

}else{

echo getProduct($pathParts[1]);

}

break;

// Create product on database using POST and view it using GET Method

case 'POST':

postProducts();

break;

// Delete requrested product from database using DELETE Method

case 'DELETE':

if (empty($pathParts[1])) {

echo("you can delete everthing!!");

}else{

deleteProducts($pathParts[1]);

}

break;

// Allow the user to Edit/Update Products on Database

case 'PATCH':

if (isset($pathParts[1])) {

updateProducts($pathParts[1]);

}

}

break;

// End of Switch Statement

}

**Products.php**

<?php

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

// This will produce a fatel error if the file or database crashes

require('DBConnect/db.php');

/\*\*

\* Get Products Function creates new database if it already does not exist on the server.

\* SQL Queries are used to select list of all products from Database

\* JASON\_Encoding used to transfer data between server and client side application

\* Create new database if it doesn't exists

\*/

function getProducts($cat = null)

{

$database = new DB();

$bind = null;

$sql = "SELECT \* FROM Products";

if ($cat) {

$cat = $database->query("SELECT categoryID from category where categoryName = '$cat'");

$sql .= " WHERE categoryID = ".$cat[0]['categoryID'];

}

return json\_encode($database->query($sql));

}

/\*\*

\* @ paramater: $ID Return single product by given ID using SQL query

\* Get Product Function creates new database if it already does not exist on the server.

\* JASON\_Encoding used to transfer data between server and client side application

\*/

function getProduct($ID){

$database = new DB();

$bind = null;

$sql = "SELECT \* FROM Products WHERE ProductID = $ID";

return json\_encode($database->query($sql));

}

/\*\*

\* This function post the record to the database

\* Receive record in JSON Formatted String from client side and decode it and convert the product into an object which can go onto the database

\* The "php://input" allows the user to read raw POST data.

\* INSERT all the required information on the the table and then database usign sql query

\* strlen will return the length of given string which is assigned to be greater than 0

\* foreach is used to provide an easy way to iterate over array

\*/

function postProducts()

{

$data = json\_decode(file\_get\_contents("php://input"));

if (strlen($data->productName) > 0) {

$sql = "INSERT INTO Products(ProductName, CategoryID, ProductDescription, ProductPrice, ProductStockLevel, ProductImage) VALUES (";

foreach ($data as $key => $value) {

$sql .= ":".$key.", ";

}

$sql = rtrim($sql, ", ");

$sql .= ");";

$dataItems = array();

//Concatenation assignment

// Joing entities together

foreach ($data as $key => $value) {

$dataItems[":".$key] = "".$value;

}

$database = new DB();

$database->query($sql, $dataItems);

}

}

/\*\*

\* Delete Product Function

\* Create new database if it doesn't exists

\* @productID: delete the product using productID. Product will be deleted by specifying productID

\*/

function deleteProducts($productID) {

$database = new DB();

$sql = "DELETE FROM Products WHERE ProductID= $productID";

$database->query($sql);

}

/\*\*

\* Update Product Function

\* Receive record in JASON Formatted String from client side and decode it and convert the product into an object which can go onto the database

\* The "php://input" allows the user to read raw POST data.

\* @parameter: ProductID: update the record/product by specifying produc id

\* var\_dump displays information about one or more expression (productName and productPrice)

\* Create new database if it doesn't exists

\*/

function updateProducts($ProductID) {

$info = json\_decode(file\_get\_contents("php://input"), true);

$database = new DB();

var\_dump(file\_get\_contents("php://input"));

$sql = "UPDATE Products SET ProductName = '" .$info['ProductName'] . "', Price = '".$info['ProductPrice']. "' WHERE ProductID = $ProductID";

$a = $database->pdo()->prepare($sql);

$a->execute();

}

**Search.php**

<?php

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

require('DBConnect/db.php');

class products {

public $ProductID;

public $CategoryID;

public $ProductName;

public $ProductDescription;

public $ProductPrice;

public $ProductStockLevel;

public $ProductImage;

}

$search = $\_GET["input"];

try {

$database = new DB();

$bind = null;

$sql = "SELECT \* FROM Products WHERE ProductName LIKE :searchterm";

$a = $database->pdo()->prepare($sql);

$a->execute(array(":searchterm" => "%" . $search . "%"));

$rows = $a->fetchAll(PDO::FETCH\_CLASS, "products");

foreach($rows as $row) {

print "<section class='layout'><p>ProductID: ".$row->ProductID ."</p><h4> ".$row->ProductName."</h4><p class='writeUp'>".$row->ProductDescription."</p><p class='writeUp'>£".$row->ProductPrice."</p><p class='writeUp'>Stock Level:".$row->ProductStockLevel."</p><img class='productImage' src=../images/productImages/".$row->ProductImage."><button id=".$row->ProductID."> Add To Cart </button></section>";

}

}

catch (PDOException $e)

{

echo $e;

}

**Category (index.php)**

<?php

require 'category.php';

/\* Author UP687776 \*/

/\* Web Script Programing \*/

/\* Rick Bookes && Kit Lester \*/

/\* 2014 && 2015 \*/

/\*\*

\* A Resoruce List for Category to include HTTP Verbs

\* Switch Statement used for HTTP Verbs (GET, POST, DELETE, and PATCH)

\* Server Request\_Method is an array which contains information of paths and locations

\* Routing for Products Starts Here

\*/

$pathParts = trim(isset($\_SERVER['PATH\_INFO'])? $\_SERVER['PATH\_INFO'] : '/' , '/');

$pathParts = explode('/', $pathParts);

switch ($pathParts[0]) { //2

case 'category':

switch ($\_SERVER['REQUEST\_METHOD'])

{

case 'GET':

if (empty($pathParts[1])){

echo getCategories();

}else{

echo getCategory($pathParts[1]);

}

break;

case 'POST':

postCategory();

break;

case 'DELETE':

if (empty($pathParts[1])) {

echo("you can delete everthing!!");

}else{

deleteCategories($pathParts[1]);

}

break;

case 'UPDATE':

if (isset($pathParts[1])) {

updateCategory($pathParts[1], $\_POST);

}

}

break;

} //2

**Category.php**

<?php

/\* Author UP687776 \*/

/\* Web Script Programing \*/

/\* Rick Bookes && Kit Lester \*/

/\* 2014 && 2015 \*/

require('../DBConnect/db.php');

/\*\*

\* JASON\_Encoding used to transfer data between server and client side application

\*/

function getCategories() // this is for all categories, change it to categories from category later

{

$database = new DB();

$sql = "SELECT \* FROM Category";

return json\_encode($database->query($sql));

}

/\*\*

\* @ paramater: $ID Return single category by given ID using SQL query

\* JASON\_Encoding used to transfer data between server and client side application

\*/

function getCategory($ID) {

$database = new DB();

$bind = null;

$sql = "SELECT \* FROM Category WHERE CategoryID = $ID";

return json\_encode($database->query($sql));

// add (or update) the record to the database

$rows = $DB -> query ($query -> $binds);

}

/\*\*

\* This function post the record to the database

\* Receive record in JSON Formatted String from client side and decode it and convert the \* category into an object which can go onto the database

\* The "php://input" allows the user to read raw POST data.

\* INSERT all the required information on the the table and then database usign sql query

\* strlen will return the length of given string which is assigned to be greater than 0

\* foreach is used to provide an easy way to iterate over array

\*/

function postCategory()

{

$data = json\_decode(file\_get\_contents("php://input"));

if (strlen($data->categoryName) > 0) {

$sql = "INSERT INTO Category(CategoryName) VALUES (";

foreach ($data as $key => $value) {

$sql .= ":".$key.", ";

}

$sql = rtrim($sql, ", ");

$sql .= ");";

$dataItems = array();

//Concatenation assignment

// Joing entities together

foreach ($data as $key => $value) {

$dataItems[":".$key] = "".$value;

}

$database = new DB();

$database->query($sql, $dataItems);

}

}

/\*\*

\* Delete Category Function

\* Create new database if it doesn't exists

\* @CategoryID: delete the product using categoryID. Category will be deleted by

\* specifying categoryID

\*/

function deleteCategory($CategoryID) {

$database = new DB();

$sql = "DELETE FROM Category WHERE CategoryID= $CategoryID";

$database->query($sql);

}

**Contact(index.php)**

<?php

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

// This will produce a fatel error if there is an error on products.php or database crashes

require 'contact.php';

\*/

// Remove white space (trim) and check if variable is set and not empty(isset)

$pathParts = trim(isset($\_SERVER['PATH\_INFO'])? $\_SERVER['PATH\_INFO'] : '/' , '/');

$pathParts = explode('/', $pathParts);

switch ($pathParts[0]) {

case 'contact':

switch ($\_SERVER['REQUEST\_METHOD'])

{

// Create product on database using POST and view it using GET Method

case 'POST':

postContactQueries();

break;

}

break;

}

**contact.php**

<?php

require('../DBConnect/db.php');

function postContactQueries()

{

$data = json\_decode(file\_get\_contents("php://input"));

if (strlen($data->Name) > 0) {

$sql = "INSERT INTO Contact(Name, Email, ContactNo, Query) VALUES (";

foreach ($data as $key => $value) {

$sql .= ":".$key.", ";

}

$sql = rtrim($sql, ", ");

$sql .= ");";

$dataItems = array();

//Concatenation assignment

// Joing entities together

foreach ($data as $key => $value) {

$dataItems[":".$key] = "".$value;

}

$database = new DB();

$database->query($sql, $dataItems);

}

}

**Order (index.php)**

<?php

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

// This will produce a fatel error if there is an error on products.php or database crashes

require 'orders.php';

/\*\*

\* A Resoruce List for Products to include HTTP Verbs

\* Switch Statement used for HTTP Verbs (GET, POST, DELETE, and PATCH)

\* Server Request\_Method is an array which contains information of paths and locations

\* Routing for Products Starts Here

\*/

// Remove white space (trim) and check if variable is set and not empty(isset)

$pathParts = trim(isset($\_SERVER['PATH\_INFO'])? $\_SERVER['PATH\_INFO'] : '/' , '/');

$pathParts = explode('/', $pathParts);

switch ($\_SERVER['REQUEST\_METHOD'])

{

// Gets List of all products or single product using GET Method

case 'GET':

if (empty($pathParts[1])){

echo getOrders();

}else{

echo getOrder($pathParts[1]);

}

break;

// Create product on database using POST and view it using GET Method

case 'POST':

postOrders(json\_decode(file\_get\_contents('php://input')));

break;

// Delete requrested product from database using DELETE Method

case 'DELETE':

if (empty($pathParts[1])) {

echo("you can delete everthing!!");

}else{

deleteOrders($pathParts[1]);

}

break;

// Allow the user to Edit/Update Products on Database

case 'PATCH':

if (isset($pathParts[1])) {

updateOrders($pathParts[1]);

}

}

// End of Switch Statement

function postOrders($postBody)

{

//var\_dump($postBody);

foreach ($postBody as $key => $value) {

echo $value;

$db = new DB();

$args = array('quantity' => 1,

'productId' => $value);

$db -> query("INSERT INTO `Order` VALUES ('placeholder' ,:quantity , :productId)", $args);

//

//

}

}

// $postBody is an array of product IDs

// insert a new order along with the product IDs

**order.php**

<?php

// /\*\*

// \* Author: UP687776

// \* Web Script Programming

// \* Rick Bookes && Kit Lester

// \* 2014 && 2015

// \*/

// This will produce a fatel error if the file or database crashes

require('../DBConnect/db.php');

// require\_once

/\*\*

\* Get Products Function creates new database if it already does not exist on the server.

\* SQL Queries are used to select list of all products from Database

\* JASON\_Encoding used to transfer data between server and client side application

\* Create new database if it doesn't exists

\*/

function geOrders()

{

$database = new DB();

$bind = null;

$sql = "SELECT \* FROM Order";

return json\_encode($database->query($sql));

// add (or update) the record to the database

$rows = $DB -> query ($query -> $binds);

}

/\*\*

\* @ paramater: $ID Return single product by given ID using SQL query

\* Get Product Function creates new database if it already does not exist on the server.

\* JASON\_Encoding used to transfer data between server and client side application

\*/

function getProduct($ID){

$database = new DB();

$bind = null;

$sql = "SELECT \* FROM Order WHERE OrderID = $ID";

return json\_encode($database->query($sql));

// add (or update) the record to the database

$rows = $DB -> query ($query -> $binds);

}

/\*\*

\* Delete Order Basket Function

\* Create new database if it doesn't exists

\* @orderID: delete the product using orderID. Order will be deleted by

\* specifying orderID

\*/

function deleteOrder($orderID) {

$database = new DB();

$sql = "DELETE FROM Orders WHERE OrderID= $OrderID";

$database->query($sql);

}

**DBConnect(connect.php)**

const HOST = 'localhost';

const USER = 'root';

const PASS = '';

const DBNAME= 'CaptureCart';

**db.php**

<?php

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

include ('connect.php');

/\*\*

\* Database Class

\* include statement for specified file in directory

\* local variable in constructor "$pdo"

\*/

class DB

{

private $pdo;

/\*\*

\* These functions has been taken from "Linbeta"

\* https://github.com/portsoc/linbeta/blob/master/inc/db.php

\* Some function might be modified accordingly.

\*/

/\*\*

\* The constructor function is used to connect to a MySQl engine

\* using HOST, Name, User and Passowrd.

\* from connect.php file.

\*/

public function \_\_construct()

{

// This will connect the database to the server

$dsn = "mysql:" . HOST . ";dbname=".DBNAME.";";

$option = array(

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION,

PDO::ATTR\_PERSISTENT => true

);

// Give error if the database doesn't exist or fail to load

try {

$this->pdo = new PDO($dsn, USER, PASS, $option);

$this->pdo->query("use ".DBNAME);

} catch (PDOException $failure) {

DB::throwException("Connect failed during construct");

}

}

public function query($query, $bindings = null)

{

if (isset($bindings)) {

$result =$this->pdo->prepare($query);

$result->execute($bindings);

} else {

$result =$this->pdo->query($query);

}

// Fetches All result from row with PDOStatement

if (strpos($query, 'SELECT') !== false) {

return $result->fetchAll(PDO::FETCH\_ASSOC);

}

// return result if exists and rowCount;

return $result->rowCount();

}

/\*\*

\* return PDO

\*/

public function pdo()

{

return $this->pdo;

}

}

**Install.php**

<!DOCTYPE html>

<html>

<head>

<title>Installation of Capture Cart</title>

<link rel="stylesheet" type="text/css" href="../../public/css/installDB.css">

</head>

<body>

<img class="images"src="../../public/images/webImages/captureCart.png" height="100" width="150" >

<section id="installationForm">

<?php

if (file\_exists('install.lock')) {

die('CaptureCart has already been installed.');

}

$step = isset($\_GET['step']) ? $\_GET['step'] : 1;

switch ($step) {

case '1':

?>

<form method="post" action="install.php?step=2">

<section>

<span>Host:</span><span class="required">\*</span> <input type="text" name="host" required="required" />

</section>

<section>

<span>Username:</span> <input type="text" name="username"/>

</section>

<section>

<span>Password:</span> <input type="text" name="password"/>

</section>

<div class="clearFloat"></div>

<section>

<input type="submit" name="submit" value="Next"/>

</section>

</form>

<?php

break;

case '2':

if (

!isset(

$\_POST['host'],

$\_POST['username'],

$\_POST['password'])

) {

die('Incomplete DB');

}

if (empty($\_POST['host'])) {

die('The DataBase host cannot be left empty.');

}

try {

$pdo = new PDO("mysql:host={$\_POST['host']}",$\_POST['username'],$\_POST['password'],[PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION]);

file\_put\_contents(

'./DBConnect/dbconfig.json',

json\_encode(

[

'host' => $\_POST['host'],

'dbname' => 'CaptureCart',

'username' => $\_POST['username'],

'password' => $\_POST['password']

]

)

);

$pdo->exec('CREATE DATABASE IF NOT EXISTS CaptureCart');

$pdo->exec('

CREATE TABLE IF NOT EXISTS CaptureCart.`Category` (

`CategoryID` int(11) NOT NULL AUTO\_INCREMENT PRIMARY KEY,

`CategoryName` varchar(100) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

');

$pdo->exec('

CREATE TABLE IF NOT EXISTS CaptureCart.`Contact` (

`ContactID` int(11) NOT NULL AUTO\_INCREMENT PRIMARY KEY,

`Name` varchar(100) NOT NULL,

`Email` varchar(50) NOT NULL,

`ContactNo` int(20) NOT NULL,

`Query` text NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

');

$pdo->exec('

CREATE TABLE IF NOT EXISTS CaptureCart.`Products` (

`ProductID` int(11) NOT NULL AUTO\_INCREMENT PRIMARY KEY,

`CategoryID` int(10) NOT NULL,

`ProductName` varchar(45) NOT NULL,

`ProductDescription` text,

`ProductPrice` decimal(11,2) NOT NULL,

`ProductStockLevel` int(11) NOT NULL,

`ProductImage` varchar(1000) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

');

$pdo->exec('

CREATE TABLE IF NOT EXISTS CaptureCart.`Order` (

`OrderID` int(11) NOT NULL AUTO\_INCREMENT PRIMARY KEY,

`Quantity` int(11) NOT NULL,

`Products\_ProductID` int(11) NOT NULL,

FOREIGN KEY (`Products\_ProductID`) REFERENCES CaptureCart.`Products` (`ProductID`) ON DELETE NO ACTION ON UPDATE NO ACTION

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

');

$pdo->exec("

INSERT INTO CaptureCart.`Category` (`CategoryID`, `CategoryName`) VALUES (null, 'Electronics'), (2, 'Cars')

");

$pdo->exec("

INSERT INTO CaptureCart.`Contact` (`ContactID`, `Name`, `Email`, `ContactNo`, `Query`) VALUES

(null, 'aa', 'aa', 0, 'aa'),

(null, 'sdafasdf', 'adsfasdf', 112, '121');

");

$pdo->exec("

INSERT INTO CaptureCart.`Products` (`ProductID`, `CategoryID`, `ProductName`, `ProductDescription`, `ProductPrice`, `ProductStockLevel`, `ProductImage`) VALUES

(null, 1, 'iPhone', 'Book', '33.00', 2, 'books.jpg'),

(null, 2, 'BMW ', 'BMW Made by ME', '222.00', 3, 'pencil.jpg'),

(null, 2, 'sdfasdf', 'asdfadsfadsfasdfasdfsdf', '222.00', 2, 'pencil.jpg'),

(null, 2, 'pencil', 'adfasdfasdfasdfasdf', '2323.00', 2323, 'pencil.jpg'),

(null, 2, 'iPhone 6s', 'Made By Me', '232.00', 2, 'iphone6.png'),

(null, 2, 'alex', 'computer science', '2333.00', 2, 'pencil.jpg'),

");

echo 'Capture Cart has been successfully installed!';

file\_put\_contents('install.lock', '');

} catch(PDOException $e) {

echo 'Installation Failed:<br/><br/>';

echo $e->getMessage();

}

break;

default:

die('Invalid step.');

}

?>

</section>

</body>

</html>

**Public**

**CSS adminPage.css**

.form {

margin: 10px auto;

max-width: 400px;

padding: 20px 12px 10px 20px;

border: 1px solid black;

}

.form li {

padding: 0px;

display: block;

list-style: none;

margin: 10px 0 0 0;

}

select {

padding: 7px;

margin: 0px;

border: 1px solid;

}

.productImage {

width: 150px;

height: 150px

}

#products {

background: white;

margin: 0;

display: block;

width: 100%;

box-sizing: border-box;

border: 1px solid black;

padding: 1em;

position: relative;

border-radius: 0 .5em .5em .5em;

}

.cart {

margin: 50px auto;

width: 300px;

overflow: hidden;

color: white;

text-shadow: 0 1px rgba(0, 0, 0, 0.6);

background: #525252;

border: 1px solid #202020;

border-radius: 3px;

-webkit-box-shadow: 0 1px 5px rgba(0, 0, 0, 0.5);

box-shadow: 0 1px 5px rgba(0, 0, 0, 0.5);

}

.catDesign {

text-align: center;

position: absolute;

left:750px;

top: 57px;

margin-left: -357px;

width: 250px;

margin: 200px;

border-style: solid;

}

.catDesign:hover {

background-color: silver;

}

.catDesignForAdmin {

text-align: center;

position: absolute;

left:1000px;

top: 157px;

margin: auto;

padding: 2em;

width: 10%;

}

.writeUp {

padding: 10px;

}

.layout {

background-color: white;

}

.adminCategory1 {

border-color: black;

width: 60px;

right: 1000px;

bottom: 100em;

}

**css1.css**

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

\* {

padding: 0em;

margin: 0em;

}

.clear {

clear: both;

}

header {

-webkit-border-radius:1em;

-moz-border-radius: 1em;

border-radius: 0px;

padding-top: 50px;

margin: auto;

width: 85%;

border-bottom: dotted;

}

h1 {

color: black;

font-size: 50px;

margin: auto;

text-align: center;

}

h2 {

color: black;

margin: auto;

text-align: center;

font-size: 30px;

}

h3 {

color: black;

margin: auto;

text-align: left;

font-size: 25px;

}

h4 {

color: black;

font-size: 20px;

margin: auto;

}

a {

color: green;

}

p {

color: blue;

margin: auto;

text-align: left;

font-size: 10px;

margin: 0.5em 0em 0.5em 0em;

}

body {

width: 100%;

padding: 0em;

height: auto;

}

article {

background-image: url('../images/webImages/background2.png');

margin: auto;

padding: 2em;

width: 85%;

margin-top: 60px;

}

section{

margin: 1em;

padding: 2em;

width: 70%;

border-style: solid;

border-width: medium;

height: auto;

background-color: white;

right: 100px;

}

/\* Nav Starst \*/

.nav {

display: inline-block;

text-align: center;

position: absolute;

left:53%;

right: 50%;

margin-left: -357px;

width: 600px;

}

.nav li {

float: left;

list-style-type: none;

position: relative;

}

.nav li a {

font-size: 16px;

line-height: 60px;

padding: 0 26px;

text-decoration: none;

font-family: Montserrat, sans-serif;

}

.nav li a:hover {

background-color: silver;

}

/\*.admin {

position: absolute;

bottom: 5px;

right: 10px;

height: 52px;

}

\*/

#dropDown a{

border-left: 0 none;

position: fixed;

left: 1100px;

top: -5px;

border-style: solid;

border-color: grey;

}

#intranet{

position: fixed;

left: 1100px;

top: -5px;

border-style: solid;

border-color: grey;

}

#dropDown > a {

background-position: 85% center;

background-repeat: no-repeat;

padding-right: 42px;

}

#dropDown:hover {

visibility: visible;

top: 100%;

opacity: 1;

}

/\*Home Logo Image\*/

[class\*="icon-"] {

display: inline-block;

width: 16px;

height: 16px;

vertical-align: text-top;

background-image: url("../images/webImages/home.png");

background-repeat: no-repeat;

background-color: transparent;

font: 0/0 serif;

text-shadow: none;

color: transparent;

}

/\* Nav End \*/

/\* Search Bar Start\*/

#search {

width: 357px;

margin: 4px;

display: inline-block;

}

#search\_text{

font-size: 16px;

border: 0 none;

margin-right: 0;

padding: 15px 0 15px 20px;

background: #00FFFF;

border: black;

float: left;

box-shadow: 0 0 5px rgba(81, 203, 238, 1);

width: 250px;

position: absolute;

top: 5px;

right: 10px;

height: 52px;

box-sizing: border-box;

}

#search\_button {

border: 0 none;

background: #00FFFF;

width: 60px;

float: left;

padding: 0;

border: black;

text-align: center;

height: 52px;\*/

cursor: pointer;

box-shadow: 0 0 5px rgba(81, 203, 238, 1);

position: absolute;

top: 5px;

right: 10px;

}

/\* Search Bar End\*/

img {

clear: both;

}

.images {

position: absolute;

top: 5px;

left: 10px;

}

footer {

border-style: solid;

margin: auto;

padding: 2em;

width: 85%;

margin-top: 20px;

bottom: 5px;

}

**installDB.css**

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

\* {

margin: 0;

padding: 0;

}

body {

width: 100%;

height: 100%;

}

#installationForm {

position: fixed;

top: 25%;

left: 50%;

margin-left: -11em;

width: 20em;

height: 8.5em;

padding: 1em;

}

#installationForm section {

margin-bottom: 1em;

float: right;

}

#installationForm span, #form input {

display: inline-block;

font-size: 1em;

}

.installationForm {

color: #b00;

}

.goback {

margin-top: 2em;

}

.clearFloat {

clear: both;

}

**HTML Folder**

**Index.html**

<!DOCTYPE html>

<html>

<!--

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

-->

<head>

<meta charset="utf-8">

<title>Capture Cart</title>

<link rel="stylesheet" type="text/css" href="public/css/css1.css">

<script type='text/javascript' src='public/js/otherJS/searchBar.js'></script>

</head>

<body>

<header>

<h1> You Order We Deliver</h1>

<a href="/687776/index.html">

<img class="images"src="public/images/webImages/captureCart.png" height="100" width="150" >

</a>

<ul class="nav">

<li>

<a href="/687776/index.html" class="nav-icon" title="Home"><span class="icon-home">Home</span></a>

</li>

<li>

<a href="/687776/public/html/products" title="Products">Products</a>

</li>

<li>

<a href="/687776/public/html/contact" title="Contact Us">Contact Us</a>

</li>

<li>

<a href="/687776/public/html/Admin/addProducts" title="Admin">Admin</a>

</li>

</ul>

<form>

<id="script" type="text/javascript" src="searchBar.js">

<input name="search\_text" id="search\_text" placeholder="Coming Soon"/>

</form>

</header>

<article>

<h2> Online Store</h2>

<h3>Please go to products Page</h3>

<p>More Coming Soon</p>

</article>

<footer>

<p>UP687776, University of Portsmouth, 2014 / 2015 All Right Reserved</p>

<p>Contact Us: 0123456789</p>

<p>Email Address: example@supoort.com</p>

</footer>

</html>

**products.html**

<!DOCTYPE html>

<html>

<!--

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

-->

<head>

<meta charset="utf-8">

<title>Products</title>

<link rel="stylesheet" type="text/css" href="../css/css1.css">

<link rel="stylesheet" type="text/css" href="../css/adminPage.css">

<script type='text/javascript' src='../js/products/clientProducts.js'></script>

<script type='text/javascript' src='../js/cart/localStorageCart.js'></script>

<script type='text/javascript' src='../js/otherJS/searchBar.js'></script>

</head>

<body>

<header>

<h1> Your Products</h1>

<a href="/687776/index.html">

<img class="images" src="../images/webImages/captureCart.png" height="100" width="150" >

</a>

<ul class="nav">

<li>

<a href="/687776/index.html" class="nav-icon" title="Home"><span class="icon-home">Home</span></a>

</li>

<li>

<a href="/687776/public/html/products" title="Products">Products</a>

</li>

<li>

<a href="/687776/public/html/contact" title="Contact Us">Contact Us</a>

</li>

<li>

<a href="/687776/public/html/Admin/addProducts" title="Admin">Admin</a>

</li>

</ul>

<form action="javascript:void(0);">

<input name="search\_text" id="search\_text" placeholder="Search" onkeydown="search()">

</form>

</header>

<article id="productsView">

<h2>Products Page</h2>

<p><div class="catDesign" id="cats"><h3>Select By Category</h3></div></p>

<div id ="viewproductByCategory"> </div>

<section id="basket" class="cart">

<h3>Your Basket</h3>

</section>

<section id="check">

<button id="checkOut" type="button">Checkout</button>

</section>

</article>

<footer>

<p>UP687776, University of Portsmouth, 2014 / 2015 All Right Reserved</p>

<p>Contact Us: 0123456789</p>

<p>Email Address: example@supoort.com</p>

</footer>

</body>

</html>

**Admin Folder (addProducts.html)**

<!DOCTYPE html>

<html>

<!--

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

-->

<head>

<meta charset="utf-8">

<title>Capture Cart Form Admin</title>

<link rel="stylesheet" type="text/css" href="../../css/css1.css">

<link rel="stylesheet" type="text/css" href="../../css/adminPage.css">

<script type='text/javascript' src='../../js/products/postCat.js'></script>

<script type='text/javascript' src='../../js/products/postProducts.js'></script>

<script type='text/javascript' src='../../js/products/viewCategory.js'></script>

</head>

</head>

<body>

<header>

<h1>Admin Add Products</h1>

<h2>Add Products </h2>

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

<img class="images" src="../../images/webImages/captureCart.png" height="100" width="150" >

</a>

<ul class="nav">

<li>

<a href="addProducts" title="Products">Add Products</a>

</li>

<li>

<a href="adminProducts" title="Products">Products</a>

</li>

<li>

<a href="#" title="Report">Report</a>

</li>

<li>

<a href="../../../index.html" title="Home">Public</a>

</li>

</ul>

<form>

<input name="search\_text" id="search\_text" placeholder="Coming Soon"/>

</form>

</header>

<article>

<h2> Category</h2>

<form id='f1'>

<ul class='form'>

<li>Category Name:<span class="required"></span><input type="text" id="categoryName" value=""/> </li>

<li><span class="required"></span><input id="submitCat" type= "button" value="submit"></li>

</ul>

</form>

<p id="adminCategory"> </p>

</article>

<article>

<h2> Products</h2>

<form id='f1' enctype="multipart/form-data">

<ul class='form'>

<li>Product Name:<span class="required"></span><input type="text" id="productName" value=""/> </li>

<li>Category ID:<span class="required"></span><input type="text" id="categoryID" value=""/>

Please Select Category ID from top of the page</li>

<li>Product Description:<textarea type="text" id="productDescripton" value="" class="field-ling"/></textarea></li>

<li>Product Price:<span class="required"></span> <input type="text" id="productPrice" value=""/></li>

<li>Product Stock:<span class="required"></span> <input type="text" id="productStock" value=""/></li>

<li>Product Image:<span class="required"></span><input type="file" id="productImage" value="" name="images" required/>

</li>

<li><span class="required"></span><input id="submit" type= "button" value="submit"></li>

</ul>

</form>

</article>

<footer>

<p>UP687776, University of Portsmouth, 2014 / 2015 All Right Reserved</p>

<p>Contact Us: 0123456789</p>

<p>Email Address: example@supoort.com</p>

</footer>

</body>

</html>

**adminProducts.html**

<!DOCTYPE html>

<html>

<!--

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

-->

<head>

<meta charset="utf-8">

<title>Products</title>

<link rel="stylesheet" type="text/css" href="../../css/css1.css">

<link rel="stylesheet" type="text/css" href="../../css/adminPage.css">

<script type='text/javascript' src='../../js/products/adminProducts.js'></script>

<script type='text/javascript' src='../../js/otherJS/searchBar.js'></script>

</head>

<body>

<header>

<h1>Admin Add Products</h1>

<h2> Products </h2>

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

<img class="images" src="../../images/webImages/captureCart.png" height="100" width="150" >

</a>

<ul class="nav">

<li>

<a href="addProducts" title="Products">Add Products</a>

</li>

<li>

<a href="adminProducts" title="Products">Products</a>

</li>

<li>

<a href="#" title="Report">Report</a>

</li>

<li>

<a href="../../../index.html" title="Home">Public</a>

</li>

</ul>

<form action="javascript:void(0);">

<input name="search\_text" id="search\_text" placeholder="Search" onkeydown="search()">

</form>

</header>

<article id="article" class="marginTop">

<h2>Products Page</h2>

<p><div class="catDesign" id="cats"><h3>Select Category</h3></div></p>

<div id ="viewproductByCategory"> </div>

<h4>Refresh Page everytime you delete product</h4>

</article>

<footer>

<p>UP687776, University of Portsmouth, 2014 / 2015 All Right Reserved</p>

<p>Contact Us: 0123456789</p>

<p>Email Address: example@supoort.com</p>

</footer>

</body>

</html>

**JavaScript Folder**

**Localstorage.js**

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

var shoppingCartItems;

var basket = [];

var productToStore;

/\*\*

\* @parameter (e (event to occur))

\* add to cart function, push the element by id

\* then add it to cart

\* add the shopping cart to local storage (recommend)

\* CustomEvents used to create new events everytime addToCart is pressed using EventName(productID)

\*/

function addCart(e) {

basket.push(e.toElement.id);

productToStore = JSON.stringify(basket);

localStorage['basket'] = productToStore;

document.dispatchEvent(new CustomEvent('cartChanged'));

}

/\*\*

\* @parameter: productId used to find products

\* Uses forloop to find products

\* if the products exits (which customer ask for) then return the product to cart

\*/

function findProduct(productId){

for (var i = 0; i < findProducts.length; i++) {

if (findProducts[i].ProductID === productId){

return(findProducts[i]);

}

}

}

document.addEventListener('cartChanged', displayCart);

/\*\*

\* Function is used to display cart on screen and add it to localstorage in console.

\* Return the

\*/

function displayCart() {

console.log('@displayCart : localStorageCart');

productToStore = JSON.parse(localStorage['basket']);

var basketElement = document.getElementById("basket");

basketElement.innerHTML = ''; // inside innerHTML and empty string (it removes the previous contents of the list)

console.log(productToStore);

for (var i = 0; i < productToStore.length; i++) {

var currentProduct = findProduct(productToStore[i]);

basketElement.innerHTML += "<p> ID: "+currentProduct.ProductID+"</p><h4>Name: "+currentProduct.ProductName+"</h4><p>Price: "+currentProduct.ProductPrice+"</p><button class='delete' id=" + findProducts[i].ProductID+">Delete</button>";

}

delbtnsCart = document.getElementsByClassName('delete');

for(a = 0; a < delbtnsCart.length; a++)

{

/\*\*

\* Delete the Product by specifying ProductID by using eventListener on click

\* grabs delete button from response function and delete the product

\* Delete product from database by sending the request to the web server using URL on xhr.open method

\*/

(function() {

var id = delbtnsCart[a].id;

delbtnsCart[a].addEventListener('click',

function(){

xhra = new XMLHttpRequest();

xhra.open('DELETE', '/687776/api/2/ orders/'+ id);

xhra.send();});

}());

}

}

window.addEventListener('load',

/\*\*

\* Once the customer have added the products to cart, and

\* when they click checkout button

\* it will post the order to database

\*/

function() {

var checkOut = document.getElementById('checkOut');

checkOut.addEventListener('click', function(e) {

xhr = new XMLHttpRequest();

xhr.open('POST', '/687776/api/2/orders/index.php', true);

xhr.send(localStorage['basket']);

xhr.onload = function () {

console.log(xhr.response);

}

});

});

**others folder (searchBar.js)**

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

/\*\*

\* Search Function

\* window.setTimeout is so it can wait for 1 milie-second after userinput to allow all of the words to be entered

\* grabs the id from products html page (search\_text)

\* grabs the ajax Get request from server side search.php to allow the user to search for the products.

\* grabs the id and place into innerHTMl and give the response to user input

\*/

function search() {

window.setTimeout(

function() {

var search = document.getElementById("search\_text").value;

AjaxGet("http://localhost/687776/api/2/search.php?input="+search, function(response)

{

document.getElementById("viewproductByCategory").innerHTML = response;

});

}, 1);

}

/\*\*

\* This code has been taken from Kit Lester Worksheet. However, it has been modified accordingly

\* Ajax get function send the httpRequestion and uses the URL to callback the function.

\*

\*/

function AjaxGet(URL, callback) {

var ajaxObject = new XMLHttpRequest();

ajaxObject.open("GET", URL, true); //True is asynchronous

ajaxObject.onreadystatechange = function()

{ if (ajaxObject.status == 200)

if (ajaxObject.readyState == 4)

callback(ajaxObject.responseText);

};

ajaxObject.send(null);

}

**products folder (adminProducts.js)**

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

var xhr = checkObject();

var findProducts;

/\*\*

\* Function checkObject checks if the browser is compatible

\* It checks what XML to run depending on browser type

\* All Modern browsers support the XMLHttpRequest object

\* However, IE5 and IE6 support ActiveXObject.

\*/

function checkObject()

{

var xhr;

if(window.XMLHttpRequest)

{

xhr = new XMLHttpRequest();

}

else

{

xhr = new ActiveXObject("MICROSOFT.XMLHTTP");

}

return xhr;

}

/\*\*

\* Function grabProducts will interact with GET Method on products.php

\* xhr.open specifies the type of the request and the URL for products.php

\* xhr.onreadystatechange storea a function/gets a response and get called automatically each time

\* xhr.send send the request to the server, however, we are not sending anything,

\* just getting products using URL

\* log the call to console and check the event

\* Allow Admin to view products on their browser and Allow them to add, edit, create new products and update and delete products

\*/

function grabProducts()

{

if(xhr != null)

{

try

{

xhr.open("GET", "../../../api/2/products", false);

xhr.onreadystatechange = getResponse;

xhr.send(null);

}

catch(e)

{

console.log(e.toString());

}

}

}

/\*\*

\* Function delete Products is use to send the request and delete product

\* @ parameter: ProductID: sends the request to the server by specifying product ID

\* xhr.onreadystatechange storea a function/gets a response and get called automatically each time

\*/

function deleteProducts(ProductID)

{

if(xhr != null)

{

try

{

xhr.open("DELETE", "../../api/2/products/"+ProductID, false);

xhr.onreadystatechange = getResponse;

xhr.send(null);

}

catch(e)

{

console.log(e.toString());

}

}

}

/\*\*

\* Function getResponse is used to check the status and go back to grabProducts from web server and database

\* check 200 status and if okay then proceede with request

\* JSON.parese(responseText) is to read data from a web server and read using XMLHTTP

\* for loop is used to get response of product for getting and deleteing the product using ProductID

\*/

function getResponse()

{

if (xhr.status == 200 && xhr.statusText == 'OK')

{

console.log("request verified");

findProducts = JSON.parse(xhr.responseText);

console.log(findProducts);

for(var i = 0; i < findProducts.length; ++i)

{

console.log(findProducts[i].ProductID);

document.getElementById("article").innerHTML += "<section class='layout'><p>ProductID: "+findProducts[i].ProductID+"</p><h4> Product: "+findProducts[i].ProductName+"</h4><p class='writeUp'> Description: "+findProducts[i].ProductDescription+"</p><p class='writeUp'>£"+findProducts[i].ProductPrice+"</p>"+"<p class='writeUp'>Stock Level: "+findProducts[i].ProductStockLevel+"</p><img class='productImage' src=../../images/productImages/"+findProducts[i].ProductImage+"><button class='delete' id=" + findProducts[i].ProductID+">Delete</button>"+"<button class='edit' id=" + findProducts[i].ProductID+"> Edit </button></section>";

}

}

delbtns = document.getElementsByClassName('delete');

for(a = 0; a < delbtns.length; a++)

{

/\*\*

\* Delete the Product by specifying ProductID by using eventListener on click

\* grabs delete button from response function and delete the product

\* Delete product from database by sending the request to the web server using URL on xhr.open method

\*/

(function() {

var id = delbtns[a].id;

delbtns[a].addEventListener('click',

function(){xhra = new XMLHttpRequest();xhra.open('DELETE', '/687776/api/2/products/'+ id);xhra.send();});

}());

}

editButton = document.getElementsByClassName('edit');

for(a = 0; a < editButton.length; a++)

{

/\*\*

\* Edit/Update the Product by specifying ProductID by using eventListener on click

\* window.locaiton will change the location of file when user click on Edit button for certain product.

\*

\*/

(function() {

var id = editButton[a].id;

editButton[a].addEventListener('click',

function() {

window.location = '/687776/public/html/Admin/editProducts?'+id;

})

}());

}

}

function grabProductCategories()

{

var xhr = new XMLHttpRequest();

xhr.open('GET', 'http://localhost/687776/api/2/categories/category', true);

xhr.send();

xhr.onload = function() {

var categ = JSON.parse(xhr.response);

var cats = document.getElementById("cats");

categ.forEach(

function(e) {

var catElement = document.createElement('div');

catElement.innerHTML = e.CategoryName;

cats.appendChild(catElement);

catElement.addEventListener('click',

function(e) {

var xhr = new XMLHttpRequest();

xhr.open("GET", "http://localhost/687776/api/2/products?category="+e.target.innerHTML, false);

xhr.send(null);

document.getElementById("viewproductByCategory").innerHTML = '';

findProducts = JSON.parse(xhr.responseText);

for(var i = 0; i < findProducts.length; ++i)

{

document.getElementById("viewproductByCategory").innerHTML += "<section class='layout'><p>ProductID: "+findProducts[i].ProductID+"</p><h4> "+findProducts[i].ProductName+"</h4><p class='writeUp'>"+findProducts[i].ProductDescription+"</p><p class='writeUp'>£"+findProducts[i].ProductPrice+"</p>"+"<p class='writeUp'>Stock Level:"+findProducts[i].ProductStockLevel+"</p><img class='productImage' src=../images/productImages/"+findProducts[i].ProductImage+"><button id=" + findProducts[i].ProductID+"> Add To Cart </button>";

}

})

})

}

}

//attach and load event and go back to get Produts from grabProducts function

window.addEventListener("load", grabProducts);

window.addEventListener("load", grabProductCategories);

**clientProduct.js**

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

var xhr = checkObject();

var findProducts;

/\*\*

\* Function checkObject checks if the browser is compatible

\* It checks what XML to run depending on browser type

\* All Modern browsers support the XMLHttpRequest object

\* However, IE5 and IE6 support ActiveXObject.

\*/

function checkObject()

{

var xhr;

if(window.XMLHttpRequest)

{

xhr = new XMLHttpRequest();

}

else

{

xhr = new ActiveXObject("MICROSOFT.XMLHTTP");

}

return xhr;

}

/\*\*

\* Function grabProducts will interact with GET Method on products.php

\* xhr.open specifies the type of the request and the URL for products.php

\* xhr.onreadystatechange storea a function/gets a response and get called automatically each time

\* xhr.send send the request to the server, however, we are not sending anything,

\* just getting products using URL

\* log the call to console and check the event

\*/

function grabProducts()

{

if(xhr != null)

{

try

{

xhr.open("GET", "../../api/2/products", false);

xhr.onreadystatechange = getResponse;

xhr.send(null); // the post form will go here

}

catch(e)

{

console.log(e.toString());

}

}

}

/\*\*

\* Function grabProductCategory will interact with GET Method on products.php

\* xhr.open specifies the type of the request and the URL for products.php

\* xhr.onreadystatechange storea a function/gets a response and get called automatically each time

\* xhr.send send the request to the server, however, we are not sending anything,

\* just getting products using URL

\* log the call to console and check the event

\* Grabs the product by Category

\*/

function grabProductCategories()

{

var xhr = new XMLHttpRequest();

xhr.open('GET', 'http://localhost/687776/api/2/categories/category', true);

xhr.send();

xhr.onload = function() {

var categ = JSON.parse(xhr.response);

var cats = document.getElementById("cats");

categ.forEach(

function(e) {

var catElement = document.createElement('div');

catElement.innerHTML = e.CategoryName;

cats.appendChild(catElement);

catElement.addEventListener('click',

function(e) {

var xhr = new XMLHttpRequest();

xhr.open("GET", "../../api/2/products?category="+e.target.innerHTML, false);

xhr.send(null);

document.getElementById("viewproductByCategory").innerHTML = '';

findProducts = JSON.parse(xhr.responseText);

for(var i = 0; i < findProducts.length; ++i)

{

document.getElementById("viewproductByCategory").innerHTML += "<section class='layout'><p>ProductID: "+findProducts[i].ProductID+"</p><h4> "+findProducts[i].ProductName+"</h4><p class='writeUp'>"+findProducts[i].ProductDescription+"</p><p class='writeUp'>£"+findProducts[i].ProductPrice+"</p>"+"<p class='writeUp'>Stock Level:"+findProducts[i].ProductStockLevel+"</p><img class='productImage' src=../images/productImages/"+findProducts[i].ProductImage+"><button id=" + findProducts[i].ProductID+"> Add To Cart </button>";

}

})

})

}

}

/\*\*

\* Get Response checks the status.

\* The json parse method will pass the parse string as JSON

\* It check if the product exists in grabProducts and grabProductsCategory

\* then creates the products using innerHTML on web page

\* There is also a add to cart button which will add the product to the shopping cart on product Page

\* Function getResponse is used to check the status and go back to grabProducts from web server and database

\* check 200 status and if okay then proceede with request

\* JSON.parese(responseText) is to read data from a web server and read using XMLHTTP

\* for loop is used to get response of product for getting and deleteing the product using ProductID

\*/

function getResponse()

{

if (xhr.status == 200 && xhr.statusText == 'OK')

{

console.log("request verified");

findProducts = JSON.parse(xhr.responseText);

console.log(findProducts);

console.log(JSON.stringify(findProducts));

for(var i = 0; i < findProducts.length; ++i)

{

document.getElementById("viewproductByCategory").innerHTML += "<section class='layout'><p>ProductID: "+findProducts[i].ProductID+"</p><h4> Product "+findProducts[i].ProductName+"</h4><p class='writeUp'> Description: "+findProducts[i].ProductDescription+"</p><p class='writeUp'>£"+findProducts[i].ProductPrice+"</p>"+"<p class='writeUp'>Stock Level: "+findProducts[i].ProductStockLevel+"</p><img class='productImage' src=../images/productImages/"+findProducts[i].ProductImage+"><button id=" + findProducts[i].ProductID+"> Add To Cart </button>";

}

var buttons = document.getElementsByTagName('button');

for (var i = 0; i < buttons.length; i++) { // for all buttons

if (buttons[i].id !== 'search\_button' && buttons[i].id !== 'checkOut') { // except search

buttons[i].addEventListener('click', addCart); // add click eventHandler calling addCart

}

}

}

}

//check the status and go back to grabProduct Function

window.addEventListener("load", grabProducts);

window.addEventListener("load", grabProductCategories);

**editProducts.js**

function editProducts() {

var productName = document.getElementById('productName').value;

var productPrice = document.getElementById('productPrice').value;

var ProductID = location.search.split(/\?/)[1];

var data = {'ProductName': productName, 'ProductPrice': productPrice};

xhr = new XMLHttpRequest();

xhr.open("PATCH", "../../../api/2/products/"+ProductID, false);

xhr.send(JSON.stringify(data));

}

/\*\*

\* This method is used to load the submit function

\* Once the user have clickec on Submit(ID) button, it will

\* response to it and change the information accordingly

\*/

window.addEventListener("load", function(){

submit = document.getElementById('submit');

submit.addEventListener("click", editProducts);

});

**postCat.js**

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

/\*\*

\* Post Products Function

\* The submit ID is exported from admin folder to allow the admin to create new products

\* Once user click submit it sends the register and sends the request to specified URL

\* If successful then it post and create the new products onto the database

\* If the request fails the returns the error and status onto the console

\* The callback function get excuted

\*/

function postCategory() {

var submit = document.getElementById('submit');

function successXHR () { this.callback.apply(this, this.arguments); }

function errorXHR () { console.error(this.statusText); }

function productsFromAPI (method, url, Catdata, callback) {

console.log(Catdata);

var register = new XMLHttpRequest();

register.callback = callback;

register.arguments = Array.prototype.slice.call(arguments, 3);

register.onload = successXHR;

register.onerror = errorXHR;

register.open(method, "/687776"+url, true);

register.send(Catdata);

}

// put on the page and it does all the ajax request for all of the pages

submitCat.addEventListener('click', function(event) {

console.log(event);

event.preventDefault();

var Catdata = {};

Catdata["categoryName"] = document.getElementById('categoryName').value;

console.log(Catdata);

productsFromAPI("POST", "/api/2/categories/category", JSON.stringify(Catdata), function(){

console.log(this.responseText);

});

});

}

window.addEventListener("load", postCategory);

**postProducts.js**

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

/\*\*

\* Post Products Function

\* The submit ID is exported from admin folder to allow the admin to create new products

\* Once user click submit it sends the register and sends the request to specified URL

\* If successful then it post and create the new products onto the database

\* If the request fails the returns the error and status onto the console

\* The callback function get excuted

\*/

function postProducts() {

var submit = document.getElementById('submit');

function successXHR () { this.callback.apply(this, this.arguments); }

function errorXHR () { console.error(this.statusText); }

function productsFromAPI (method, url, Productsdata, callback) {

console.log(Productsdata);

var register = new XMLHttpRequest();

register.callback = callback;

register.arguments = Array.prototype.slice.call(arguments, 3);

register.onload = successXHR;

register.onerror = errorXHR;

register.open(method, "/687776"+url, true);

register.send(Productsdata);

}

// put on the page and it does all the ajax request for all of the pages

submit.addEventListener('click', function(event) {

console.log(event);

event.preventDefault();

var Productsdata = {};

Productsdata["productName"] = document.getElementById('productName').value;

Productsdata["categoryID"] = document.getElementById('categoryID').value;

Productsdata["productDescripton"] = document.getElementById('productDescripton').value;

Productsdata["productPrice"] = document.getElementById('productPrice').value;

Productsdata["productStock"] = document.getElementById('productStock').value;

Productsdata["productImage"] = document.getElementById('productImage').files[0].name; //value;

productsFromAPI("POST", "/api/2/products", JSON.stringify(Productsdata), function(){

console.log(this.responseText);

});

});

}

//check the status and go back to post the products

//attach and load event and go back to get Produts from postProducts function

window.addEventListener("load", postProducts);

**viewCategory.js**

/\*\*

\* Author: UP687776

\* Web Script Programming

\* Rick Bookes && Kit Lester

\* 2014 && 2015

\*/

var xhr = checkObject();

var findCategories;

/\*\*

\* Function checkObject checks if the browser is compatible

\* It checks what XML to run depending on browser type

\* All Modern browsers support the XMLHttpRequest object

\* However, IE5 and IE6 support ActiveXObject.

\*/

function checkObject()

{

var xhr;

if(window.XMLHttpRequest)

{

xhr = new XMLHttpRequest();

}

else

{

xhr = new ActiveXObject("MICROSOFT.XMLHTTP");

}

return xhr;

}

/\*\*

\* Function grabProducts will interact with GET Method on products.php

\* xhr.open specifies the type of the request and the URL for products.php

\* xhr.onreadystatechange storea a function/gets a response and get called automatically each time

\* xhr.send send the request to the server, however, we are not sending anything,

\* just getting products using URL

\* log the call to console and check the event

\* Allow Admin to view products on their browser and Allow them to add, edit, create new products and update and delete products

\*/

function grabCategories()

{

if(xhr != null)

{

try

{

xhr.open("GET", "../../../api/2/categories/category", false);

xhr.onreadystatechange = getResponse;

xhr.send(null);

}

catch(e)

{

console.log(e.toString());

}

}

}

/\*\*

\* Function delete Products is use to send the request and delete product

\* @ parameter: ProductID: sends the request to the server by specifying product ID

\* xhr.onreadystatechange storea a function/gets a response and get called automatically each time

\*/

function deleteProducts(CategoryID)

{

if(xhr != null)

{

try

{

xhr.open("DELETE", "../../api/2/categories/category/"+CategoryID, false);

xhr.onreadystatechange = getResponse;

xhr.send(null);

}

catch(e)

{

console.log(e.toString());

}

}

}

/\*\*

\* Function getResponse is used to check the status and go back to grabCategories from web server and database

\* check 200 status and if okay then proceede with request

\* JASON.parese(responseText) is to read data from a web server and read using XMLHTTP

\* for loop is used to get response of product for getting and deleteing the product using CategoryID

\*/

function getResponse()

{

if (xhr.status == 200 && xhr.statusText == 'OK')

{

console.log("request verified");

findCategories = JSON.parse(xhr.responseText);

console.log(findCategories);

for(var i = 0; i < findCategories.length; ++i)

{

console.log(findCategories[i].CategoryID);

document.getElementById("adminCategory").innerHTML += "<section class='adminCategory1'><p>CategoryID: "+findCategories[i].CategoryID+"</p><p> Name: "+findCategories[i].CategoryName+"</p><button class='delete' id=" + findCategories[i].CategoryID+">Delete</button></section>";

}

}

delbtns = document.getElementsByClassName('delete');

for(a = 0; a < delbtns.length; a++)

{

/\*\*

\* Delete the Category by specifying CategoryID by using eventListener on click

\* grabs delete button from response function and delete the product

\* Delete category from database by sending the request to the web server using URL on xhr.open method

\*/

(function() {

var id = delbtns[a].id;

delbtns[a].addEventListener('click',

function(){xhra = new XMLHttpRequest();xhra.open('DELETE', '/687776/api/2/categories/category/'+ id);xhra.send();});

}());

}

}

//attach and load event and go back to get Produts from grabProducts function

window.addEventListener("load", grabCategories);