allbooks.php:

```
<!--Andrew Robson - w16011147-->
<!--This file will be used to display all of the books to the user --> <!--
The user will need to be logged in if they wish to access the data --> <!
DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8"/>
   <meta name="viewport" content="width=device-width"/>
   <link rel="stylesheet" href="styling.css"> <!--Links the stylesheet-->
   <title>List All Books</title>
</head>
<body>
<?php
require_once ('functions.php'); //links to functions.php
ini_set("session.save_path", "/home/unn_w16011147/sessionData"); //initialises the session data
session_start(); //begins a session

$dbConn = getConnection(); //connects to the database

displayLoginLogout(); //calls function to display login functionality
displayNav(); //calls a function to display navigation

if (checkLogin()) //if the user is logged in then proceed
{
   //sql statement to retrieve the data from nbc_books
   $sqlSelect = "SELECT bookISBN, bookTitle, bookYear, pubID, catID, bookPrice
            FROM nbc_books
            ORDER BY bookTitle";

   $result = $dbConn->prepare($sqlSelect); //prepare the statement
   $result->execute(); //execute the statement

   while ($returnedBook = $result->fetchObject()) //while values are being returned echo out the following
   {
      //applies a link to the results, when clicked the user is redirected to the edit page
      echo "<div class='bookOption'>\n
      <span class='ISBN - '>
      <a href='editBook.php?bookISBN={$returnedBook->bookISBN}'>{$returnedBook->bookISBN}</span>\n
      <span class='Title - '>{$returnedBook->bookTitle}</span>\n
            <span class='Year - '>{$returnedBook->bookYear}</span>\n
            <span class='PublisherID - '>{$returnedBook->pubID}</span>\n
            <span class='CategoryID - '>{$returnedBook->catID}</span>\n
            <span class='Price - '>{$returnedBook->bookPrice}</span>\n
            </div>\n";
   }
```

```
}
else //if the user is not logged in then echo out the following
{
   echo "<p>You must be logged in to access this content.</p>\n";
}

?>
</body>
</html>
```

Credits:

```
<!--Andrew Robson - w16011147-->
<!--This file is used to display all of the references I have used in the creation of this project.-->
<!--The user will need to be logged in to access the data-->
<html lang="en">
<head>
   <meta charset="UTF-8"/>
   <meta name="viewport" content="width=device-width"/>
   <link rel="stylesheet" href="styling.css"> <!--Links the stylesheet-->
   <title>Credits</title>
</head>
<body>
<?php
require_once ('functions.php'); //link to functions.php
ini_set("session.save_path", "/home/unn_w16011147/sessionData"); //initialises the session data
session_start(); //begins a session

$dbConn = getConnection(); //connects to the database

displayLoginLogout(); //displays the login functionality
displayNav(); //displays the navigation functionality

if (checkLogin()) //if the user is logged in then display the following
{
   echo "<p>Andrew Robson - w16011147 </p>\n";
   echo "<p><em>References: </em> </p>\n";
   echo "<p>(Meloni, 2007) Meloni, J. (2007). PHP, MYSQL, and Apache. Indianapolis, Ind.: Sams.</p>\n";
   echo "<p>(Negrino and Smith, 2009) Negrino, T. and Smith, D. (2009). JavaScript and Ajax for the web.
      Berkeley: Peachpit Press.</p>\n";
   echo "<p>Unn-isrd1.newnumyspace.co.uk. (2017). The Wheel: Teaching.
      [online] Available at: http://unn-isrd1.newnumyspace.co.uk/learn
      [Accessed 13 Dec. 2017]. </p>\n";
   echo "<p>W3schools.com. (2017). W3Schools Online Web Tutorials.
      [online] Available at: https://www.w3schools.com/ [Accessed 15 Dec. 2017].</p>";
   echo "<p>Codecademy. (2017). JavaScript.
      [online] Available at: https://www.codecademy.com/en/tracks/javascript [Accessed 15 Dec. 2017].</p>";
}
```

```
else //if the user is not logged in then display the following
{
   echo "<p>You must be logged in to access this content.</p>\n";
}
?>

</body>
</html>
```

editBook.php:

```
<!--Andrew Robson - w16011147-->
<!--This file will display the edit form that the user be directed to from 'allBooks.php'-->
<!--The user will need to be logged in to access this page-->
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8"/>
   <meta name="viewport" content="width=device-width"/>
   <link rel="stylesheet" href="styling.css"> <!--Links the stylesheet-->
   <title>Edit Books</title>
</head>
<body>
<?php
require_once("functions.php"); //links the file to functions.php
$dbConn = getConnection(); //connects the file to the database
ini_set("session.save_path", "/home/unn_w16011147/sessionData"); //initialises the session data
session_start(); //begins a session
displayLoginLogout(); //displays the login functionality
displayNav(); //displays the navigation functionality

if(checkLogin()) //if the user is logged in then proceed
{
   $bookISBN = filter_has_var(INPUT_GET, 'bookISBN') ? $_GET['bookISBN'] : null;
   //retrieves the bookISBN number from the previous page
   $bookISBN = trim($bookISBN); //trims the bookISBN in case there is any spaces either side

   if (!$bookISBN) //if bookISBN is null then display the following
   {
      echo "<p>No ISBN number has been passed.</p>\n";
      echo "<p>To return to book list, click <a href='allBooks.php'>Here</a></p>"; //provides a link to
book list
      exit;
   }
```

```
try
   {
      $sqlSelect = "SELECT bookISBN, bookTitle, bookYear, pubID, catID, bookPrice
            FROM nbc_books
            WHERE bookISBN = $bookISBN"; //retrieve all the data relevant for a specific bookISBN

      $queryResult  = $dbConn->query($sqlSelect);
      $returnedBook = $queryResult->fetchObject();

      echo "<span class='bookTitle'>{$returnedBook->bookTitle} . $bookISBN</span><br>";
      echo "<form method='get' action='updateBook.php'>
      <!--when the form is submitted, it will send its data to updateBook.php-->

Book ISBN:
<input type='text'
      name='bookISBN'
      value='$bookISBN' readonly/> <br /><br />

Title:
<input type='text'
      name='bookTitle'
      value='{$returnedBook->bookTitle}' /> <br /><br />
Year:
<input type='text'
      name='bookYear'
      value='{$returnedBook->bookYear}' /><br /><br />
Publisher ID:
<input type='text'
      name='pubID'
      value='{$returnedBook->pubID}' /><br /><br />
Category ID:
<input type='text'
      name='catID'
      value='{$returnedBook->catID}' /><br /><br />
Price:
<input type='text'
      name='bookPrice'
      value='{$returnedBook->bookPrice}' /><br /><br />

<input type=\"submit\" value=\"Update Book\">

</form>";

   } catch (Exception $e)
   {
      echo "<p>Query failed: " . $e->getMessage() . "</p>\n"; //informs the user if the query failes
   }
}
else //if the user is not logged in then display the following
{
   echo "<p>You must be logged in to access this content.</p>\n";
}
?>
</body>
</html>
```

functions.php:

```
<!--Andrew Robson - w16011147-->
<!--this file contains all of the functions that will be used throughout the rest of the website
   a great deal of this code is reusable and so serves the purpose of saving time and increasing
   efficiency by being here-->
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8"/>
   <meta name="viewport" content="width=device-width"/>
   <link rel="stylesheet" href="styling.css"> <!--Links the stylesheet-->
   <title>Functions</title>
</head>
<body>
<?php
/*
 * this function makes the connection to the database at phpmyadmin
 * Using PDO the statement will make the connection to the database
 */
function getConnection()
{
   try {
      $connection = new PDO("mysql:host=localhost;dbname=unn_w16011147",
         "unn_w16011147", "Fasran1997"); //connects to the server the data is stored on
      $connection->setAttribute(PDO::ATTR_ERRMODE,
         PDO::ERRMODE_EXCEPTION);
      return $connection;
   }
   catch (Exception $e)
   { //if the server cannot be connected to then it will throw an exception
      throw new Exception("Connection error ". $e->getMessage(), 0, $e);
   }
}

/*
 * this function takes two parameters. It sets $data to be equal to $name
 * When the function is complete it will return true to show that it has finished
 */
function setSession($name, $data) {
   $_SESSION[$name] = $data;
   return true;
}
```

```
/*
 * this function takes two parameters. It sets $data to be equal to $name
 * When the function is complete it will return true to show that it has finished
 */
function setSession($name, $data) {
   $_SESSION[$name] = $data;
   return true;
}

/*
 * This function retrieves the $name which is located within the session array
 * If the $name can't be retrieved then it will return null in its place.
 */
function getSession($name) {
   if(isset($_SESSION[$name])) {
      return $_SESSION[$name];
   }
   else {
      return null;
   }
}

/*
 * This function tests to see if the user is logged in or not, to do this it
 * checks if the value 'pass' is empty or not. If the value is empty it returns false,
 * if there is a value, it will return that value.
 */
function checkLogin() {
   if(getSession('pass')===null) { //test if 'pass' is null
      return false; //returns false indicating 'pass' was null
   }
   else {
      return getSession('pass'); //returns the value, indicating the user is logged in
   }
}

/*
 * This function is used to create the login and logout functionality on every webpage
 * I found it more efficient to make this code into a function to increase its reusability
 * when 'displayLoginLogout()' is called it will generate the html below.
 */
function displayLoginLogout() {

   if (checkLogin()) {
      echo "<form method='post' action='logout.php'> <!--Logs the user out-->
         <input type='submit' value='Logout'/> <!--logout button-->
      </form>";
   }
```

```
else {
      echo "<form method='post' action='login.php'> <!--Logs the user in-->
        <label>Username:
          <input type='text' name='username' value=''/> <!--the user needs to enter a username-->
        </label>
        <label>Password:
          <input type='password' name='password' value=''/> <!--The user needs to enter a password-->
        </label>
        <input type='submit' value='Login'/> <!--login button-->
      </form>";
   }
}

/*
 * This function is used to create the navigation for the webpages
 * Similarly to 'displayLoginLogout()' I found it more efficient to put this code into a
 * function and call upon it when necessary
 */
function displayNav() {
   if (checkLogin()) { //checks to see if the user is logged in or not before displaying
      echo "<nav>
   <ul>
      <li><a href=\"index.php\">Home</a></li>
      <li><a href=\"allBooks.php\">Display All Books</a></li>
      <li><a href=\"orderBooksForm.php\">Order Form</a></li>
      <li><a href=\"credits.php\">Credits</a></li>
   </ul>
</nav>";
   }
}
?>
</body>
</html>
```

getOffers.php:

```
<?php
try {
	// include the file for the database connection
	require_once('functions.php');
	// get database connection
	$dbConn = getConnection();
}
catch (Exception $e) {
	throw new Exception("Connection error " . $e->getMessage(), 0, $e);
}

if (isset($_REQUEST['useJSON'])) {
	// echo what getJSONOFfer returns
	echo getJSONOffer($dbConn);
}

else {	// otherwise just an html record is required

   // so echo whatever getHTMLOffer returns to the browser or back to the ajax script
   echo getHTMLOffer($dbConn);
}

function getHTMLOffer($dbConn) {
	try {
	   // store the sql for a random special offer, the sql wraps things using concat in an html
'wrapper'
	   $sql = "select concat('<p>&#8220;',bookTitle,'&#8221;<br />\n<span class=\"category
\">Category: ',catDesc,'</span><br />\n<span class=\"price\">Price: ',bookPrice,'</span></p>') as
offer from nbc_special_offers inner join nbc_category on nbc_special_offers.catID =
nbc_category.catID order by rand() limit 1";

		// execute the query
		$rsOffer = $dbConn->query($sql);

		// get the one offer returned
		$offer = $rsOffer->fetchObject();

		// return the offer
		return $offer->offer;
	}
	catch (Exception $e) {
		return "Problem: " . $e->getMessage();
	}
}
```

```
function getJSONOffer($dbConn) {
	try {
	   $sql = "select bookTitle, catDesc, bookPrice from nbc_special_offers inner
join nbc_category on nbc_special_offers.catID = nbc_category.catID order by rand()
limit 1";
		$rsOffer = $dbConn->query($sql);;
	   $offer = $rsOffer->fetchObject();
	   return json_encode($offer);
	}
	catch (Exception $e) {
		return "Problem: " . $e->getMessage();
	}
}
?>
```

index.php:

```
<!--Andrew Robson - w16011147-->
<!--This page is essentially the homepage, all of the other pages can be reached
from this page-->
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8"/>
   <meta name="viewport" content="width=device-width"/>
   <link rel="stylesheet" href="styling.css"> <!--Links the stylesheet-->
   <title>Home page</title>
</head>
<body>

<?php
require_once ('functions.php'); //links to the function.php
ini_set("session.save_path", "/home/unn_w16011147/sessionData"); //initialise the session
session_start(); //begin the session

$dbConn = getConnection(); //connect to the database

displayLoginLogout(); //call the function to display the login/logout functionality
displayNav(); //call the function to display the navigation functionality
?>
<script type="text/javascript" src="offersAJAX.js"> //connect to offersAjax.js
</script>

<script type="text/javascript">
   //<![CDATA[
   'use strict';

   window.addEventListener('load', function initialise()
   {
      var offerReturned = function() //the 'offerReturned' is the function that will be altered every 5
      //seconds
      {
         getRequest('getOffers.php', transferOffer); //this returns a request to 'getOffers.php' and transfers it
         setTimeout(offerReturned, 5000); //this calls the function 'offeredReturn' every 5 seconds
         //5000 stands for 5000 millisecondsis, which is 5 seconds
      };
      offerReturned(); //calls itself, sending itself back though code
   });
   //]]>
</script>

<aside id='offers'> <!--the id 'offers' is crucial for the AJAX to work in this context-->

</aside>



</body>
</html>
```

login.php:

```
<!--Andrew Robson - w16011147-->
<!--this page performs all of the necessary functionality for the login process-->
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8"/>
   <meta name="viewport" content="width=device-width"/>
   <title>Login</title>
</head>
<body>
<?php
require_once("functions.php"); //link to functions.php
ini_set("session.save_path", "/home/unn_w16011147/sessionData"); //initialise the session
session_start(); //begin the session

$username = filter_has_var(INPUT_POST, 'username') ? $_POST['username']: null;
//retrieves the username from the previous page
$username = trim($username); //trims the whitespace around the variable
$password = filter_has_var(INPUT_POST, 'password') ? $_POST['password']: null;
//retrieves the password from the previous page
$password = trim($password); //trims the whitespace around the variable

if (empty($username) || empty($password))
{ //if either the username or password variable are empty, do the following
   echo "<p>Your username and password were incorrect. Please try <a href='allBooks.php'>again</a>.</p>";
   //informs the user of their error and provides a link back to the book list
}
else //if there are values in username and password then continue
   {
   try
   {
      $dbConn = getConnection(); //connect to the database

      $sqlSelect = "SELECT passwordHash FROM nbc_users WHERE username = :username";
      //retrieve the hashed password where username equals what the user entered

      $queryResult = $dbConn->prepare($sqlSelect); //prepare the statement
      $queryResult->execute(array(':username'=>$username)); //execute the statement

      $pass = $queryResult->fetchObject(); //retrieve the data from the database

      if(!$pass)
      {
         echo "<p>Your username and password were incorrect. Please try <a href='allBooks.php'>again</a>.</p>";
         exit;
      }
```

```
else
      {
         $hashedPassword = $pass->passwordHash; //compare database hashed password with our own

         if(!password_verify($password, $hashedPassword))
         {
            echo "<p>Your username and password were incorrect. Please try <a href='allBooks.php'>again</a>.</
p>";
            exit;
         }
         else
         {
            setSession('pass', true); //pass is true to show that the user has successfully logged in
            echo "<p>Your username and password were correct. Click
            <a href='allBooks.php'>OK</a> to continue .</p>";
         }
      }
   }
   catch (Exception $e)
   {
      echo $e; //display any errors that have occurred
   }
}
?>

</body>
</html>
```

logout.php:

```
<!--Andrew Robson - w16011147-->
<!--This page performs the functionality required for logging out-->
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8"/>
   <meta name="viewport" content="width=device-width"/>
   <title>Logout</title>
</head>
<body>
<?php
require_once("functions.php"); //link to functions.php, allows use of functions declared within it
ini_set("session.save_path", "/home/unn_w16011147/sessionData"); //initialise the session data
session_start(); //begin a session

$_SESSION = array(); //makes the $_SESSION an empty array, therefore web pages wont load
automatically logged in

session_destroy(); //destroys the session
//Gives the users the option to return to allBooks.php
echo "<p>Logout successful. <a href='allBooks.php'>Return</a>.</p>";


?>

</body>
</html>
```

offersAJAX.js:

```
/*
   Andrew Robson - w16011147
   This page will determine the type of object to create as well as pass in the offer
   to the index page
 */

function getRequest(url, callbackFunction)
{
   var httpRequest; //creates the variable: httpRequest

   if (window.XMLHttpRequest) //if the window is XMLHttpRequest compatible, proceed
   {
      httpRequest = new XMLHttpRequest(); //change the httpRequest to a XMLHttpRequest
   }
   else if (window.ActiveXObject) //if the window is ActiveXObject compatible, proceed
   {
      try
      {
         //change the httpRequest variable to a ActiveXObject variable
         httpRequest = new ActiveXObject("Msxml2.XMLHTTP");
      }
      catch (exception)
      {
         try
         {
            //change the variable httpRequest to a different ActiveXObject
            httpRequest = new ActiveXObject("Microsoft.XMLHTTP");
         }
         catch (exception)
         {
         }
      }
   }

if (!httpRequest) //check if the httpRequest was not made
   {
      alert('An error has occurred, XMLHTTP has failed');
   }
```

```
else
   {
      httpRequest.onreadystatechange = function() //when httpRequest is in the ready state, proceed
      {
         var completed = 4, successful = 200, returnValue; //declaring variables
         if (httpRequest.readyState == completed) //if httpRequest has completed its ready state, proceed
         {
            if (httpRequest.status == successful) //if httpsRequest's status is successful, proceed
            {
               //set 'returnValue' to be equal to the response text from httpRequest
               returnValue = httpRequest.responseText;
               callbackFunction(returnValue);
            }
            else
               {
               alert("Problem with request");
            }
         }
      };
      httpRequest.open('get', url, true);
      httpRequest.send(null);
   }
}

function transferOffer(offer) //used to display the offers
{
   var retrieveAside = document.getElementById('offers'); //retrieves the offers id and stores it in retrieveAsside
   var asideOffer = "<h1>Amazing Deals:</h1>\n"; //adds a header to the new offer
   asideOffer = asideOffer + offer;

   retrieveAside.innerHTML = asideOffer; //displays the offer
}
```

orderBooksForm.php:

```
<!--Andrew Robson - w16011147-->
<!--this file will perform the javascript for the orderBooksForm-->
<!doctype html>
<html lang="en">
<head>
        <meta charset="UTF-8" />
   <link rel="stylesheet" href="styling.css"> <!--Links the stylesheet-->
        <title>Order Books Form</title>
</head>
<body>



<?php
require_once ('functions.php'); //links this page to the functions.php page
ini_set("session.save_path", "/home/unn_w16011147/sessionData"); //initialises the session data
session_start(); //starts a session
displayLoginLogout(); //displays the logout functionality
displayNav(); //displays the navigation functionality

if (checkLogin()) //if the user is logged in, proceed
{
   /* This code dynamically generates a web page containing a form designed with the html required to display one
   checkbox for each of the books currently held in the nbc_books database table has been provided for you in the
   assessment section for the module on blackboard. The user can select one or more books that they are interested in
   ordering by clicking the checkboxes.
   Use the browser to look at the structure of the html generated by the php code. */

   $url  = "http://unn-izge1.newnumyspace.co.uk/KF5002/assessment/orderBooksFormInc.php";
   $curl = curl_init($url);
   curl_setopt($curl, CURLOPT_RETURNTRANSFER, 1);
   $result = curl_exec($curl);
   curl_close($curl);
   echo $result;
}
else
{
   echo "<p>You must be logged in to access this content.</p>\n";
}
?>

<!-- Here you need to add Javascript or a link to a script to process the form as required for the assignment -->

<script type="text/javascript">
   //<![CDATA[
   'use strict';
```

```
window.addEventListener('load', function initialise() //when the window first loads, call the initialise function
  {
    var form = document.getElementById('orderForm'); //declares 'orderForm' as the object: 'form'
    //calls 'dataValidation' function when a change is made within the form
    form.addEventListener('change', dataValidation);
    //calls 'dataValidation' function when a change is made on forename
    form.forename.addEventListener('change', dataValidation);
    //calls 'dataValidation' function when a change is made on surname
    form.surname.addEventListener('change', dataValidation);
    //calls 'dataValidation' function when a change is made on companyName
    form.companyName.addEventListener('change', dataValidation);
    //hides the customer detail fields because they automatically appear on startup
    form.querySelector('#retCustDetails').style.visibility = "hidden";
    form.total.value = "0.00"; //states the form.total.value as being "0.00"

    /*
       this function performs the functionality to alter the text in line with the
       terms and conditions checkbox
     */
    function changeTermsAndConditions()
    {
       var termAndCond = form.termsChkbx; //converts form.termsChkbx into a variable
       //converts the text in line with the checkbox into a variable
       var textLine = form.querySelector('#termsText');

       if (termAndCond.checked) //if the checkbox has been ticked then proceed
       {
          textLine.style.color = '#000000'; //makes the font colour black
          textLine.style.fontWeight = '400'; //makes the boldness normal

          return true;
       }
       else
       {
          textLine.style.color = '#ee000b'; //makes the font colour red
          textLine.style.fontWeight = '700'; //makes the font style bold

          return false;
       }
    }
```

```
/*
   This function is used to validate the input fields that the user will be utilising
   it hides and makes the input fields visible
   it also checks the length of the data input to test it
 */
function userOption()
{
   var customerOption = form.customerType; //converts form.customerType into a variable: customerOption
   //allows for access to the "hidden" field so we can alter it
   var customer = form.querySelector('#retCustDetails');
   //allows for access to the "hidden" field so we can alter it
   var trade = form.querySelector('#tradeCustDetails');

   if (customerOption.value === "") //if nothing has been selected (because the string is empty) proceed
   {
      customer.style.visibility = "hidden"; //change the visibility to hidden
      trade.style.visibility = "hidden"; //change the visibility to hidden

      return false;
   }
   else if (customerOption.value === "ret") //if the value = "ret" then proceed
   {
      customer.style.visibility = "visible"; //show the customer input fields
      trade.style.visibility = "hidden"; //hide the companyName input field

      //trim both of the variables and check if it is empty
      if (form.forename.value.trim === '' || form.surname.value.trim() === '')
      {
         return false;
      }

      return true;
   }
   else if (customerOption.value === "trd") //if the value = "trd", proceed
   {
      customer.style.visibility = "hidden"; //hide the customer fields
      trade.style.visibility = "visible"; //show the companyName field

      if (form.companyName.value.trim === '') //trim the companyName field and check if it is empty
      {
         return false;
      }

      return true;
   }
}
```

```
/*
    this function will be used to validate all of the data at once, deciding whether the user can submit or not
 */
function dataValidation()
{
    var purchase = productPrice(); //convert the function result to the variable: purchase
    //convert the function result to the variable: termsAndconditions
    var termsAndConditions = changeTermsAndConditions();
    var user = userOption(); //convert the function result to the
    var submit = form.submit; //convert the submit button into a variable called submit

    if (purchase && termsAndConditions && user) //if all of these variables are true, proceed
    {
        submit.disabled = false; //enable the submit button
    }
    else
    {
        submit.disabled = true; //disable the submit button
    }
}

/*
    this function is used to calculate the price of the products ticked, including delivery
 */
function productPrice()
{
    //return all of the checked checkboxes and store them in a variable
    var numberOfCheckboxes = form.querySelectorAll('.item input[type=checkbox]:checked');
    var totalField = form.total; //converts form.total into a variable called totalField
    var price = 0.00; //creates price variable
    var iteration = 0; //creates iteration variable
    var test = false; //creates a test boolean

    //adds the price of each checkbox together and displays it in the total field
    for (iteration; iteration < numberOfCheckboxes.length; iteration += 1)
    {
        var currentCheckBox = numberOfCheckboxes[iteration];

        price = price + parseFloat(currentCheckBox.dataset.price);
    }
    if (numberOfCheckboxes.length > 0) //compares the amount of checkboxes to 0
    {
        price = price + deliveryPrice();
        test = true; //set the variable test to be true
    }
    totalField.value = price.toFixed(2);
    return test; //return test (true)
}
```

```
/*
        this function calculates the price of the delivery
     */
    function deliveryPrice()
    {
        var price = 0.00; //the overall price of the delivery
        //returns all of the checked radio buttons and stores them in a variable
        var numberOfRadioButtons = form.querySelectorAll('input[type=radio]:checked');
        var iteration;
        //iterates through the amount of radio buttons that have been checked
        for (iteration = 0; iteration < numberOfRadioButtons.length; iteration += 1)
        {
            var currentRadioButton = numberOfRadioButtons[iteration];

            price = price + parseFloat(currentRadioButton.dataset.price);
        }
        return price;
    }
  });
  //]]>
</script>
</body>
</html>
```

styling.css:

```
.bookOption {
    clear: both;
    /* displays when both sides are clear */
    padding: 0.1em 2em;
    /* defines the size of the padding */
    color: #000000;
    /* defines the colour of the text */
    border-bottom: solid 1px #BDBDBD;
    /* defines the margins properties */
    margin: 0;
    /* Defines the size of the margin */
}

nav {
    float: left;
    /* Decides which way the tag/object is floated */
    width: 1920px;
    /* Defines how wide the tag/object is */
    height: 200px;
    /* Defines the height of the tag/object */
    font-family: arial, helvetica, sans-serif;
    /* Defines the font families, these are default among many different browsers */
    font-size: x-large;
    /* Defines the size that the font will be */
}

nav ul {
    float: left;
    /* Decides which way the tag/object is floated */
    list-style: none;
    /* Defines the list style */
    margin: 0;
    /* Defines the size of the margin */
    padding: 20px;
    /* Defines the size of the padding */
    width: 200px;
    /* Defines how wide the tag/object is */
    text-align: center;
    /* Defines the alignment of the text */
}

nav li {
    margin-bottom: 10px;
    /* Defines the size of the margin */
    margin-top: 10px;
    /* Defines the size of the margin */
    display: block;
    /* Displays the element as a block element */
    background-color: #9c9999;
    /* Defines the colour of the background */
    border: solid thin #464646;
    /* Creates a border around the selected element */
}
```

```
nav li a:link, a:visited {
   display: block;
   /* Displays the element as a block element */
   text-decoration: none;
   /* Displays any text decoration on the element */
   color: #7b4ec3;
   /* Defines the colour of the text */
}

nav li a:hover {
   background-color: #ddd4d7;
   /* Defines the colour of the background */
   color: #c3a520;
   /* Defines the colour of the text */
}
```

updateBook.php:

```
<!--Andrew Robson - w16011147-->
<!--this page holds the functionality that updates the book and executes the changes to the
database-->
<!--the user will need to be logged in if they want to be able to update a book-->
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8"/>
   <meta name="viewport" content="width=device-width"/>
   <link rel="stylesheet" href="styling.css"> <!--Links the stylesheet-->
   <title>Update Book</title>
</head>
<body>
<?php
require_once("functions.php"); //link to functions.php
ini_set("session.save_path", "/home/unn_w16011147/sessionData"); //initialises session
session_start(); //starts the session
displayLoginLogout(); //displays the login/logout functionality
displayNav(); //displays the navigation functionality

if (checkLogin()) //if the user is logged in, proceed
{
   $errors = false; //declare errors as false since it is the start of the program
   //retrieves the bookISBN from the previous page
   $bookISBN  = filter_has_var(INPUT_GET, 'bookISBN') ? $_GET['bookISBN'] : null;
   //retrieves the bookTitle from the previous page
   $bookTitle = filter_has_var(INPUT_GET, 'bookTitle') ? $_GET['bookTitle'] : null;
   //retrieves the bookYear from the previous page
   $bookYear  = filter_has_var(INPUT_GET, 'bookYear') ? $_REQUEST['bookYear'] : null;
   //retrieves the catID from the previous page
   $catID     = filter_has_var(INPUT_GET, 'catID') ? $_GET['catID'] : null;
   //retrieves the pubID from the previous page
   $pubID     = filter_has_var(INPUT_GET, 'pubID') ? $_GET['pubID'] : null;
   //retrieves the bookPrice from the previous page
   $bookPrice = filter_has_var(INPUT_GET, 'bookPrice') ? $_GET['bookPrice'] : null;

   $bookISBN  = trim($bookISBN); //trims the whitespace around the variable
   $bookTitle = trim($bookTitle); //trims the whitespace around the variable
   $bookYear  = trim($bookYear); //trims the whitespace around the variable
   $catID     = trim($catID); //trims the whitespace around the variable
   $pubID     = trim($pubID); //trims the whitespace around the variable
   $bookPrice = trim($bookPrice); //trims the whitespace around the variable
```

```
if (empty($bookTitle)) //checks if the variable is empty
  {
    echo "<p>Please enter a title for the book.</p>\n";
    $errors = true;
  }

  if (empty($bookYear)) //checks if the variable is empty
  {
    echo "<p>Please enter a year for the book.</p>\n";
    $errors = true;
  }

  else {
    if (!filter_var($bookYear, FILTER_VALIDATE_INT)) //validates the bookYear by making sure it is an integer
    {
      echo "<p>Please enter the year as a number e.g. 2008</p>\n";
      $errors = true;
    }
  }

  if (empty($catID)) //checks if the variable is empty
  {
    echo "<p>Please enter a category ID for the book.</p>\n";
    $errors = true;
  }

  else {
    //makes sure that only the alphabet is being used, no symbols
    if (!filter_var($catID, FILTER_VALIDATE_REGEXP, array("options" => array("regexp" => "/^[A-Z]+$/i"))))
    {
      echo "<p>Please enter the catID using only letters, no symbols.</p>\n";
      $errors = true;
    }
  }

  if (empty($pubID)) //checks if the variable is empty
  {
    echo "<p>please enter a pubID for the book.</p>\n";
    $errors = true;
  }

  else
  {
    //makes sure that the alphabet is being used, no symbols
    if (!filter_var($pubID, FILTER_VALIDATE_REGEXP, array("options" => array("regexp" => "/^[A-Z]+$/i"))))
    {
      echo "<p>Please enter the pubID using only letters, no symbols.</p>\n";
      $errors = true;
    }
  }
```

```
    if (empty($bookPrice))
    {
        echo "<p>Please enter a price for the book.</p>\n";
        $errors = true;
    }

    else
    {
        if (!filter_var($bookPrice, FILTER_VALIDATE_FLOAT)) //makes sure that the variable is a floar
        {
            echo "<p>Please enter the price as a number followed by a decimal number e.g. 23.56</p>\n";
            $errors = true;
        }
    }

    if ($errors)
    {
        echo "<p>One or more errors have occurred, please try <a href='allBooks.php'>again</a></p>\n";
    }

    else
    {
        try
        {
            $dbConn = getConnection(); //connects to the database

            //updates the database
            $sqlUpdate = "UPDATE nbc_books
              SET  bookTitle = '$bookTitle', bookYear = '$bookYear',
              pubID = '$pubID', catID = '$catID', bookPrice = '$bookPrice'
              WHERE bookISBN = $bookISBN";
            $dbConn->exec($sqlUpdate);

            echo "<p>Book has been successfully updated</p>\n";
        }
        catch (Exception $e)
        {
            echo "<p>Book not updated: " . $e->getMessage() . "</p>\n";
        }
    }
}
else
{
    echo "<p>You must be logged in to access this content.</p>\n";
}
?>
</body>
</html>
```