**Checking Out**

After your customers have had the opportunity to select products from the online store, they'll want to check out and purchase the items. The checkout process is a three-part adventure:

1. The customer logs into the storefront with an existing account or registers for a new account.
2. The customer either confirms or modifies the order.
3. The storefront application program creates a record for the new order with the proper information for processing.

The Food Store application handles these three processes using several code files, as shown.

The checkout process program flow

The first step in the process is to determine if the customer is a returning customer. The Food Store keeps information about current customers in the *customers* table. This helps shorten the checkout process for returning customers, as they don't have to enter any shipping information because you already have it.

When the customer is ready to check out, the navigation bar provides a link to the index.php file using the checkout.inc.php include file. This is the starting point of the checkout process. Let's build that code file.

1. Create the file *checkout.inc.php* in the store folder in your application area.
2. Open the file with a text editor, and add the following code:

<h2><u>Check Out Procedure</u></h2><br><br>

<p>If you are a returning customer, please enter your e-mail address and password<p>

<form action="verifycust.php" method="post">

<table width="50%" cellpadding="1" border="1">

<tr>

<td>e-mail address</td>

<td><input type="text" name="email"></td>

</tr>

<tr>

<td>password</td>

<td><input type="password" name="password1"></td>

</tr>

</table>

<input type="submit" name="button" value="Login">

</form>

<br><br><br>

<form action="index.php" method="get">

<input type="submit" name="button" value="Click here if you're a new customer">

<input type="hidden" name="content" value="newcust">

</form>

1. Save the file and exit the editor.

This code creates two separate forms on the Web page. The first form asks for the e-mail address and password if the person is a returning customer. If you've never used them before, the *password* type of input form works just like the text form but hides the text that's typed in it. This helps keep some sense of security as your customer enters a password.

The other form on the Web page is just a simple submit button for the person to click if they're a new customer. This sends the customer to another page to enter his or her personal information (which we'll walk through in a little bit). Each form has its own action attribute to pass control off to a different code file, depending on which form button the customer selects.

The customer checkout page

If the customer is a returning customer, the form action sends the data to the verify.php file. (Note that this isn't part of the index.php include files; I'll explain that in a little bit). If the customer is a new customer, the form action sends the data to the index.php file, using the newcust.inc.php include file.

First, let's take a look at how to handle returning customers.

**Processing Returning Customers**

If the customer is a returning customer, he or she will already have a record on file in the customers table. The login form asks for his or her e-mail address and password. You'll need to verify the e-mail address and password provided. We'll use the verifycust.php file to do that.

By making this a separate file apart from the index.php include files, we can use the header() function to automatically redirect the customer to the next step in the process without any manual intervention on the customer's part. After the program verifies the customer login information, it passes directly to the next step in the checkout process.

Let's create the verifycust.php file and then take a look at what it does.

1. Create the file *verifycust.php* in the store folder of the application area.
2. Open the file in a text editor and enter the following code:

<?php

session\_start();

include("mylibrary/login.php");

login();

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

$password = $\_POST['password1'];

$query = "SELECT \* from customers where email = '$email' and password = PASSWORD('$password')";

$result = mysql\_query($query);

$row = mysql\_num\_rows($result);

if ($row)

{

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

$custid = $row['custid'];

$\_SESSION['cust'] = $custid;

header("Location: index.php?content=confirmorder");

} else

{

echo "<h2>Sorry, Unable to verify customer</h2>\n";

echo "<a href=\"index.php?content=checkout\">Go back to check out</a>\n";

}

?>

1. Save the file and exit the editor.

Because this isn't part of the index.php code, you'll have to use the session\_start() and login() functions here to access the session cookies and log into the MySQL server. After retrieving the form data, it prepares a SELECT query to check the e-mail address and password in the customers table. The password is stored in the customer table using the *PASSWORD()* function, which produces a 41-byte encrypted password from a text string. You must also use the PASSWORD() function in the SQL query to generate the encrypted password based on the text value entered into the form.

If they match, it creates a new session cookie called *cust*, places the assigned customerid value from the table in it, then redirects the customer's browser to the next Web page in the checkout process. This is the *confirmorder* page.

If the user account and password values don't match what's in the database, the customer is asked to go back to the main checkout page and try again. This just provides a quick link back to the checkout Web page.

Let's put the new customer registration programs off to the side for a little while and follow the flow to the next step in the process, the confirmorder section. This is where the customer can confirm or reject the order.

**Confirming an Order**

After the returning customer has logged in (or a new customer has registered), you want to offer them the ability for one last chance to change their order. The *confirmorder.inc.php* include file provides them that last chance.

Let's create this file for the application.

1. Create the file *confirmorder.inc.php* in the store folder in the application area.
2. Open the file in a text editor and add the following code:

<?php

echo "<h2><u>Confirming Order</u></h2><br>\n";

$total = 0;

echo "<table width=\"100%\" cellpadding=\"1\" border=\"1\">\n";

echo "<tr><td>Product</td><td>Price</td><td>Quantity</td><td>Total</td>\n";

foreach($\_SESSION['cart'] as $prodid => $quantity)

{

$query = "SELECT description, price FROM products WHERE prodid = $prodid";

$result = mysql\_query($query);

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

$description = $row['description'];

$price = $row['price'];

$subtotal = $price \* $quantity;

$total += $subtotal;

printf("<tr><td>%s</td><td>%s</td><td>%d</td><td>$%.2f</td></tr>\n",

$description, $price, $quantity, $subtotal);

}

printf("<tr><td colspan=\"3\"><b>Total</b></td><td>$%.2f</td></tr>\n", $total);

echo "</table>\n";

echo "<form action=\"index.php\" method=\"post\">\n";

echo "<input type=\"hidden\" name=\"content\" value=\"finishorder\">\n";

echo "<input type=\"submit\" name=\"button\" value=\"Confirm order\">\n";

echo "</form>\n";

echo "<form action=\"index.php\" method=\"post\">\n";

echo "<input type=\"hidden\" name=\"content\" value=\"reviewcart\">\n";

echo "<input type=\"submit\" name=\"button\" value=\"Change Order\">\n";

echo "</form\">\n";

?>

1. Save the file and exit the editor.

The confirmorder.inc.php file extracts the information from the shopping cart and displays it in a table in the main section of the storefront Web page. This page creates two buttons after the shopping cart information. The customer has the option to return to the review cart page to modify the cart contents or proceed with checking out.

The confirm order page

|  |
| --- |
| **A word about payments:**Obviously, getting your customers to pay for their purchases is an important part of any Web storefront. However, this can be a dangerous activity to take on by yourself. The last thing you need is to have your database, full of credit card numbers, stolen from your server. If you're putting a storefront application online, look into one of the many third-party companies that process credit card information for you, such as the popular PayPal site. These companies provide all the HTML code required to forward customers to their site for payment. All you need to do is plug in your vendor code and the payment total. |

That's enough coding for this chapter. In Chapter 3, we'll pick up the checkout story by showing how to register a new customer.