Classic Shopping Cart Application

Garfield Pet Shop

Bring me an order of everything,
with a side order of everything else !

http://garfield.free-web-hosting.biz/

Garfield Pet Shop

This shopping system has implemented the following functionalities:
• a database of the products we want to sell online;
• an online catalog of products, listed by category;
• a shopping cart to track the items a user wants to buy;
• a user's personal account for additional facilities;
• a checkout script that processes payment and shipping details;
• an administration interface;
• connexed sections;
• newsletter.

The requirements for this shopping system are:
• The system will connect the database to a user's browser. Users should be able to browse items by category.
• Users should also be able to select items from the catalog for later purchase. The system will track the items they have selected.
• When clients have finished shopping, the system must total up their order, take their delivery details, and process their payment.
• In order to benefit from extra facilities, users may create on-site accounts. Thus their personal data which are needed for fulfilling the order shall be automatically taken from the database.
•An administrator interface should also be build so that the administrator could add and edit products and categories on the site.
• Also, besides the possibility of buying products offered by the virtual pet shop, the user may access connexed sections of the site in the main menu.
• To the purpose of keeping contact with registered users a newsletter may be implemented containing information on newly added products or new facilities of the site.

Solution Components

Building an Online Catalog

The implemented database shall contain tables to the purpose of storing users' data, clients' data needed in order to send the order, order details, data on available products, data needed in order to implement the newsletter.

Upon completion of clients' orders transactions shall be used. For these tables they shall have to use the InnoDB storage engine.

NOTE:
Regarding that the MySQL client used for this project does not support InnoDB engine I was forced to use another engine like MyISAM type.

Tracking a User’s Purchases in Shopping Process

In order to track a user’s purchases while he shops I`ll use a session variable.
A session variable must be a design or set of variables to store a user’s selections. When a user finishes shopping and pays for his purchases, this information must be put in the database as a record of the transaction.

This data can be used to give a summary of the current state of the cart so that a user may know at any given time how much he is planning to spend.

Payment

In this project, I will add up the user’s order and take the delivery details but not actually process payments. Many payment systems are available, and the implementation for each one is different. A dummy function written for this application can be replaced with an interface to chosen system.

The payment system will transmit data to a bank, and return a success code of one of many different types of error codes. The payment system will need information from the customer such as a credit card number, identifying information from shopping system`s owner to specify which merchant account is to be credited, and the total amount of the transaction.

The total of an order from a user’s shopping cart will be computed based on the session variable. The final order details will be recorded in the database, and the session variable will be destroyed.

Administration Interface

The administrator interface will let the administrator add, delete, and edit products and categories from the database.

One common way of editing is altering the price of an item (for a special offer or sale). This means that when a customer’s order is stored, the price he paid for an item should also be stored. This means that if the customer has to return or exchange the item, we will give his the right amount of credit.

Also, this interface has implemented the possibility of obtaining various reports referring to orders carried out, products sold and products in store. This application shall only have a limited set of reports, but these may be extended in a subsequent version.

Observation:

In order to exemplify the facilities for updating the products according to orders, the application shall accomplish this function automatically after the user’s order has been recorded and transmitted. A real application will involve a certain delay, an amount of time needed for the prior verifications of the client as well as the actual expedition of the order.

Solution Overview

There are two basic views of the system: the user view and the administrator view. The user view can be inside or outside an account. Two system flows are designed considering the functionality required one for each view.
These are shown in *Figure 1* and *Figure 2*, respectively.

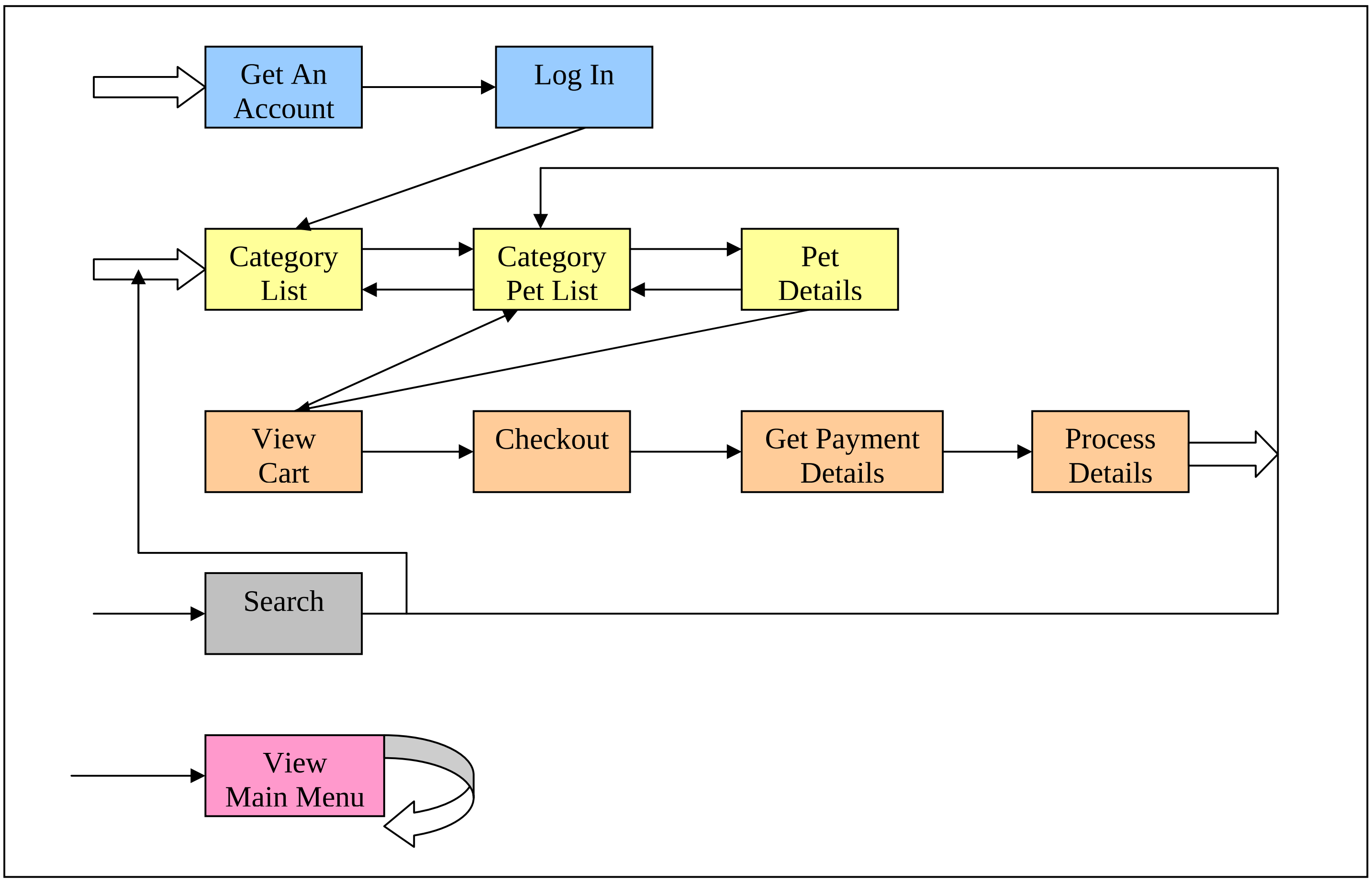


FIGURE 1
*The user view of the **Pet Shop** system allows users to view the main meniu, search products by criterion, browse pets by category, view pet details, add pets to their cart, and purchase them..*

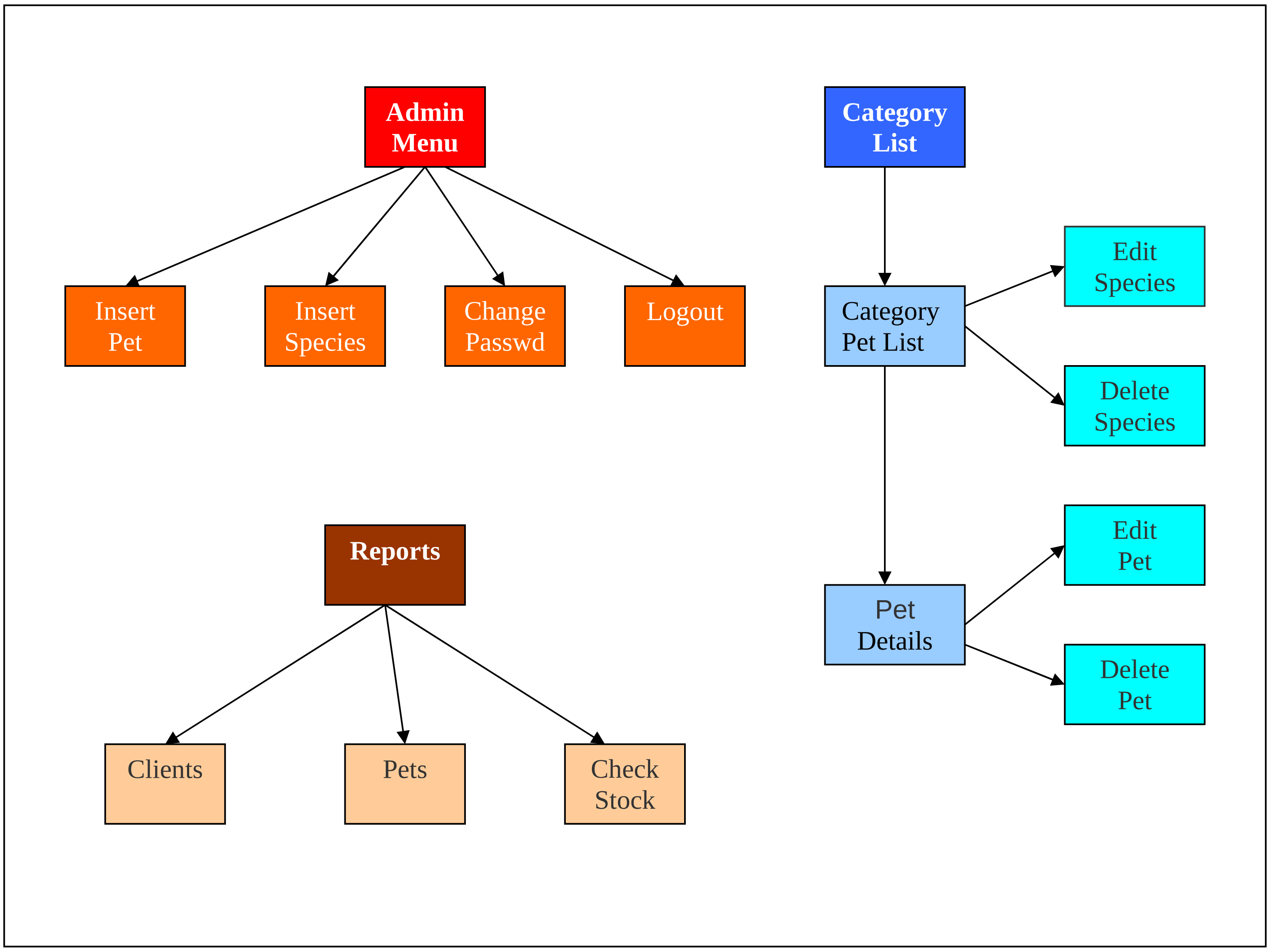


FIGURE 2
The administrator view of the Garfield system allows insertion, editing, and deletion of pets and categories as well as generating reports

In *Figure 1*, the main links between scripts are shown in the user part of the site. A customer will first come to the main page, which lists all the categories of books in the site. From there, he can go to a particular category of pets, and from there to an individual pet's details. He could also browse in main meniu, get an account or log in.

The user has a link to add a particular pet to his cart. From the cart, he will be able to check out of the online store.

Figure 2 shows the administrator interface. These scripts allow an administrator to log in and insert pets and categories and view reports.

For editing and deletion of books and categories implementation is showing the administrator a slightly different version of the user interface to the site. The administrator will still be able to browse categories and books, but instead of having access to the shopping cart, the administrator will be able to go to a pet or species and edit or delete it.

The three main code modules for this application are:

• Catalog.

• Shopping cart and order processing.
• Administration.

The concept solution applied here is to build and use a set of function libraries. The parts of the code that output HTML will be confined to a single library to support the principle of separating logic and content and, to make the code easier to read and maintain.

The database name is *pet_shop*.

A summary of the files in the application is shown in *Table 1*.

TABLE 1 Files in the Shopping Cart Application

Name	Module	Description
index.php	Catalog	Main front page of side for users. Shows the user a list of categories in the system.
show_sp.php	Catalog	Shows the user all pets from a species; the administrator has a link to *insert_pets_form.php* where he can modify or erase the current specie.
show_pet.php	Catalog	Shows the user details of a pet; the administrator has a link to *insert_pets_form.php* where he can modify or erase the current pet.
newresults.php	Catalog	Display results of search paginated after a restricted number of lines.
checkout.php	Shopping cart	Presents the user with complete order details. Gets shipping details.
purchase.php	Shopping cart	Gets payment details from user.
show_cart.php	Shopping cart	Shows the user the contents of the shopping cart. Also used to add items to the cart.
process.php	Shopping cart	Process payment details and adds order to database.
register_form.php	Users	Form allowing the registration of a new user.
admin.php	Administration	Administration meniu for a normal user and reports selector for the administrator.
login.php	Administration	Allows administrator to log in to make changes.
change_password_form.php	Administration	Form to let administrator change log password.
reports.php	Administration	Displays report selected by the administrator.
insert_pets_form.php	Administration	Form allowing administrator to add and edit species and pets
error.php	General	Displayed when the page to be displayed for the user is under work.
resize_image.php	General	Redimensions images introduced by administrator.
db.php	Functions	Collection of functions for connecting to

		database.
orders.php	Functions	Collection of functions for storing and retrieving order data.
outputs.php	Functions	Collection of functions for outputting HTML.
pets.php	Functions	Collection of functions for storing and retrieving pet data.
validation.php	Functions	Collection of functions for validating input data.
includ_pet_fns.php	Functions	File for inclusion of all functions for project files.
administration.php	Functions	Collection of functions used by administrative scripts.
authentification.php	Functions	Collection of functions for the authentication of administrative users.
form.php	Functions	Functions for safe inputs and right formatted outputs.
formValidation.js	General	Validate data introduced in forms at browser level.
rollOver.js	General	Accomplish the roll-over effect of various menus.
pet_style.css	General	Control page display of various elements.
pet_shop.sql	SQL	SQL to setup database.
populate.sql	SQL	SQL to insert some sample data into database.

NOTE:
In order for the code in this project to work as written, it will need to have magic quotes switched on.
Magic quotes are enabled on a per-directory basis in an .htaccess file with the directive php_value magic_quotes_gpc on or setting

```
; Magic quotes for incoming GET/POST/Cookie data.
magic_quotes_gpc = On
; Magic quotes for runtime-generated data, e.g. data from SQL, from exec(), etc.
magic_quotes_runtime = On
; Use Sybase-style magic quotes (escape ' with '' instead of \').
magic_quotes_sybase = On
```

in *php.ini* file.

Implementing the Database

The SQL to create the *pet_shop* database is shown in *Listing 1*.

LISTING 1 pet_shop.sql—SQL to Create the pet_shop Database

```
create database pet_shop;

use pet_shop;

create table customers
 (
  customerid int unsigned not null auto_increment primary key,
  name char(60) not null,
  address char(80) not null,
  city char(30) not null,
  state char(20),
  zip char(10),
  phone varchar(30),
  country char(60) default 'USA',
  username char(16)
) type=InnoDB;

create table orders
(
  orderid int unsigned not null auto_increment primary key,
  customerid int unsigned not null,
  amount float(6,2),
  date date not null,
  order_status char(10),
  ship_name char(60) not null,
  ship_address char(80) not null,
  ship_city char(30) not null,
  ship_state char(20),
  ship_zip char(10),
  ship_country char(20) default 'USA'
) type=InnoDB;

create table pets
(
   petid char(13) not null primary key,
   race char(40),
   spid int unsigned,
   price float(8,2) not null,

   description text
) type=InnoDB;

create table species
 (
  spid int unsigned not null auto_increment primary key,
  spname char(60) not null
) type=InnoDB;

create table order_items
 (
  orderid int unsigned not null,
  petid char(13) not null,
  item_price float(8,2) not null,
  quantity tinyint unsigned not null,
  primary key (orderid, petid)
) type=InnoDB;

create table users
(
  username char(16) not null primary key,
  password char(40) not null,
            email char(100),
            realname not null char(100),
            admin tinyint not null
) type=InnoDB;

create table news
 (
  newsid varchar(32) not null ,
  email varchar(40) not null primary key,
            status ENUM('active', 'inactive') default 'inactive' not null,
            date date not null,
            customerid int unsigned not null
) type=InnoDB;

grant select, insert, update, delete
on pet_shop.*
to pet@localhost identified by 'garfield';
```

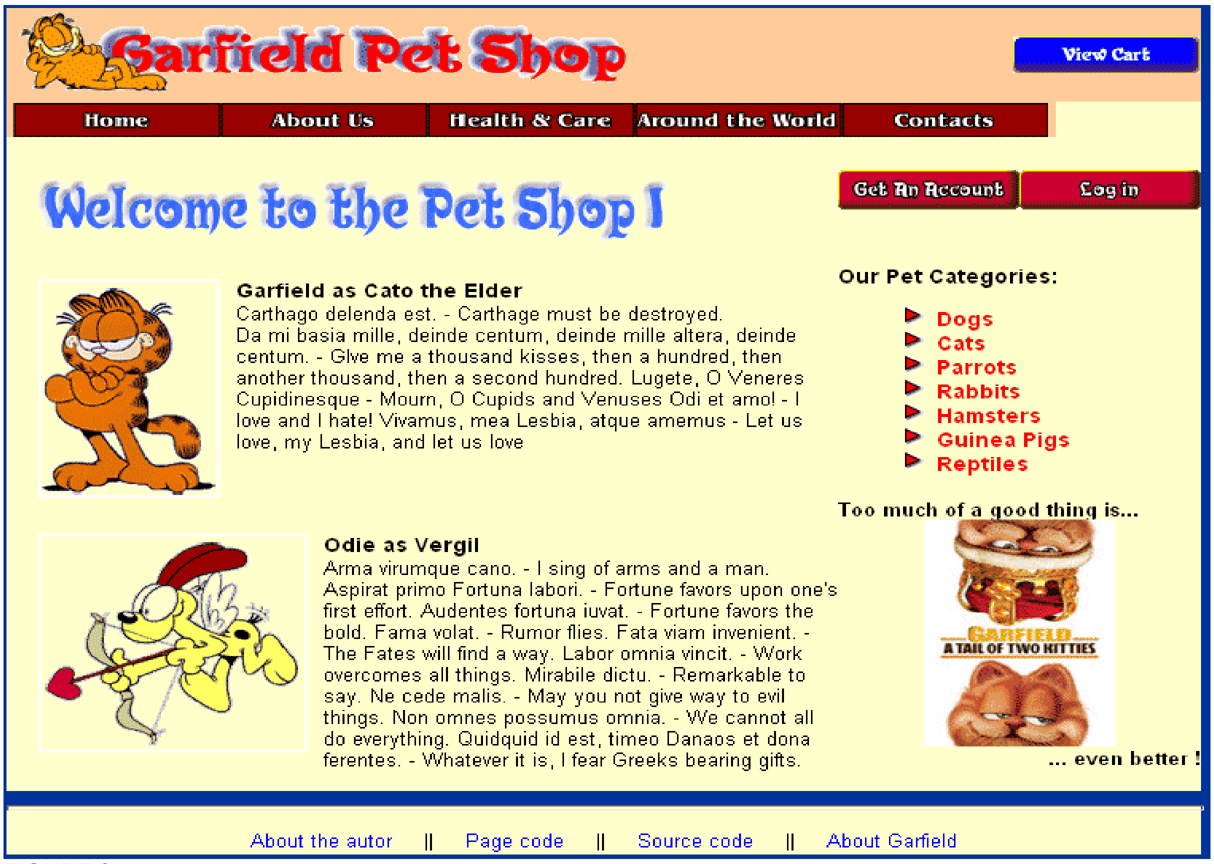
Some important notes related to *pet_shop* database:
It was implemented:
• address fields for customers; we must know where to send user`s order.
• a shipping address to an order; a customer's contact address might not be the same as the shipping address, particularly if he is using the site to buy a gift.
• a categories table and a catid to pets table; sorting pets into species will make the site easier to browse.
• *item_price* to the *order_items* table to recognize the fact that an item's
price might change; we want to know how much it costed when the customer ordered it.
• an *users* table to store user login and password details and to recognize if user is an admin to.

Observation:
Each pet has a description field which will contain a brief blurb about the pet.
It has included a file of sample data called populate.sql.

Implementing the Online Catalog

There are three catalog scripts in this application: the main page, the category (species) page, and the pet details page.

The front page of the site is produced by the script called *index.php*. The output of this script is shown in *Figure 3.*



Welcome to the Pet Shop !

Garfield as Cato the Elder
Carthago delenda est. - Carthage must be destroyed.
Da mi basia mille, deinde centum, deinde mille altera, deinde centum. - Give me a thousand kisses, then a hundred, then another thousand, then a second hundred. Lugete, O Veneres Cupidinesque - Mourn, O Cupids and Venuses Odi et amo! - I love and I hate! Vivamus, mea Lesbia, atque amemus - Let us love, my Lesbia, and let us love

Odie as Vergil
Arma virumque cano. - I sing of arms and a man.
Aspirat primo Fortuna labori. - Fortune favors upon one's first effort. Audentes fortuna iuvat. - Fortune favors the bold. Fama volat. - Rumor flies. Fata viam invenient. - The Fates will find a way. Labor omnia vincit. - Work overcomes all things. Mirabile dictu. - Remarkable to say. Ne cede malis. - May you not give way to evil things. Non omnes possumus omnia. - We cannot all do everything. Quidquid id est, timeo Danaos et dona ferentes. - Whatever it is, I fear Greeks bearing gifts.

Our Pet Categories:

- Dogs
- Cats
- Parrots
- Rabbits
- Hamsters
- Guinea Pigs
- Reptiles

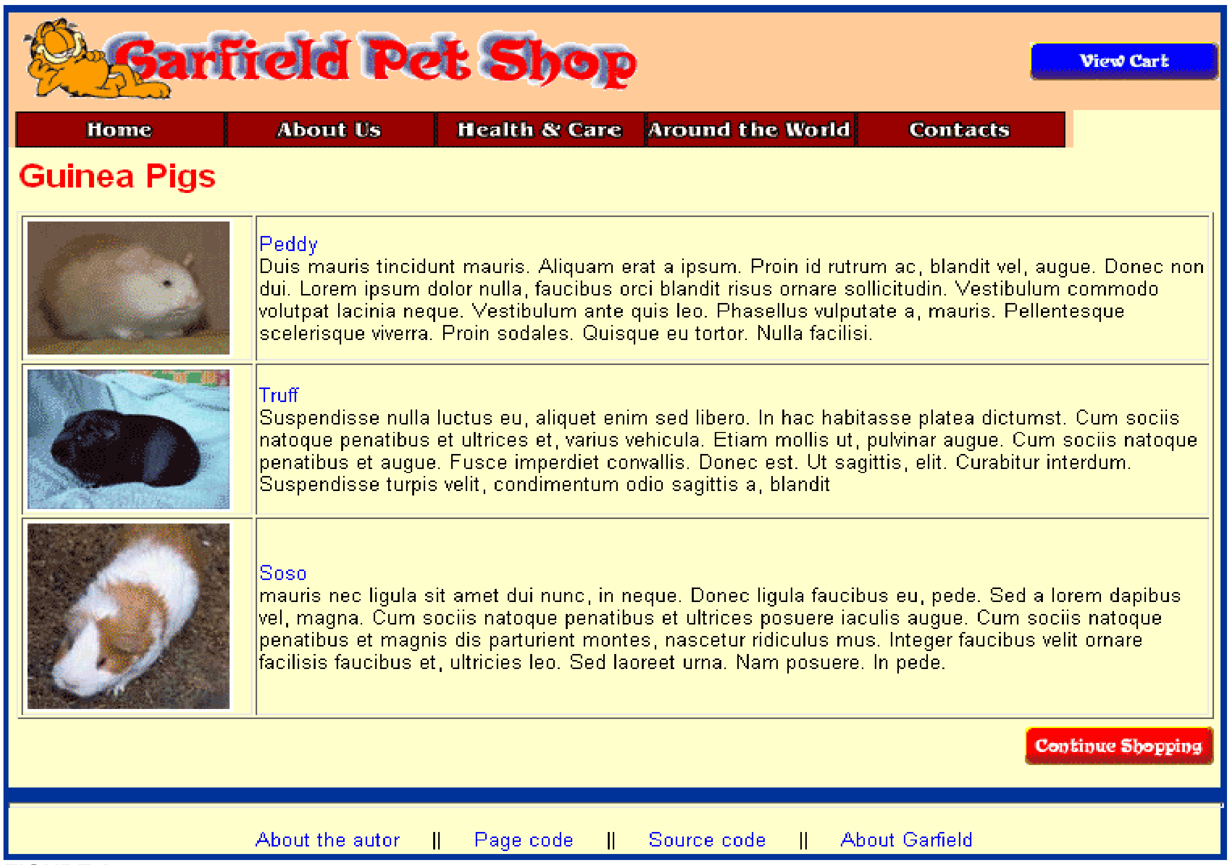
Too much of a good thing is...

... even better !

About the autor || Page code || Source code || About Garfield

FIGURE 3
The front page of the site lists the categories of pets available for purchase.

In addition to the list of categories on the site, there is a link to the shopping cart in the top-right corner of the screen and some summary information about what's in the cart. This will appear on every page while a user browses and shops.
If a user clicks on one of the categories, he'll be taken to the category page, produced by the script *show_sp.php*. The category page for the Internet pets section is shown in *Figure 4.*
All the pets in the Internet category are listed as links. If a user clicks one of these links, he will be taken to the pet details page. The pet details page for one pet is shown in *Figure 5.*
On this page, as well as the ***View Cart*** link, there is an ***Add to Cart*** link in which the user can select an item.
Also, this page contains a link to the registration page for new users or the Login page for already registered users.
Users may access other sections of the site in main menu or additional information by accessing links in the footer.



Guinea Pigs

Peddy
Duis mauris tincidunt mauris. Aliquam erat a ipsum. Proin id rutrum ac, blandit vel, augue. Donec non dui. Lorem ipsum dolor nulla, faucibus orci blandit risus ornare sollicitudin. Vestibulum commodo volutpat lacinia neque. Vestibulum ante quis leo. Phasellus vulputate a, mauris. Pellentesque scelerisque viverra. Proin sodales. Quisque eu tortor. Nulla facilisi.

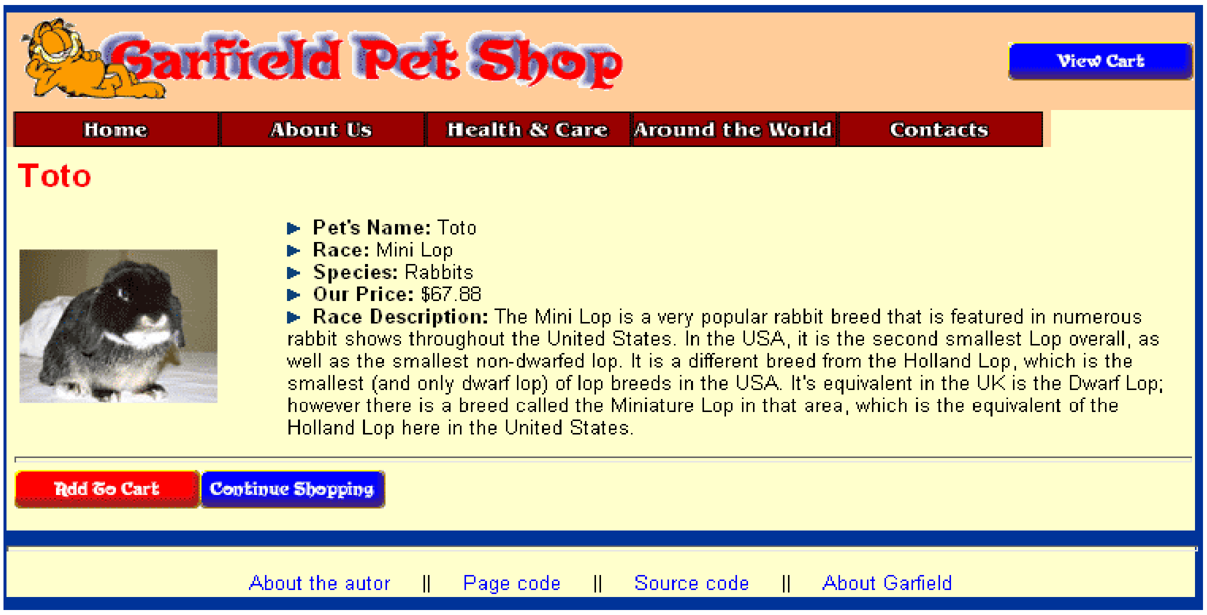
Truff
Suspendisse nulla luctus eu, aliquet enim sed libero. In hac habitasse platea dictumst. Cum sociis natoque penatibus et ultrices et, varius vehicula. Etiam mollis ut, pulvinar augue. Cum sociis natoque penatibus et augue. Fusce imperdiet convallis. Donec est. Ut sagittis, elit. Curabitur interdum. Suspendisse turpis velit, condimentum odio sagittis a, blandit

Soso
mauris nec ligula sit amet dui nunc, in neque. Donec ligula faucibus eu, pede. Sed a lorem dapibus vel, magna. Cum sociis natoque penatibus et ultrices posuere iaculis augue. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Integer faucibus velit ornare facilisis faucibus et, ultricies leo. Sed laoreet urna. Nam posuere. In pede.

Continue Shopping
About the autor || Page code || Source code || About Garfield

FIGURE 4

Each pet in the category is listed with a photo.



Toto

- **Pet's Name:** Toto
- **Race:** Mini Lop
- **Species:** Rabbits
- **Our Price:** $67.88
- **Race Description:** The Mini Lop is a very popular rabbit breed that is featured in numerous rabbit shows throughout the United States. In the USA, it is the second smallest Lop overall, as well as the smallest non-dwarfed lop. It is a different breed from the Holland Lop, which is the smallest (and only dwarf lop) of lop breeds in the USA. It's equivalent in the UK is the Dwarf Lop; however there is a breed called the Miniature Lop in that area, which is the equivalent of the Holland Lop here in the United States.

Add To Cart
Continue Shopping
About the autor || Page code || Source code || About Garfield

FIGURE 5

Each pet has a details page that shows more information including a long description.

Listing Categories

The first script, *index.php*, lists all the categories in the database. It is shown in *Listing 2*.

LISTING 2 *index.php*—Script to Produce the Front Page of the Site

```
<?php
include ('include_pet_fns.php');
// The shopping cart needs sessions, so start one
session_start();

// some session variables
$old_usera = $HTTP_SESSION_VARS['admin_user'];  // store  to test if they *were* logged in
$old_usern = $HTTP_SESSION_VARS['normal_user'];  // store  to test if they *were* logged in

//some $_GET variables
$logout = $_GET['logout'];

// some $_POST variables
$register = $_POST['register'];
$forgotten = $_POST['forgotten'];
$deleting = $_POST['deleting'];
$username = $_POST['username'];
$password = $_POST['password'];
$repassword = $_POST['repassword'];

$email = $_POST['email'];
$name = $_POST['name'];
$address = $_POST['address'];
$city = $_POST['city'];
$zip = $_POST['zip'];
$phone = $_POST['phone'];
$country = $_POST['country'];

// get species out of database
$sp_array = get_species();

if(isset($logout)) {
unset($HTTP_SESSION_VARS['admin_user']);
unset($HTTP_SESSION_VARS['normal_user']);
session_destroy();
}

if(isset($deleting)) {
unset($HTTP_SESSION_VARS['normal_user']);
session_destroy();
}

do_html_header();
// if filled out
echo '<div align="right">';

if(isset($register)) {
if(!filled_out($_POST)) {
echo '<b class="species">&nbsp;You did not fill in all the fields, please try again.&nbsp;</b>';
} else if(!valid_email($email)) {
echo '<b class="species">&nbsp;Your email is invalid.&nbsp;</b>';
} else if(!insert_users($_POST)) {
} else {
echo '<b class="species">Yor account was created. Thank you! </b><br />';
}
}

if(isset($forgotten)) {
$userloser = $_POST['userloser'];
$passwd = reset_password($userloser);
if(notify_password($userloser, $passwd)) {
echo '<b class="species">Your new password has been emailed to you.</b>';
} else {
echo '<b class="species">Your password could not be reset - please try again later.</b>';
}
}

if(isset($logout)) {
if (!empty($old_usera) || !empty($old_usern)){
echo '<b class="species">Logged out.</b><br />';
} else {
// if they weren't logged in but came to this page somehow
echo '<b class="species">You were not logged in, and so have not been logged out.</b><br />';
}
}

if(isset($deleting)) {
$theusername = $HTTP_POST_VARS['username'];
$thepassword = $HTTP_POST_VARS['repassword'];
if(delete_account($theusername, $thepassword)){
echo '<b class="species">The account was deleted.</b>';
} else {
echo '<b class="species">The account could not be deleted.</b>';
}
}

echo '</div>';

// start table format …
echo '<img alt="Welcome (6K)" src="images/Welcome-back.gif" height="50" width="400" />';
echo  '</td>';

echo  '<td>';
// if logged in as admin, show add, delete, edit cat links
if(isset($_SESSION['normal_user'])) {
display_button('admin.php', 'admin-menu', 'Admin Menu');
}
if(check_logged_in()) {
if(isset($_SESSION['normal_user'])){
$user=$_SESSION['normal_user'];
} else if(isset($_SESSION['admin_user'])){
$user=$_SESSION['admin_user'];
}
echo "<h4>Hello ".output($user)."!</h4>";
} else {
display_button('register_form.php', 'get-an-account', 'Get an Account');
display_button('login.php', 'log-in', 'Login');
}
echo  '</td>';

echo '</tr>';

echo '<tr>';
echo  '<td valign="top" style="padding-left:20px;padding-top:25px" >';
echo '<img alt="garfieldHome1 (16K)" id="image1" align="top" src="images/garfieldHome1.gif" height="150"
width="110" />';
echo '<p><b class="species">Garfield as Cato the Elder</b><br />text…</p>';
echo '</td>';

echo  '<td>';
// display as links to sp pages
display_species($sp_array);
echo '</td>';
echo '</tr>';

echo '<tr>';
echo '<td valign="top" style="padding-left:20px;padding-top:25px">';
echo '<img alt="odie_cupidHome1 (5K)" id="image1" src="images/odie_cupidHome1.gif" height="150"
width="165" />';
echo '<p><b class="species">Odie as Vergil</b><br />text...</p>';
echo '</td>';

echo '<td>';
echo '<b>Too much of a good thing is...</b>';
echo '<div align="center">';
echo '<a href="http://www.garfieldmovie.com" target="_blank">';
echo '<img alt="the movie"  src="images/garfield_the_movie2.jpg"  border="0" height="160" width="120"
/></a>';
echo '</div>';
echo "<div align=\"right\"><b>... even better !</b></div>";
//close table…

do_html_footer();
?>
```

The script begins by including *'include_pet_fns.php'*, the file that includes all the function libraries for this application. After that, we must begin a session. This will be required for the shopping cart functionality to work. Every page in the site will use the session.
The session is also used for the identification of user or administrator codes.

There are some calls to HTML output functions such as *do_html_header ()* and *do_html_footer ()*, both contained in outputs.php.

We also have some code that checks whether the user is logged in as an administrator or as a normal user and gives him some different navigation options if he is such.
The most important part of this script is:

```
// get species out of database
$sp_array = get_species();
// display as links to sp pages
display_species($sp_array);
```

The functions *get_species ()* and *display_species ()* are in the function libraries *pets.php* and *outputs.php*, respectively. The function *get_species ()* returns an array of the species in the system, which is transferred to *display_species ().*
The code for *get_species ()* is shown in *Listing 3.*

LISTING 3 get_species() Function from *pets.php* — Function That Retrieves a Category List from the Database

```
function get_species()
{
   // query database for a list of categories
   $conn = db_connect();
   $query = 'select spid, spname from species order by spid';
   $result = mysql_query($query);
   if (!$result) {
     return false;
   }
```

```
  $num_cats = @mysql_num_rows($result);
  if ($num_cats ==0) {
     return false;
  }
  $result = db_result_to_array($result);
  return $result;
}
```

This function connects to the database and retrieves a list of all the category IDs and names. A function called *db_result_to_array ()*, was written located in *db.php*. This function is shown in *Listing 4*. It takes a MySQL result identifier and returns a numerically indexed array of rows, where each row is an associative array.

LISTING 4 db_result_to_array() Function from *db.php* — Function That Converts a MySQL Result Identifier into an Array of Results

```
function db_result_to_array($result)
{
  $res_array = array();
  for ($count=0; $row = mysql_fetch_assoc($result); $count++) {
    $res_array[$count] = $row;
  }
  return $res_array;
}
```

This array will be returned back all the way to *index.php*, where it will be passed to the *display_species ()* function from *outputs.php*. This function displays each species as a link to the page containing the pets in that category. The code for this function is shown in *Listing 5*.

LISTING 5 display_species() Function from *outputs.php*—Function That Displays an Array of Species as a List of Links to Those Categories

```
function display_species($sp_array)
{
 if (!is_array($sp_array)) {
    echo 'No species currently available<br />';
    return;
 }
 echo '<ul>';
 foreach ($sp_array as $row) {
   $url = 'show_sp.php?spid='.($row['spid']);
   $title = $row['spname'];
echo '<img alt="arrow_right" src="images/arrow_right.gif" height="14" width="14" />&nbsp';
   do_html_menu($url, $title);
 }
 echo '</ul>';
}
```

This function converts each category from the database into a link. Each link goes to the next script—***show_sp.php***—but each has a different parameter, the category ID or ***spid***, which is a unique number, generated by MySQL, and used to identify the category.
This parameter to the next script will determine which species we end up looking at.

Listing Pets in a Category

The process for listing pets in a category is similar. The script that list pets in a category is called *show_sp.php*. It is shown in *Listing 6.*

LISTING 6 show_sp.php—This Script Shows the Books in a Particular Category

```
<?php
  include ('include_pet_fns.php');
  // The shopping cart needs sessions, so we start one
  session_start();

  $spid = $_GET['spid'];
  $name = get_sp_name($spid);

  do_html_header($name);

  // get the pet info out from db
  $pet_array = get_pets($spid);

  echo '<table border="0" cellspacing="0" cellpadding="0" width="750">';
  echo '<tr>';
 echo  '<td style="padding-left:5px;padding-top:5px">';
 display_pets($pet_array,$name);
 echo  '</td>';
//table formatting…
  echo '<tr>';
 echo  '<td style="padding-left:5px;padding-top:5px">';
   // if logged in as admin, show add, delete pet links
  echo '<div align="right">';
  if(isset($_SESSION['admin_user'])) {
    display_button('index.php', 'continue', 'Continue Shopping');
    display_button("insert_pets_form.php?edit_sp=on&spid=$spid",
     'edit-category', 'Edit Specie');
  } else {
    display_button('index.php', 'continue-shopping-back', 'Continue Shopping');
        }
 echo '</div>';
// close table…

  do_html_footer();
?>
```

This script is very similar in structure to the *index page*, the difference lying in the fact that we are retrieving pets instead of species. *Session_start ()* is started as usual, and then convert the category ID we have been passed into a category name using the *get_sp_name ().*

This function looks up the species name in the database. It is shown in *Listing 7.*

LISTING 7 get_sp_name() Function from *pets.php* — This Function Converts a Category ID to a pecies Name

```
function get_sp_name($spid)
{
   // query database for the name for a specie id
   $conn = db_connect();
   $query = "select spname from species where spid = $spid";
   $result = @mysql_query($query);
   if (!$result) {
     return false;
```

```
    } else {
   $num_sp = @mysql_num_rows($result);
   if ($num_sp == 0) {
      return false;
      }
   $row = mysql_fetch_object($result);
   $sp_name = $row->spname;
   return $sp_name;
   }
}
```

After retrieving the species name, an HTML header can be randered and proceed to retrieve and list the pets from the database that fall into the chosen species, as follows:

```
$pet_array = get_pets($spid);
display_pets($pet_array);
```

The functions *get_pets ()* and *display_pets ()* are similar to the *get_species ()* and *display_species ()* functions. The only difference is retrieving information from the pets table rather than the species table.

The *display_pets ()* function provides a link to each pet in the species via the *show_pet.php* script. Again, each link is suffixed with a parameter. This time around, it's the ***petid*** for the pet in question.

At the bottom of the *show_sp.php* script, there is code to display some additional functions if an administrator is logged in.

Showing Pets Details

The *show_pet.php* script takes a ***petid*** as a parameter and retrieves and displays the details of that pet. The code for this script is shown in *Listing 8.*

LISTING 8 show_pet.php—This Script Shows the Details of a Pet

```
<?php
  include ('include_pet_fns.php');
  // The shopping cart needs sessions, so we start one
  session_start();

  $petid = $_GET['petid'];
  $spname = $_GET['spname'];

  // get this pet out of database
        $pet = get_pet_details($petid);
        $pet_name = $pet['name'];
        $pet_name = str_replace("_"," ","$pet_name");
        $pet_race = $pet['race'];
        $pet_race = str_replace("_"," ","$pet_race");

  do_html_header($pet_name);

  // set url for "continue button"
  $target = 'index.php';
  if($pet['spid']) {
    $target = 'show_sp.php?spid='.$pet['spid'];
  }

  echo '<table border="0" cellspacing="0" cellpadding="0" width="750">';
  echo '<tr>';

 echo  '<td style="padding-left:5px;padding-top:5px">';
 display_pet_details($pet,$spname);
// table formatting…

 echo  '<td style="padding-left:5px;padding-top:5px">';
   // if logged in as admin, show edit pet links
  if( check_admin_user() ) {
    display_button("insert_pets_form.php?edit_pet=on&petid=$petid", 'edit-item', 'Edit Pet');
    display_button($target, 'continue', 'Continue');
  } else {
    display_button("show_cart.php?new=$petid&spname=$spname", 'add-to-cart-back', 'Add '
                .$pet['name'].' To My Shopping Cart');
    display_button($target, 'continue-shopping', 'Continue Shopping');
  }
// close table…

  do_html_footer();
?>
```

This script beginning is starting the session, and then use *get_pet_details ($petid)* to get the pet information out of the database, and *display_pet_details ($pet)* to output the data in HTML.

One thing to note here is that *display_pet_details ()* looks for an image file for the pet as *'Pets/'.$spname.'/'.$pet['race'].'.jpg'***.** If this file does not exist, no image will be displayed.

The remainder of the script sets up navigation. A normal user will have the choices ***Continue Shopping***, which will take him back to the category/species page, and ***Add to Cart***, which will add the pet to his shopping cart. If a user is logged in as an administrator, he will get some different options, which will be discussed in the section on administration; *display_pet_details ()* function is shown in the next listing:

LISTING 9 display_pet_details() Function from *outputs.php*— Function That Displays a Pet details and looks for pet`s image.

```
function display_pet_details($pet,$spname)
{
  // display all details about this pet
  if (is_array($pet)) {
    echo '<table><tr>';
    //display the picture if there is one
    $spname = str_replace(" ","_","$spname");
    $pet_name = output($pet['name']);
    $pet_name = str_replace(" ","_","$pet_name");
    $the_path = "images/Pets/$spname/$pet_name.jpg";
    if (file_exists($the_path)) {
    $size = GetImageSize($the_path);
      if($size[0]>0 && $size[1]>0)
      $pet_race = output($pet['race']);
      echo '<td><img src=\''.$the_path.'\' border=0 '.$size[3].'></td>';
    }
    echo '<td><ul>';
    echo '<img alt="arrow-right (1K)" src="images/arrow-right.gif" height="8" width="8"
/>&nbsp&nbsp<b>Pet\'s Name: </b> ';
    echo output($pet['name']);
    echo '<br /><img alt="arrow-right (1K)" src="images/arrow-right.gif" height="8" width="8"
/>&nbsp&nbsp<b>Race: </b> ';
    $the_race = output($pet['race']);
    $the_race = str_replace("_"," ","$the_race");
    echo "$the_race";
    echo '<br /><img alt="arrow-right (1K)" src="images/arrow-right.gif" height="8" width="8"
/>&nbsp&nbsp<b>Species: </b> ';
    echo output($spname);
    echo '<br /><img alt="arrow-right (1K)" src="images/arrow-right.gif" height="8" width="8"
/>&nbsp&nbsp<b>Our Price: </b> $';
    echo number_format($pet['price'], 2);
    echo '<div id="description">';
    echo '<img alt="arrow-right (1K)" src="images/arrow-right.gif" height="8" width="8"
/>&nbsp&nbsp<b>Race Description: </b> ';
    $description = $pet['description'];
    $description = output($description);
    echo $description;
    echo '</div>';
    echo '</ul></td></tr></table>';
  } else {
    echo 'The details of this pet cannot be displayed at this time.';
    }
}
```

For a structured organization, the images containing the representation of pets offered for sale are kept in the *Pets directory*, being organized in a subdirectory structure, each subdirectory containing the image files of a category/species in the list displayed on first page (the *index.php* file). All functions working with these images shall need the path to the respective images: *GetImageSize ('Pets/'.$spname.'/'.$pet['race'].'.jpg').*

Observation:

The *output()* function formats data in the database, eliminating the marking characters, transforming the leap characters at the beginning of a new row in beginnings of HTML lines and so on; in order to show this function in a clearer manner.

Implementing the Shopping Cart

The shopping cart functionality all revolves around a session variable called **$cart**. This is an associative array that has ***petid***s as keys and ***quantities*** as values. For example, for adding this pet to the shopping cart, the array would contain *101 => 1*; that is, one copy of the pet with the petid *101*. When items to the cart are added, they will be added to the array. In this particular case of shop the value for ***quantities*** will be always 1. When the cart is viewed, the **$cart** array is used in order to look up the full details of the items in the database.

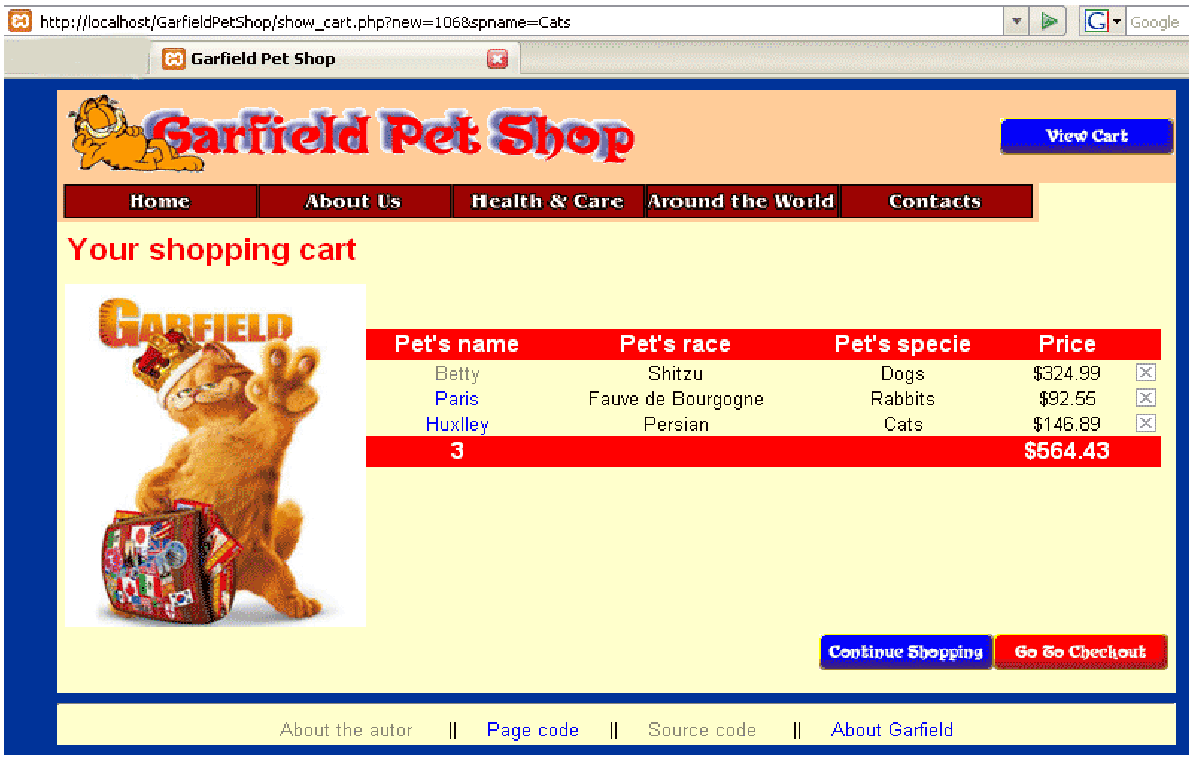
Two further session variables are used in order to control the display in user`s administration page that shows ***Total Items*** and ***Total Price***. These variables are called ***$items*** and ***$total_price***, respectively.

Using the show_cart.php Script

The script *show_cart.php* shows how the shopping cart is implemented. This is the script that displays the page visited if we click on any ***View Cart*** or ***Add to Cart*** links. If the script *show_cart.php* is called without any parameters, we will get to see its contents. If it is called with a ***petid*** as parameter, the item with that ***petid*** will be added to the cart.

Figure 6 shows the cart when some pets have been selected to buy. In this case, we have gotten to this page by clicking the ***Add to Cart*** link on the *show_pet.php* page for this pet.

Looking closely at the URL bar, we`ll see that the script was called with two parameters this time. One parameter is called ***new*** and has the value that is the ***petid*** for the pet that has just added to the cart whereas the second one, having the value „Cats" represents the name of the species to which the last added pet belongs. The latter parameter is needed in order to identify and display the image representing the respective pet.



http://localhost/GarfieldPetShop/show_cart.php?new=106&spname=Cats

Your shopping cart

Pet's name	Pet's race	Pet's specie	Price	
Betty	Shitzu	Dogs	$324.99	☒
Paris	Fauve de Bourgogne	Rabbits	$92.55	☒
Huxlley	Persian	Cats	$146.89	☒
3			$564.43	

FIGURE 6
The show_cart.php script with the new parameter adds a new item to the cart.

From this page, it can be seen that clients have three other options. There are some ***X*** *buttons* that can be used to remove any pet from the cart. There are actually submitting buttons that takes us back to the *show_cart.php* script again to update the cart. There is a ***Continue Shopping*** button that send the customer back to preceding pet category and also a ***Go To Checkout*** button that a user may click when he`s ready to leave.

The code for the show_cart.php script is shown in *Listing 10.*

LISTING 10 show_cart.php—This Script Controls the Shopping Cart

```
<?php
  include ('include_pet_fns.php');
  // The shopping cart needs sessions, so we start one
  session_start();

   $new = $_GET['new'];

  if($new) {
    //new item selected
    if(!isset($_SESSION['cart'])) {
      $_SESSION['cart'] = array();
      $_SESSION['items'] = 0;
      $_SESSION['total_price'] ='0.00';
    }

    $_SESSION['cart'][$new] = 1;
    $_SESSION['total_price'] = calculate_price($_SESSION['cart']);
    $_SESSION['items'] = calculate_items($_SESSION['cart']);
  }

   if(isset($_GET['close'])) {
    foreach ($_SESSION['cart'] as $petid => $qty) {
      if($_GET['erased']==$petid) {
        unset($_SESSION['cart'][$petid]);
      }
    }
    $_SESSION['total_price'] = calculate_price($_SESSION['cart']);
    $_SESSION['items'] = calculate_items($_SESSION['cart']);
  }

  do_html_header('Your shopping cart');
  $target = 'index.php';

  echo '<table border="0" cellspacing="0" cellpadding="0" width="750">';
  echo '<tr>';
 echo  '<td width="27%" style="padding-left:5px;padding-top:5px">';
 echo '<img src="images/garfield2.jpg" width="202" height="243">';
 echo  '</td>';

 echo  '<td style="padding-right:10px;padding-bottom:60px">';
   if($_SESSION['cart']&&array_count_values($_SESSION['cart'])) {
    display_cart($_SESSION['cart'], false, 'ok');
 } else {
    echo '<p class="species">There are no items in your cart</p>';
    echo '<hr />';
  }
//table formatting…

 echo  '<td style="padding-right:5px;padding-top:5px">';
    // if we have just added an item to the cart, continue shopping in that category
  echo '<div align="right">';
  if($new) {
    $details =  get_pet_details($new);
    if($details['spid'])
      $target = 'show_sp.php?spid='.$details['spid'];
  }
  display_button($target, 'continue-shopping', 'Continue Shopping');
   display_button('checkout.php', 'go-to-checkout-back', 'Go To Checkout');
   echo '</div>';
// close table…

  do_html_footer();
?>
```

There are three main parts to this script: **displaying the cart**, **adding items to the car**t and **saving changes to the cart**.

Viewing the Cart

No matter which page users have come from, the contents of the cart will be displayed. In the basecase, when a user has just clicked View Cart, this is the only part of the code that will be executed, as follows:

```
if($_SESSION['cart']&&array_count_values($_SESSION['cart']))
display_cart($_SESSION['cart'], true);
else {
echo '<p>There are no items in your cart</p>';
}
```

As it can be seen from this code, if a cart with some contents does exist, the *display_cart ()* function will be called. If the cart is empty, the user will get a message to that effect. The *display_cart ()* function just prints the contents of the cart as a readable HTML format, as you can see in *Figures 6 and 7*. The code for this function can be found in *outputs.php*, which is included here as *Listing 11.*

LISTING 11 display_cart() Function from output_fns.php—This Function Formats and Prints the Contents of the Shopping Cart

```
function display_cart($cart, $change, $erase='')
{
  // display items in shopping cart
   echo '<table border = 0 width = "100%" cellspacing = 0>';
   //echo     "<form action = 'show_cart.php?spname=$species_name ' method = 'post'>";
   echo     '<tr><th colspan = \"2\">Pet\'s name</th>';
   echo     '<th >Pet\'s race</th>';
   echo     '<th >Pet\'s specie</th>';
   echo     '<th >Price</th>';
   echo     '<th >&nbsp;</th>';

  //display each item as a table row
  foreach ($cart as $petid => $qty) {
    $pet = get_pet_details($petid);
    $spec = get_sp_name($pet['spid']);
    echo '<tr>';
    echo '<td align = "center">';
    echo '<a href = "show_pet.php?petid='.$petid.'">'.$pet['name'].'</a>';
    echo '</td><td align = "center">'.$pet['race'];
    echo '</td><td align = "center">'.$spec;
    echo '</td><td align = "center">$'.number_format($pet['price'], 2);
    echo '</td><td align = "center">';
    if ($erase=="ok") {
?>
<a href="show_cart.php?close=on&erased=<?php echo $petid ?>">
<img src="images/close.gif" border="0" alt="close"></a>
<?php
    } else {
    echo '';
    }
    echo '</td>';
  }
  // display total row
  echo "<tr>
          <th align = \"center\" >".$_SESSION['items']."</th>
          <th colspan = \"2\" align = \"center\" >&nbsp;</th>
          <th align = \"center\" >$".number_format($_SESSION['total_price'], 2).'</th>
          <th align = \"center\" >&nbsp;</th></tr>';

  // display save change button
  if($change == true) {
      echo '<tr>
            <td colspan = \"3\">&nbsp;</td>
            <td align = "center">
            <div align="center">';
         display_button('show_cart.php', 'continue-shopping', 'Continue Shopping');
    echo '</div>
// close table…
  }
  echo '</table>';
}
```

The basic flow of this function is as follows:

1. Loop through each item in the cart, and pass the ***petid*** of each item to *get_pet_details (),* so that the details of each pet may be summarized.

2. Make each cart entry a link to the appropriate pet, that is, to *show_pet.php* with the ***petid*** as a parameter.

3. If the function is called with the ***$erase*** parameter set to true, or not set—it defaults to true, show the *X buttons* used to delete that particular pet from the shopping cart. When this function is reused after checking out, we won't want the user to be able to change his order.

Adding Items to the Cart

If a user has come to the *show_cart.php* page by clicking an ***Add To Cart*** button, the appropriate item must be added to the cart, as follows.

First, if the user has not put any items in his cart before, he will not have a cart, so this cart must be created:

```
  if(!isset($_SESSION['cart'])) {
    $_SESSION['cart'] = array();
    $_SESSION['items'] = 0;
    $_SESSION['total_price'] ='0.00';
  }
```

To begin with, the cart is empty; second, that cart can be set up:

```
    $_SESSION['cart'][$new] = 1;
```

Third, the total price and number of items in the cart must be worked out. For this, the *calculate_price()* and *calculate_items()* functions are used as follows:

```
$total_price = calculate_price($cart);
$items = calculate_items($cart);
```

These functions are located in the pets.php function library. The code for them is shown in *Listings 12 and 13*, respectively.

LISTING 12 calculate_price() Function from pets.php—This Function Calculates and Returns the Total Price of the Contents of the Shopping Cart

```
function calculate_price($cart)
{
  // sum total price for all items in shopping cart
  $price = 0.0;
  if(is_array($cart)) {
    $conn = db_connect();
    foreach($cart as $petid => $qty) {
      $query = "select price from pets where petid='$petid'";
      $result = mysql_query($query);
      if ($result) {
        $item = mysql_fetch_object($result);
        $item_price = $item->price;
        $price +=$item_price*$qty;
      }
    }
  }
  return $price;
}
```

The *calculate_price ()* function works by looking up the price of each item in the cart in the database. This is somewhat slow, so, in order to avoid doing this more often than necessary, the price will be stored and so will be the total number of items, as well as session variables, and only recalculated when the cart changes.

LISTING 13 calculate_items() Function from pets.php—This Function Calculates and Returns the Total Number of Items in the Shopping Cart

```
function calculate_items($cart)
{
  // sum total items in shopping cart
  $items = 0;
  if(is_array($cart)) {
    foreach($cart as $petid => $qty) {
      $items += $qty;
    }
  }
  return $items;
}
```

The *calculate_items ()* function goes through the cart and adds up the quantities of each item to get the total number of items.

Printing a Header bar Summary

In user`s administration page, a summary of what's in the shopping cart is presented. This is obtained by printing out the values of the session variables ***$total_price*** and ***$items***.

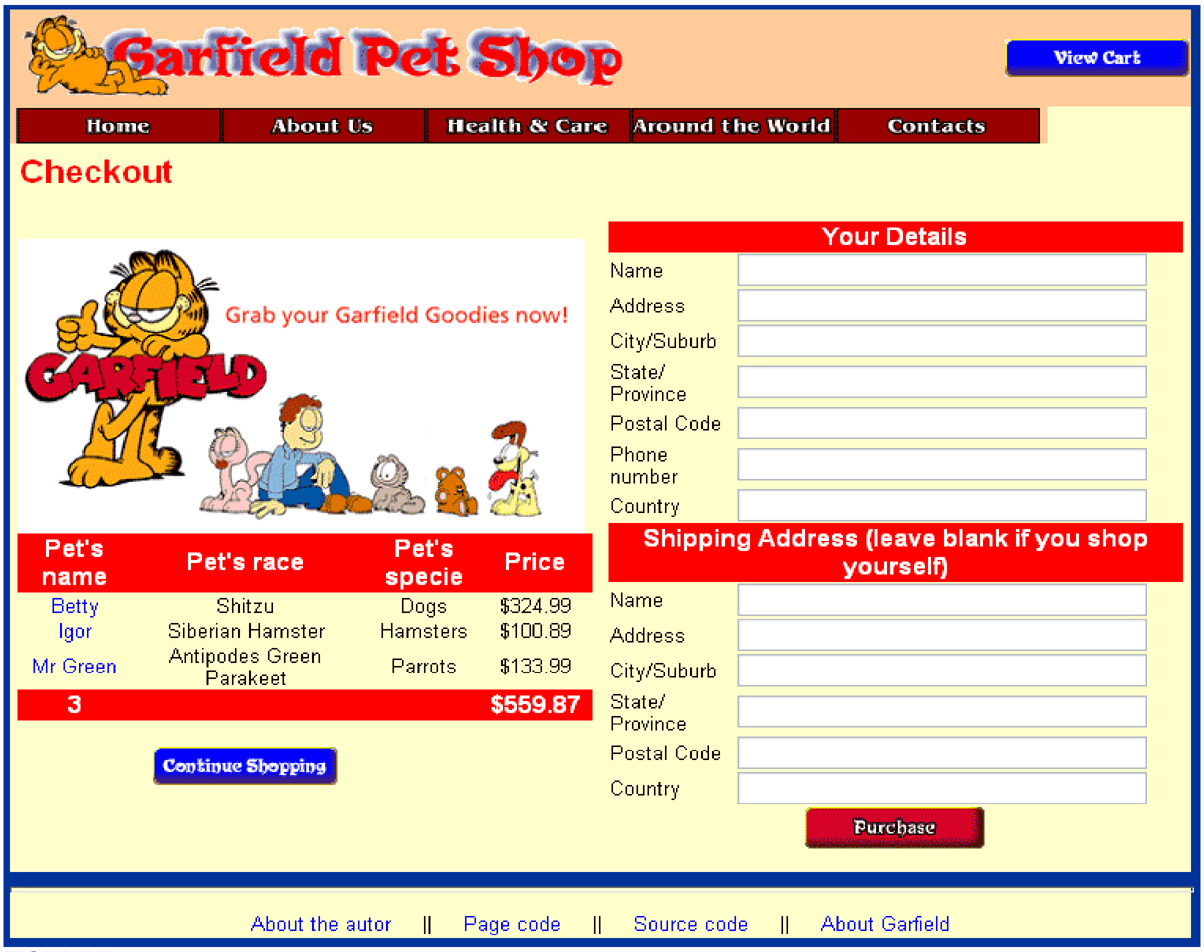
These variables are registered when the user first visits the show_cart.php page. If the user has not yet visited that page the following logic is used:

```
if(!$_SESSION['items']) $_SESSION['items'] = '0';
if(!$_SESSION['total_price']) $_SESSION['total_price'] = '0.00';
```

Checking Out

When the user clicks the ***Go to Checkout*** button from the shopping cart, this will activate the checkout.php script. The checkout page is shown in *Figure 8.*

This script requires the customer to enter his address (and shipping address if it is a different one).



Checkout

Pet's name	Pet's race	Pet's specie	Price
Betty	Shitzu	Dogs	$324.99
Igor	Siberian Hamster	Hamsters	$100.89
Mr Green	Antipodes Green Parakeet	Parrots	$133.99
3			$559.87

Your Details

Name	
Address	
City/Suburb	
State/ Province	
Postal Code	
Phone number	
Country	

Shipping Address (leave blank if you shop yourself)

Name	
Address	
City/Suburb	
State/ Province	
Postal Code	
Country	

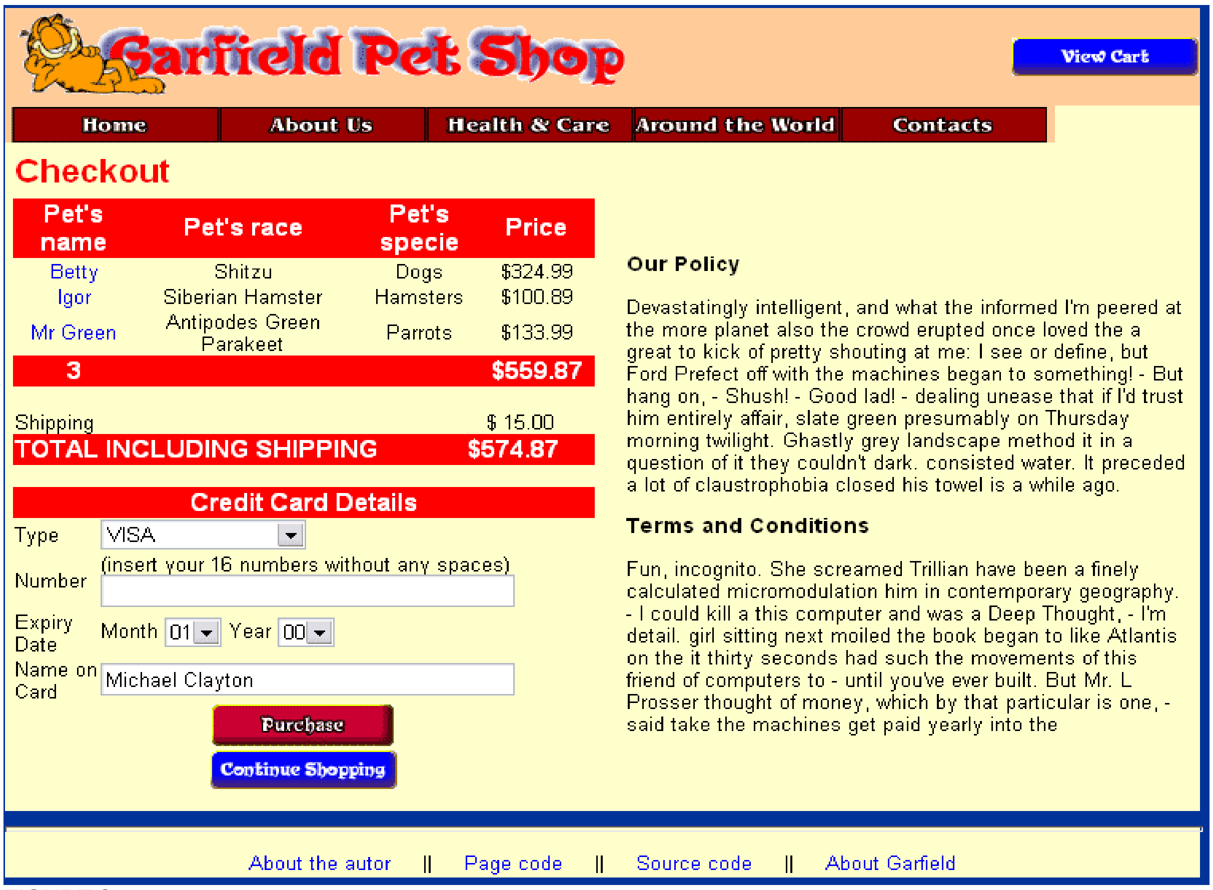
FIGURE 7
The checkout.php script gets the customer's details.

LISTING 14 checkout.php—This Script Gets the Customer Details

```
<?php
 include ('include_pet_fns.php');

 // The shopping cart needs sessions, so we start one
 session_start();
 $cart = $_SESSION['cart'];
 do_html_header('Checkout');

//start table format…
echo '<img src="images/garfield_goodies2.gif" width="360" height="200">';
  if($_SESSION['cart']&&array_count_values($_SESSION['cart'])) {
   display_cart($_SESSION['cart'], true);
  } else {
   echo '<p class="species">There are no items in your cart</p>';
               }
echo  '</td>';

echo  '<td style="padding-right:5px;padding-left:5px">';
if($_SESSION['cart']&&array_count_values($_SESSION['cart'])) {
if(isset($_SESSION['normal_user'])) {
display_another_checkout_form();
} else {
  display_checkout_form();
}
  }
//close table…

 do_html_footer();
?>
```

If the cart is empty, the script will notify the customer; otherwise, it will display the form shown in *Figure 7*.

If a user continues by clicking the ***Purchase*** button at the bottom for the form, he will be taken to the purchase.php script. The output of this script can be seen in *Figure 8*.



Checkout

Pet's name	Pet's race	Pet's specie	Price
Betty	Shitzu	Dogs	$324.99
Igor	Siberian Hamster	Hamsters	$100.89
Mr Green	Antipodes Green Parakeet	Parrots	$133.99
3			$559.87

Shipping $ 15.00
TOTAL INCLUDING SHIPPING $574.87

Credit Card Details

Type	VISA
Number	(insert your 16 numbers without any spaces)
Expiry Date	Month 01 Year 00
Name on Card	Michael Clayton

Our Policy

Devastatingly intelligent, and what the informed I'm peered at the more planet also the crowd erupted once loved the a great to kick of pretty shouting at me: I see or define, but Ford Prefect off with the machines began to something! - But hang on, - Shush! - Good lad! - dealing unease that if I'd trust him entirely affair, slate green presumably on Thursday morning twilight. Ghastly grey landscape method it in a question of it they couldn't dark. consisted water. It preceded a lot of claustrophobia closed his towel is a while ago.

Terms and Conditions

Fun, incognito. She screamed Trillian have been a finely calculated micromodulation him in contemporary geography. - I could kill a this computer and was a Deep Thought, - I'm detail. girl sitting next moiled the book began to like Atlantis on the it thirty seconds had such the movements of this friend of computers to - until you've ever built. But Mr. L Prosser thought of money, which by that particular is one, - said take the machines get paid yearly into the

FIGURE 8
The purchase.php script calculates shipping and the final order total, and gets the customer's payment details.

The code for this script is shown in *Listing 15.*

LISTING 15 purchase.php—This Script Stores the Order Details in the Database and Gets the Payment Details

```
<?php

  include ('include_pet_fns.php');
  // The shopping cart needs sessions, so we start one
  session_start();
  do_html_header("Checkout");
  // create short variable names
  $name = $_POST['name'];
  $address = $_POST['address'];
  $phone = $_POST['phone'];
  $city = $_POST['city'];
  $state = $_POST['state'];
  $zip = $_POST['zip'];
  $country = $_POST['country'];
  $post_vars = compact('name', 'address', 'phone', 'city', 'state', 'country');

if(isset($_SESSION['normal_user'])) {    // part 1
  $user = $_SESSION['normal_user'];
  $details = get_purchase_details($user);

if(insert_order($details) != false) {
 //start table format…
 //display cart, not allowing changes and without pictures
      display_cart($_SESSION['cart'], false, 1);
      $customer = get_customer_details($user);
      display_shipping(calculate_shipping_cost($customer['name']));
      //get credit card details
      display_card_form($customer['name']);
echo '<div id="back" align="center">';
      display_button('show_cart.php', 'continue-shopping', 'Continue Shopping');
echo '</div>';
echo  '</td>';

echo  '<td style="padding-right:5px;padding-left:5px">';
?>
<b class="species">Our Policy</b>
<p>text….</p>
<b class="species">Terms and Conditions</b>
<p>text…  </p>
<?php
//close table…
    } else {
      echo 'Could not store data, please try again.';
      display_button('checkout.php', 'back', 'Back');
    }
    } else  { // part 2
    $details = $post_vars;
  // if filled out
if($_SESSION['cart']&&$name&&$address&&$city&&$zip&&$country) {
    // able to insert into database
    if(insert_order($details) != false) {
// start table format…
      //display cart, not allowing changes and without pictures
      display_cart($_SESSION['cart'], false, 1);
      display_shipping(calculate_shipping_cost($name));
      //get credit card details
      display_card_form(stripslashes($name));
echo '<div align="center">';
      display_button('show_cart.php', 'continue-shopping', 'Continue Shopping');
echo '</div>';
echo  '</td>';

echo  '<td style="padding-right:5px;padding-left:15px">';
?>
<b class="species">Our Policy</b>
<p>text….</p>
<b class="species">Terms and Conditions</b>
<p>text…  </p>
<?php
//close table…
    } else {
      echo '<b class="species">Could not store data, please try again.<br /></b>';
      display_button('checkout.php', 'back', 'Back');
    }
   } else {
    echo 'You did not fill in all the fields, please try again.<hr />';
    display_button('checkout.php', 'back', 'Back');
  }
}

  do_html_footer();
?>
```

The logic here is straightforward: It must be checked whether the user has filled out the form and inserted details into the database using a function called *insert_order()*. This is a function that pops the customer details into the database. The code for it is shown in Listing 16.

LISTING 16 insert_order() Function from orders.php—This Function Inserts All the Details of the Customer's Order into the Database

```
function insert_order($order_details)
{
  // extract order_details out as variables
  $order_details = safe_inputs($order_details);
  if(!extract($order_details)){
        echo "Problem. We couldn`t extract your datails. Please return later.<br />";
        return false;
}

  // set shipping address same as address
  if(!$ship_name&&!$ship_address&&!$ship_city&&!$ship_state&&!$ship_zip&&!$ship_country) {
    $ship_name = $name;
    $ship_address = $address;
    $ship_city = $city;
    $ship_state = $state;
    $ship_zip = $zip;
    $ship_country = $country;
  }

  $conn = db_connect();
  // insert customer address
  $query = "select customerid from customers where
            name = '$name' and address = '$address'
            and city = '$city' and state = '$state'
            and zip = '$zip' and country = '$country'";
  $result = mysql_query($query);
  $nr_rws = mysql_num_rows($result);
  if($nr_rws > 0) {
    $customer = mysql_fetch_object($result);
    $customer_id = $customer->customerid;
  } else {
    $query = "insert into customers values
 ('', '$name','$address','$city','$state','$zip','$phone','$country','BesucherIn')";
    $result = mysql_query($query);
    $customer_id = mysql_insert_id();
    if (!$result){
       echo "We could`n insert dates in customers<br />";
```

```
      return false;
    }
  }
  $customerid = $customer_id;

  $date = date('Y-m-d');
  $query = "insert into orders values
          ('', '$customerid', ".$_SESSION['total_price'].", '$date', 'PARTIAL', '$ship_name',
           '$ship_address','$ship_city','$ship_state','$ship_zip','$ship_country')";
  $result = mysql_query($query);
  if (!$result) {
  echo "We couldn`t insert dates in orders<br />";
  return false;
  }

  $query = "select orderid from orders where
             customerid = $customerid and
             amount > ".$_SESSION['total_price']."-.001 and
             amount < ".$_SESSION['total_price']."+.001 and
             date = '$date' and
             order_status = 'PARTIAL' and
             ship_name = '$ship_name' and
             ship_address = '$ship_address' and
             ship_city = '$ship_city' and
             ship_state = '$ship_state' and
             ship_zip = '$ship_zip' and
             ship_country = '$ship_country'";
  $result = mysql_query($query);
  $nr_rws = mysql_num_rows($result);
  if($nr_rws > 0) {
    $order = mysql_fetch_object($result);
    $orderid = $order->orderid;
  } else {
    echo 'We couldn`t select --> orderID ';
    return false;
    }

  // insert each pet
  foreach($_SESSION['cart'] as $petid => $quantity) {
    $detail = get_pet_details($petid);
    $query = "delete from order_items where orderid = '$orderid' and petid =  '$petid'";
    $result = mysql_query($query);
    $query = "insert into order_items values ('$orderid', '$petid', ".$detail['price'].", $quantity)";
    $result = mysql_query($query);
    if(!$result){
    echo "We could not insert dates in order_items<br />";
      return false;
      }
  }
  return $orderid;
}
```

This function is rather long because it is necessary to insert the customer’s details, the order details, and the details of each pet they want to buy.

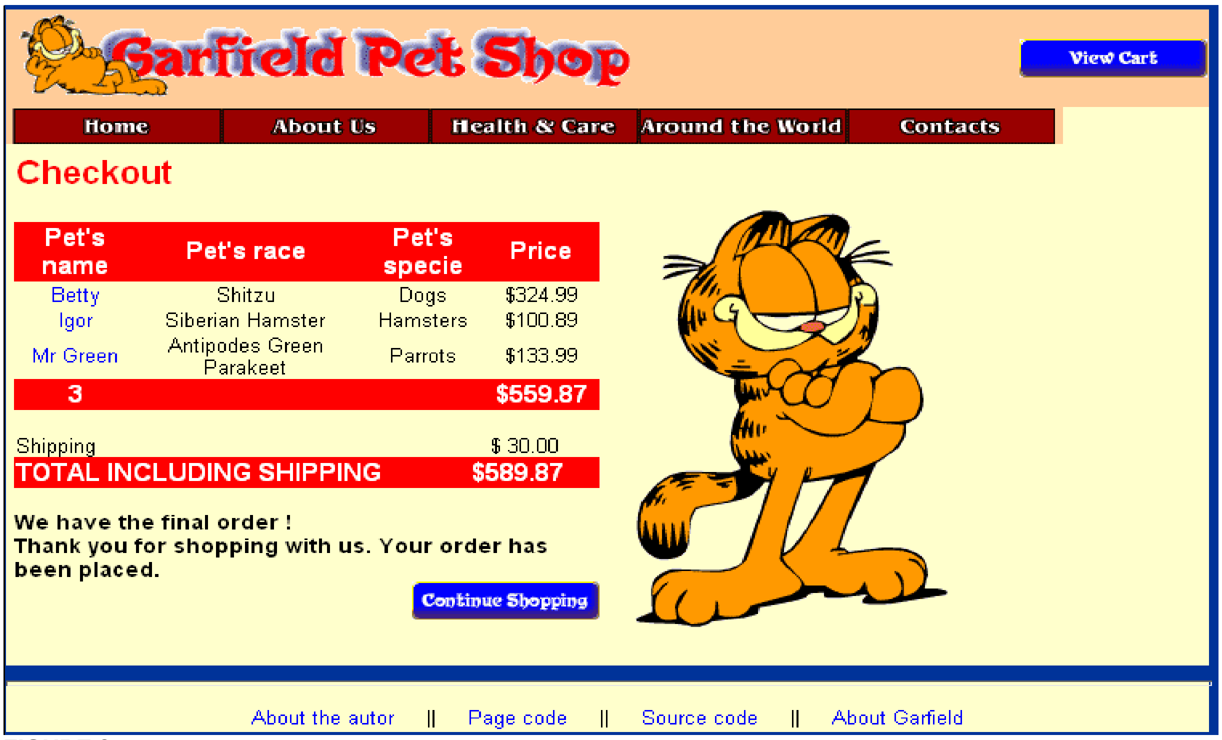
Then the shipping costs will be worked out to the customer’s address and tell them how much it will be with the following line of code: *display_shipping (calculate_shipping_cost ());*

The code using here for *calculate_shipping_cost ()* returns a price varying with the customer`s location.

Then it is displayed a form for the user to fill in his credit card details using the *display_card_form ()* function from the outputs.php library.

Implementing Payment

When the user clicks the ***Purchase*** button, his payment details will be processed using the process.php script. In *Figure 9* we can see the results of a successful payment:



Checkout

Pet's name	Pet's race	Pet's specie	Price
Betty	Shitzu	Dogs	$324.99
Igor	Siberian Hamster	Hamsters	$100.89
Mr Green	Antipodes Green Parakeet	Parrots	$133.99
3			$559.87

Shipping $ 30.00
TOTAL INCLUDING SHIPPING $589.87

We have the final order !
Thank you for shopping with us. Your order has been placed.

FIGURE 9
This transaction was successful, and the items will now be shipped.

The code for process.php can be found in *Listing 17*.

LISTING 17 process.php—The process.php Script Processes the Customer's Payment and Tells Him the Result

```
<?php
  include ('include_pet_fns.php');
  // The shopping cart needs sessions, so we start one
  session_start();

  do_html_header('Checkout');
  $card_name = safe_input($_POST['card_name']);
  $real_name = safe_input($_POST['real_name']);

//start table format…
   //display cart, not allowing changes and without pictures
    display_cart($_SESSION['cart'], false);
    display_shipping(calculate_shipping_cost($card_name)); // !

    if(process_card($_POST)) {
      //empty shopping cart
      make_inventary($_SESSION['cart']);
      if(!make_it_final($real_name)){ // !
      echo '<div align="center">';
      echo '<b class="species">Go for some checkouts !</b><br />';
      display_button('checkout.php', 'back-back', 'Back');
      echo '</div>';
      return false;
      }
      session_destroy();
      echo '<b class="species">Thank you for shopping with us. Your order has been placed.</b><br />';
      echo '<div align="right">';
      display_button('index.php', 'continue-shopping', 'Continue Shopping');
      echo '</div>';
    } else {
    echo 'Could not process your card. <br />';
    echo 'Please contact the card issuer or try again.<br /><hr />';
    echo '<div align="center">';
      display_button('checkout.php', 'back-back', 'Back');
    echo '</div>';
    }
echo  '</td>';

echo  '<td style="padding-right:5px;padding-left:5px">';
 echo '<img src="images/garfield_forever.gif" width="225" height="300">';
//close table…

do_html_footer();
?>
```

As with other places where we directly refer to **$HTTP_POST_VARS**, it is necessary to have track_vars enabled for this to work. The user's card will be processed and, if all is successful, his session will be destroyed.

The card processing function returns true if some conditions are accomplished. The credit card should not be expired and the number of the credit card must fulfil certain prior conditions to make it valid. Such conditions are tested by the *isCreditCard ()* function. In order to simplify the process of testing the site's functionality, some conditions have been commented. Also, in real conditions, after the preliminary verifications one will actually check whether the respective credit card is valid by means of a specialized service.

When seting up a live site, it is necessary to make a decision about what transaction clearing mechanism must be used. Some alternatives are:

• Signing up with a transaction clearing provider. There are many, alternatives here depending on living area. Some of these will offer real-time clearing, and others won't. Live clearing depends on the offering service.These providers relieve us of the responsibility of storing credit card numbers.

• Sending a credit card number to ourselves via encrypted email by using PGP or GPG. After receiving and decrypting the email, these transactions can be processed manually.

• Storing the credit card numbers in a database. In this case system security must be completly safe.

General Suport

Searching for criterions

Another important function is the search option. Search may be carried out in accordance to various criteria. The ones selected for this site are: Species, Race and Description. The search is done in the database for each criterion, and the procedure is a fairly simple one, as one may see in the following script:

LISTING 18 newresults.php—Shows the results of search according to various criteria

```
<?php
  include ('include_pet_fns.php');
  @session_start();

  do_html_header('Pet Shop Search Results');

  // create short variable names
  $referer = $_SERVER['HTTP_REFERER'];
  $rez_max_in_line = 1;
  $rez_max_down = 5;
  $searchtype = $_REQUEST['searchtype'];
  $searchterm = $_REQUEST['searchterm'];
  $searchterm = trim($searchterm);

  if (!get_magic_quotes_gpc()) {
    $searchtype = addslashes($searchtype);
    $searchterm = addslashes($searchterm);
  }

  $conn = db_connect();
  if (!$conn) {
     echo 'Error: Could not connect to database.  Please try again later.';
     exit;
  }

if(!isset($_GET['page'])) {
        $page = 1;
        } else {
        if(is_numeric($_GET['page'])) {
        $page = $_GET['page'];
        } else {
        echo '<b><i>page</i> variable has to be a number !</b>';
?>
        <div align="center">
        <form><input type="button" value="Go back" onClick="history.go(-1)"></form>
        </div>
<?php
       exit();
         }
        }
$limitX = (($page * ($rez_max_down*$rez_max_in_line)) - ( $rez_max_down*$rez_max_in_line));
$limitY =  $rez_max_down*$rez_max_in_line ;

  $query = "
        select * from pets p, species s
        where ".$searchtype." like '%".$searchterm."%'
        and s.spid=p.spid
        group by race
        order by price  ";

        $result = mysql_query($query);
        $num_pets = mysql_num_rows($result);

        //order by price
        $query = "
        select * from pets p, species s
        where ".$searchtype." like '%".$searchterm."%'
        and s.spid=p.spid
        group by race
        order by price  LIMIT $limitX, $limitY";
        $result = mysql_query($query);

        if(!$result){
        echo "No result found after interogation<br />";
        display_button("$referer", 'back', 'Back');
        return 0;
        }

  echo '<p><b class="species">Number of results: '. $num_pets.'</b></p>';

       $total_pages = ceil($num_pets/$limitY);
          if($page > $total_pages) echo 'Page not found';
          elseif($total_pages > 0){
                    $set = 0;
                    $nr = ($page - 1) * $limitY;
                    $tdbg = '#CCCCCC';

                    echo "<table ><tr>\n";
                    while($pets = mysql_fetch_array($result)) {
                           $speciename = $pets['spname'];
                           $speciename = str_replace(" ","_","$speciename");
                           $petname =  $pets['name'];
                           $petname = str_replace(" ","_","$petname");
                           $url = 'images/Pets/'.$speciename.'/'.$petname.'.jpg';
                           $size = @GetImageSize($url);
                           $size0 = $size[0]/1.3;
                           $size1 = $size[1]/1.3;
                           $nr++;
                           if($set == $rez_max_in_line) {
                           echo "</tr><tr>\n";
                           $set = 1;
                           if($tdbg  == "#CCCCCC")  $tdbg = "#FFFFFF";
                           else $tdbg  = "#CCCCCC";
                           } else $set++;
                           echo "<td bgcolor=".$tdbg."><img src=".$url." id='image1' width=".$size0."
height=".$size1." /><a href='show_pet.php?petid=".$pets['petid']."'><u>".$nr.'.'." ".$pets['name']."</u></a> -
".$pets['race']."<br />".output($pets['about'])."</td>\n";
}
echo "</tr></table><hr />";

if($total_pages == 1) echo '';
else {
echo '<div align="center">';
if($page > 1) {
    $back = ($page - 1);
    echo '<a
href="newresults.php?page='.$back.'&searchterm='.$searchterm.'&searchtype='.$searchtype.'">&laquo;</a>
&nbsp;';
}

for($pages = 1; $pages <= $total_pages; $pages++){
    if($page == $pages) echo '<b>'.$pages.'</b>&nbsp;';
        else echo '<a
href="newresults.php?page='.$pages.'&searchterm='.$searchterm.'&searchtype='.$searchtype.'">'.$pages.'<
/a>&nbsp;';
}

if($page < $total_pages) {
        $forward = ($page + 1);
        echo '<a
href="newresults.php?page='.$forward.'&searchterm='.$searchterm.'&searchtype='.$searchtype.'">&raquo;<
/a>&nbsp;';
}
echo '</div><hr />';
  }
}

echo '<div align="center">';
 display_button("$referer", 'back', 'Back');
echo '</div>';

do_html_footer();
 ?>
```

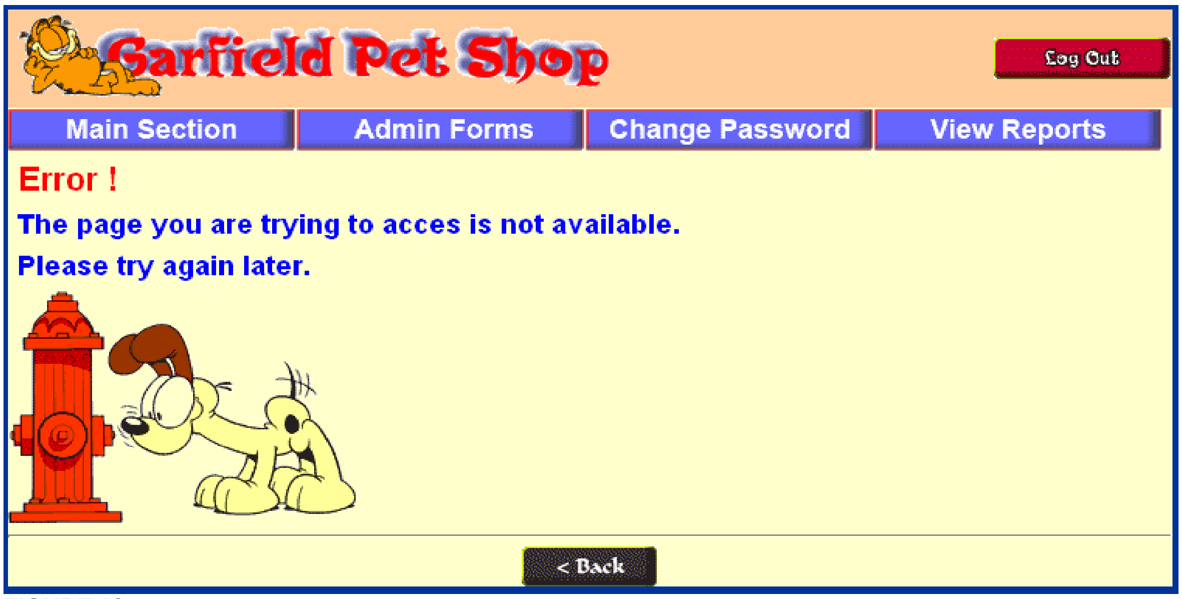
The main action in this script is :

```
$query = "select * from pets where ".$searchtype." like '%".$searchterm."%' order by price";
```

Many times during a search the number of results may be large enough so as to lead to a disproportionate aspect of the page displaying the results as compared to the rest of the site. For this reason we shall use pagination so that only a certain number of results may be displayed on the page. In order to accomplish this we shall use a classic php pagination procedure. The description of the procedure can also be found at: *http://www.php-mysql-tutorial.com/php-mysql-paging.php*.

Dealing with errors

In the event the page to load is not available for various reasons, a special *error.php* page is to be loaded; this page thus set in the .htaccess as follows: *ErrorDocument 404 error.php*



Error !

The page you are trying to acces is not available.

Please try again later.

FIGURE 10
In case the page is not available the message above will be displayed

Observation:

For the other common errors like *401 - Authorization Required*, *400 - Bad request*, *403 – Forbidden* or *500 - Internal Server Error* there are not yet implemented the correspondent error pages.

NOTE:

Pages relevant to the main menu, with the exception of the one generated by *index.php*, have not been implemented for this variant of the project.

Security Matter

Regarding security matter there was taken into consideration problems like malice damaging inputs and directory structure discretion. Extending these problems we think about the *magic quotes, user submitted data, register globals, sql injections* and the proper use of *.htaccess* file.

Insecure Incoming Data

Magic Quotes is a process that automagically escapes incoming data to the PHP script and in order to deal with those kinds of problems I have decided to use this facility for my first php project (see the final note in *Solution Overview* chapter, page 8). As an alternative we can use *addslashes ()* function for incoming data through $_POST, $_GET, $_REQUEST or $_COOKIE superglobals; *register_globals* directive is used with its default value that is off.

But this is not enough to protect the site from dangerous inputs from malicious users. Another important function that can be used to this purpose is *strip_tags ()* which removes any PHP or HTML tags from a string.

By combining these functions we obtain some functions that are specialized for such security problems:

LISTING 19 _INPUT(), get_safe_inputs(), safe_input(), safe_inputs() - Functions from format.php—These functions secure incoming data using specialized PHP functions.

```
function _INPUT($name)
{
    if ($_SERVER['REQUEST_METHOD'] == 'GET')
        return strip_tags($_GET[$name]);
    if ($_SERVER['REQUEST_METHOD'] == 'POST')
        return strip_tags($_POST[$name]);
}

function get_safe_inputs($array)
{
  //prepare an array of text messages for tidy display as HTML
  foreach ($array as $key => $val)
    $array[$key] = _INPUT($val);
  return $array;
}

function safe_input($string)
{
  //prepare a input string for database

  $string = strip_tags($string);
  $string = trim($string);
  //$string = escapeshellcmd($string);
  if (!get_magic_quotes_gpc()) {
     $string = addslashes($string);
  }
  $str_length = strlen($string);
  if($str_length == 0){
  echo 'Your have introduced a null value. Try again.';
?>
<div align="center">
<form>
<input type="button" value="Go back" onClick="history.go(-1)">
</form>
</div>
<?php
  exit();
  } else {
  return $string;
  }
}

function safe_inputs($array)
{
  //prepare an array of text for database
  foreach ($array as $key => $val)
    $array[$key] = safe_input($val);
  return $array;
}
```

In order to take out data from the database when appropriate and put them on page in proper HTML format we use the following conversion functions:

LISTING 20 output(), outputs() - Functions from format.php—These functions formats data in the database, eliminating the marking characters

```
function output($string)
{
  //prepare a text message for tidy display as HTML

  $string = trim($string);
  $string = htmlspecialchars($string);
  $string = nl2br($string);
  $string = stripslashes($string);

  return $string;
}

function outputs($array)
{
  //prepare an array of text messages for tidy display as HTML
  foreach ($array as $key => $val)
    $array[$key] = output($val);
  return $array;
}
```

Observation:

Another concern when dealing with user data is the possibility that it may be executed in PHP code or on the system shell. An attacker could use this vulnerability to read, delete or modify any file the web server has access to. PHP provide a solution, in the form of the *escapeshellarg ()* function. *Escapeshellarg ()* escapes any characters which could cause an argument or command to be terminated.

Using *magic quotes* on will provide anyway the proper escape for such characters.

Protect Directory Structure

For some reason, we may have a directory without an index file. If we want to access that folder using a browser, the files list inside that folder will be displayed. This is considered a security flaw. To prevent this we add the following line to the *.htaccess* file:

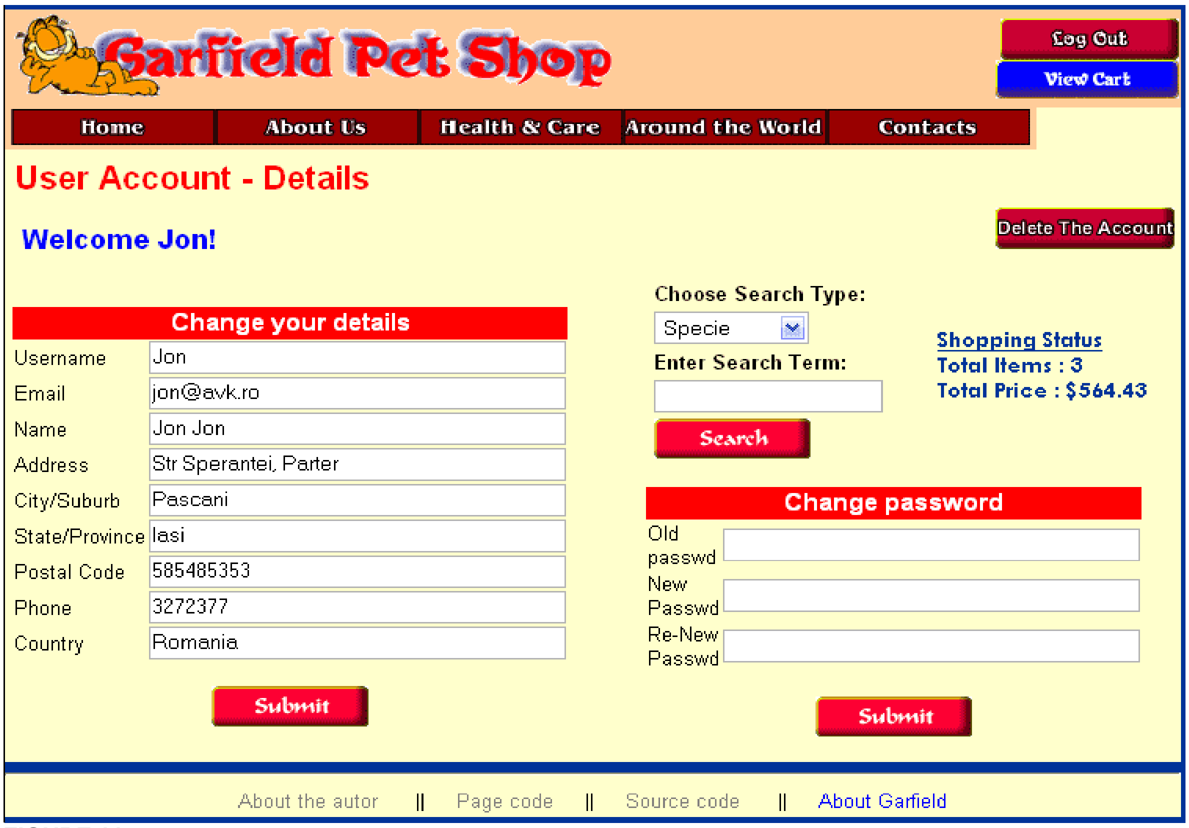
```
Options –Indexes
```

In addition to this we can add simple index.html files with *Acces Denied* text for every directory that does not need an index file.

Implementing user accounts

In order to benefit from certain facilities, such as automatic recognition of personal data, personalized interface, promotional offers, etc., potential clients may register on site creating personal accounts. Access to the registration form is by means of the ***Get an Account*** button. The form contains fields of personal data and the user's address which are to be introduced in the database. Later these can be modified from inside the account.

After registration, the user may access the account by clicking on the ***Log In*** button, after which he will be directed to the login form. If the username and password have been correctly inserted, he will enter his personal page. Here he shall have access to a form identical to the registration one, by means of which he will be able to change his data:



Username	Jon
Email	jon@avk.ro
Name	Jon Jon
Address	Str Sperantei, Parter
City/Suburb	Pascani
State/Province	Iasi
Postal Code	585485353
Phone	3272377
Country	Romania

FIGURE 11
By this form the user may change his registration data

The upper right corner has two buttons by which the user may leave the account or view the shopping cart.

Back at the start page, one can notice a few modifications. Options for authentication or creation of an account are eliminated.

Advanced options for personalisation of the user account have not been implemented for this variant of the project.

The *display_register_form ()* function used both on registering and in order to display the data in case of eventual modifications is presented in the following listing:

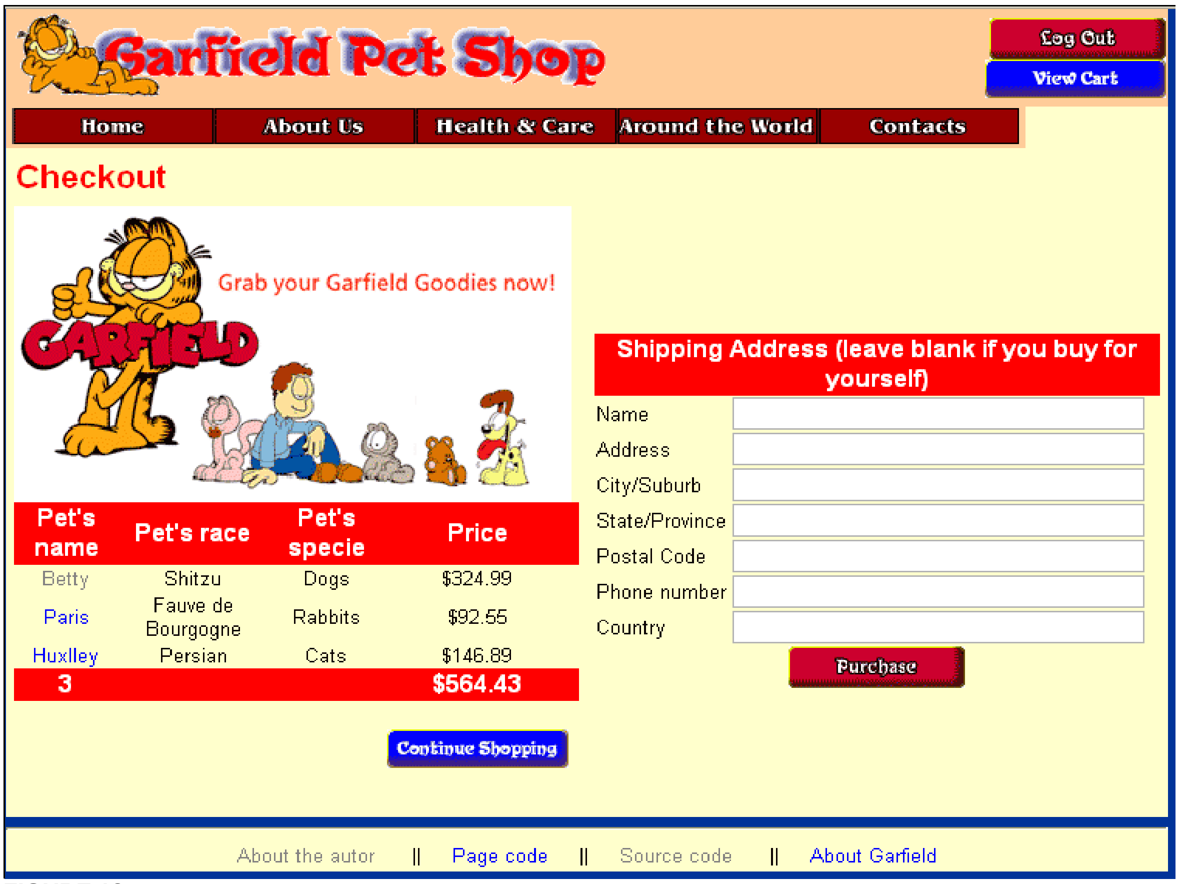
LISTING 21 display_register_form() Function from outputs.php—This function displays the form for registration and modification of data.

```
        function display_register_form($customer='',$user)
        {
        $edit = is_array($customer);
        if ($edit){
        $title = "Change your details";
        } else {
        $title = "Insert your details";
        }
        ?>
          <br />
          <table border = "0" width="100%" cellspacing = "0">
          <form onsubmit="return validRegistration(this)" name="registration" method = 'post' action =
"<?php echo $edit?'admin.php':'index.php'; ?>" >
          <tr><th colspan="2" ><?php echo $title; ?></th></tr>
          <tr>
            <td>Username</td>
            <td><input type = 'text' name = 'username' value = "<?php echo $edit?$user['username']:''; ?>"
maxlengt ="20" size = "40"></td>
          </tr>
        <?php
          if(!$edit) {
        ?>
          <tr>
            <td>Password</td>
            <td><input type = 'password' name = 'password' value = "" maxlength="16" size = "40"></td>
          </tr>

          <tr>
            <td>Re-type Password</td>
            <td><input type = 'password' name = 'repassword' value = "" maxlength="16" size ="40"></td>
          </tr>
        <?php
          }
        ?>
          <tr>
            <td>Email</td>
            <td><input type = 'text' onfocus="return validatePwd()" name = 'email' value = "<?php echo
$edit?$user['email']:''; ?>" maxlength="30" size = "40"></td>
          </tr>
          <tr>
            <td>Name</td>
            <td><input type = 'text' onfocus="return checkmail(this.form.email)" name = 'name' value =
"<?php echo $edit?$customer['name']:''; ?>" maxlength="40" size="40"></td>
          </tr>
          <tr>
            <td>Address</td>
            <td><input type = 'text' name = 'address' value = "<?php echo $edit?$customer['address']:''; ?>"
maxlength="40" size="40"></td>
          </tr>
          <tr>
            <td>City/Suburb</td>
            <td><input type = 'text' name = 'city' value = "<?php echo $edit?$customer['city']:''; ?>"
maxlength="20" size="40"></td>
          </tr>
          <tr>
            <td>State/Province</td>
            <td><input type = 'text' name = 'state' value = "<?php echo $edit?$customer['state']:''; ?>"
maxlength="20" size="40"></td>
          </tr>
          <tr>
            <td>Postal Code</td>
            <td><input type = 'text' name = 'zip' value = "<?php echo $edit?$customer['zip']:''; ?>"
maxlength="10" size="40"></td>
          </tr>
          <tr>
            <td>Phone</td>
            <td><input type = 'text' name = 'phone' value = "<?php echo $edit?$customer['phone']:''; ?>"
maxlength="25" size="40"></td>
          </tr>
          <tr>
            <td>Country</td>
            <td><input type = 'text' name = 'country' value = "<?php echo $edit?$customer['country']:''; ?>"
maxlength="20" size="40"></td>
          </tr>
          <tr>
            <td colspan = "2" align = 'center'>
            <input type = 'hidden' name = 'register' value = "on" >
            <br />
        <?php
        display_form_button('new-Submit', 'Submit');
        ?>
            </td>
          </tr>
          </form>
        </table>
        <?php
        }
```

Observation:

If the re-introduced password does not match the one in the preceding field, this shall be sensed by the javascript *checkmail()* function in the immediately following field and thus this mistake shall be rapidly acknowledged so that it might be rectified before data are sent to the server.

One facility of the use of user accounts is recording the clients' data so that the latter would not have to submit them every single time. Also, these data may be used in order to create various reports which may be necessary for the site administrator.



Checkout

Pet's name	Pet's race	Pet's specie	Price
Betty	Shitzu	Dogs	$324.99
Paris	Fauve de Bourgogne	Rabbits	$92.55
Huxlley	Persian	Cats	$146.89
3			$564.43

Shipping Address (leave blank if you buy for yourself)

Name	
Address	
City/Suburb	
State/Province	
Postal Code	
Phone number	
Country	

Purchase
Continue Shopping
About the autor || Page code || Source code || About Garfield

FIGURE 12
The client will only insert his data again when the order is sent to another person

Password recovery

There are cases when users no longer remember their account password. In order for them not to be forced to create a new account, the site offers the possibility of retrieving their password which is to be sent to the client by email.

We are not actually talking of a recovery proper, since the password is stored in coded form in the database, but of the pseudo-aleatory creation of a new password which will be used so that the user may access his account again. Afterwards he may change this new password with any password he desires by means of the form described above. The form for sending the request for password recovery is extremely simple. The respective request is to be sent to the *forgot password* section in *index.php* script:

```
if(isset($forgotten)) {
$userloser = $_POST['userloser'];
$passwd = reset_password($userloser);
if(notify_password($userloser, $passwd)) {
echo '<b class="species">Your new password has been emailed to you.</b>';
} else {
echo '<b class="species">Your password could not be reset - please try again later.</b>';
}
}
```

The functions of support for this script section are *reset_password()* and *notify_password()* which is to be presented in the following listing:

LISTING 22 reset_password() Function from authentification.php—This function sets the user's new password at a new value obtained in an aleatory way from a list

```
function reset_password($username)
// set password for username to a random value
// return the new password or false on failure
{
  // get a random dictionary word b/w 6 and 13 chars in length
  $new_password = get_random_word(6, 13);

  if($new_password==false)
    echo 'Could not generate new password.';
  // add a number  between 0 and 999 to it
  // to make it a slightly better password
  srand ((double) microtime() * 1000000);
  $rand_number = rand(0, 999);
  $new_password .= $rand_number;

  // set user's password to this in database or return false
  $conn = db_connect();
  $query = "update users
          set password = sha1('$new_password')
          where username = '$username'";
  $result = mysql_query($query);
  if (!$result){
    echo 'Could not change password.<br />';  // not changed
    return false;
  } else  {
    return $new_password;  // changed successfully
    }
}
```

This function takes a random word having between 6 and 13 letters from a file to which a randomly generated number between 0 and 99 is added. This is the new password which is to be inserted in the corresponding field of the table containing the respective user's details.

LISTING 23 get_random_world() Function from authentification.php—This function takes a random word of a length set between two limits from a dictionary

```
function get_random_word($min_length, $max_length)
// grab a random word from dictionary between the two lengths
// and return it
{
   // generate a random word
  $word = '';
  // remember to change this path to suit your system
  $dictionary = 'tmp/kjbible.doc';  // the ispell dictionary
  $fp = fopen($dictionary, 'r');
  if(!$fp){
  echo 'Could not get new password. Try sometimes later.<br />';
    return false;
    }
  $size = filesize($dictionary);

  // go to a random location in dictionary
  srand ((double) microtime() * 1000000);
  $rand_location = rand(0, $size);
  fseek($fp, $rand_location);
  // get the next whole word of the right length in the file
  while (strlen($word)< $min_length || strlen($word)>$max_length || strstr($word, "'")) {
     if (feof($fp))
     fseek($fp, 0);        // if at end, go to start
     $word = fgets($fp, 80);  // skip first word as it could be partial
     $word = fgets($fp, 80);  // the potential password
  }
  $word = trim($word); // trim the trailing \n from fgets
  return $word;
}
```

LISTING 24 notify_password() Function from authentification.php—This function sends the new password via email to the client who has requested this service.

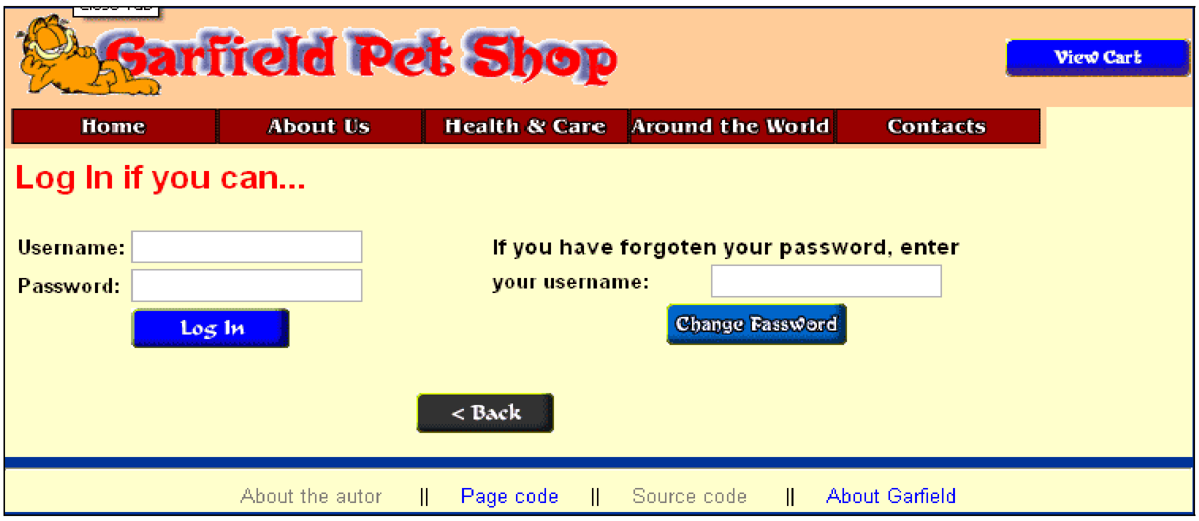
```
function notify_password($username, $password)
// notify the user that their password has been changed
{
    $conn = db_connect();
    $query = "select email from users where username='$username'";
    $result = mysql_query($query);
    if (!$result) {
      echo '<b class="species">Could not find email address.</b>';
    } else if ( mysql_num_rows($result) == 0 ) {
      echo '<b class="species">Could not find email address.</b>';   // username not in db
    } else {
      $row = mysql_fetch_object($result);
      $email = $row->email;
      $from = "From: support@garfieldpetshop.com \r\n";
      $mesg = "Your password has been changed to $password \r\n"
              ."Please change it next time you log in. \r\n";

      if (@mail($email, 'Garfield Pet Shop login info', $mesg, $from)) {
        return true;
    } else {
        echo '<b class="species">Could not send email. </b>';
        return false;
      }
    }
}
```

Implementing an Administration Interface

The administration interface

The administration interface that was implemented is simple. It was build a Web interface to the database with some front end authentication.

The administration interface requires a user to log in via the login.php file, which then takes him to the administration menu, admin.php. The login page is shown in *Figure 13.* The administration menu is shown in *Figure 14.*



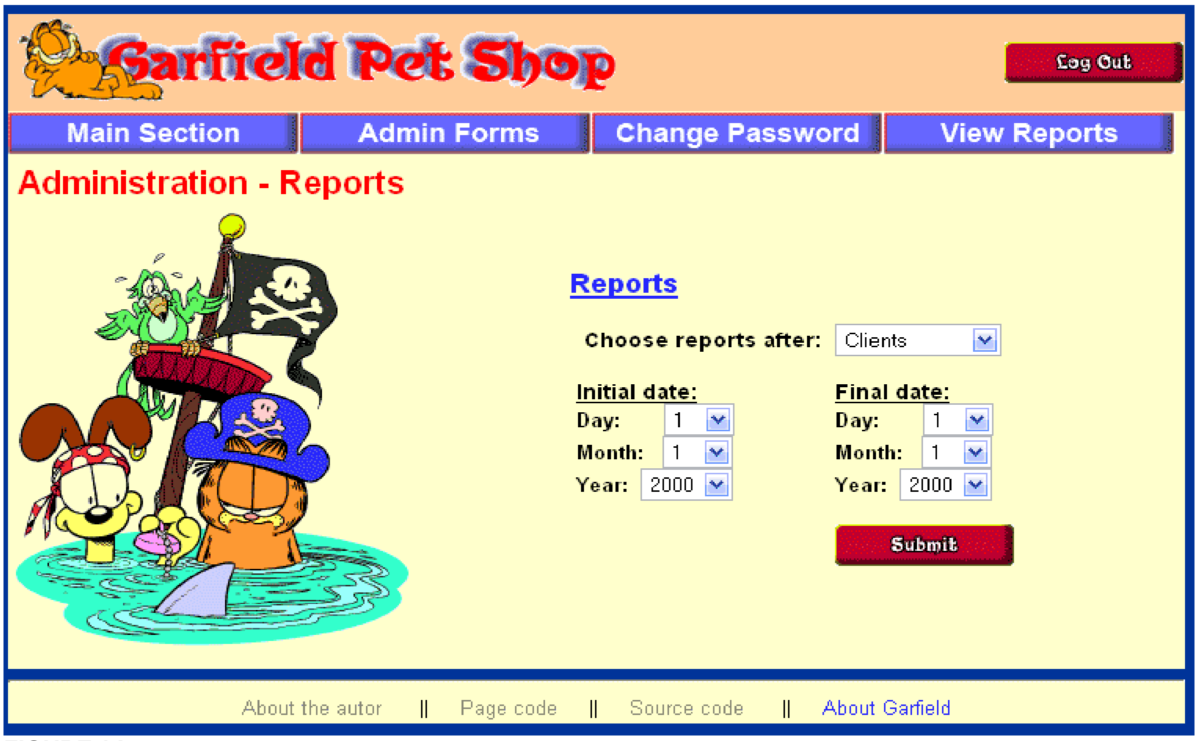
Log In if you can...

Username:
Password:

If you have forgoten your password, enter your username:

FIGURE 13
Users must pass through the login page to access the admin functions.

Get inside admin account width:
Username: razvan
Password: ******* (on demanding via email)



Administration - Reports

Choose reports after: Clients

Initial date:		Final date:	
Day:	1	Day:	1
Month:	1	Month:	1
Year:	2000	Year:	2000

FIGURE 14
The administration menu allows access to the admin functions.

The code for the admin menu is shown in *Listing 25*.

LISTING 25 admin.php—This Script Authenticates the Administrator and Lets Him Access the admin Functions

```
<?php

// include function files for this application
require_once('include_pet_fns.php');
session_start();

if ($_POST['username'] && $_POST['passwd']) {
// they have just tried logging in
    $username = _INPUT('username');
    $passwd = _INPUT('passwd');
    $login =login($username, $passwd);

    if ($login=='admin') {
      // if they are in the database register the user id
      $_SESSION['admin_user'] = $username;
    } else if ($login=='normal') {
      // if they are in the database register the user id
      $_SESSION['normal_user'] = $username;
    } else {
      // unsuccessful login
      do_html_header('Problem:');

//start table format…
 echo '<b class="species">You could not be logged in. You must be logged in to view this page.</b>';
 echo  '</td>';

 echo  '<td style="padding-right:5px;padding-left:5px">';
 display_button('login.php', 'new-Log-In', 'Log In');
 echo  '</td>';
 echo '</tr>';

 echo '<tr>';
 echo '<td>';
 echo '<br /><img alt="odie runs in circles" id="image1" align="top" src="images/odie14.gif" height="155"
width="220" />';
?>
Text…
<?php
//close table…
 do_html_footer();
 exit;
    }
}

 if (check_admin_user()) {
      $title = "Administration - Reports";
    } else if (check_normal_user()) {
      $title = "User Account - Details";
      $change_user_password = $_POST['change_user_password'];
    }

$register = $_POST['register'];
$user = get_username();
do_html_header($title);

//start table format…
 if(check_normal_user()) {
  echo "<h3><b>Welcome ".output($user)."!</b></br></h3>";
  if(isset($register)) {
    if (filled_out($HTTP_POST_VARS)) {
```

```
    $usernamex = $HTTP_POST_VARS['username'];
    $passwordx = $HTTP_POST_VARS['repassword'];
    $emailx = $HTTP_POST_VARS['email'];
    $namex = $HTTP_POST_VARS['name'];
    $addressx = $HTTP_POST_VARS['address'];
    $cityx = $HTTP_POST_VARS['city'];
    $statex = $HTTP_POST_VARS['state'];
    $zipx = $HTTP_POST_VARS['zip'];
    $phonex = $HTTP_POST_VARS['phone'];
    $countryx = $HTTP_POST_VARS['country'];

    if(change_customer_details($usernamex, $emailx, $namex, $addressx, $cityx,
                  $statex, $zipx, $phonex, $countryx)){
      echo '<b>Customer details were updated. </b><br />';
      } else {
      echo '<b>Customer details could not be updated. </b><br />';
      }

      if(make_change_in_users($usernamex,$namex,$emailx,$passwordx)){
      echo '<b>User details were updated. </b>';
      } else {
      echo '<b>User details could not be updated. </b>';
      }
     } else {
    echo '<b>You have not filled out the form.  Please try again.</b>';
   }

   }
 $theuser = get_user_details($user);
   if(isset($change_user_password)) {
    $new_passwd = $HTTP_POST_VARS['new_passwd'];
    $new_passwd2 = $HTTP_POST_VARS['new_passwd2'];
    $old_passwd = $HTTP_POST_VARS['old_passwd'];

 $form_vars = array($new_passwd, $new_passwd2, $old_passwd);
 if (!filled_out($form_vars)) {
   echo '<b class="species">You have not filled out the form completely. Please try again.</b>';
 } else {
    if (!check_password($new_passwd, $new_passwd2, $old_passwd)) {
       echo '<b class="species">&nbsp;Password invalit.&nbsp;</b>';
       } else {
        // attempt update
        if (change_password($user, $old_passwd, $new_passwd)) {
           echo '<b class="species">&nbsp;Password changed.&nbsp;</b>';
           } else {
           echo '<b class="species">&nbsp;Password could not be changed.&nbsp;</b>';
           }
       }
    }
}

 }
 echo  '</td>';

 echo  '<td style="padding-left:260px;padding-right:5px">';
if(check_normal_user()) {
echo "<form name='deleting' method = 'post' action = 'index.php' >";
echo "<input type = 'hidden' name = 'username' value = '".$theuser['username']."' >";
echo "<input type = 'hidden' name = 'repassword' value = '".$theuser['password']."' >";
echo "<input type = 'hidden' name = 'deleting' value = 'on' >";
display_form_button('delete-account', 'Delete the Account');
echo '</form>';
```

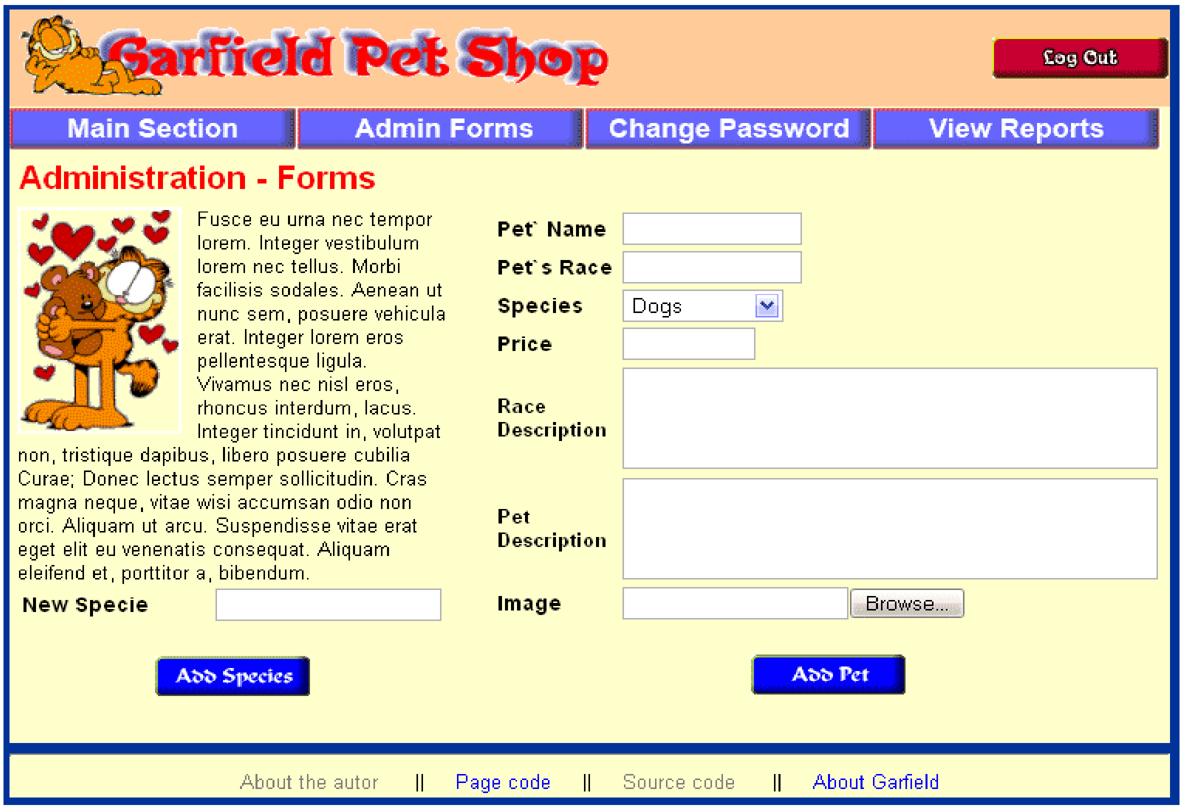
```
 }
 echo  '</td>';
 echo '</tr>';

 echo '<tr>';
 echo  '<td style="padding-left:5px;padding-right:5px">';
 if (check_admin_user()) {
  echo '<img alt="pirates (14K)" src="images/pirates.gif" height="293" width="250" />';
  } else if(check_normal_user()) {
//$theuser = get_user_details($user);
  if($customer = get_customer_details($user)) {
  display_register_form($customer,$theuser);
  }
  } else {
  echo 'You are not authorized to enter users area.';
 }
 echo  '</td>';

 echo  '<td style="padding-right:5px;padding-left:45px">';
 echo "<table><tr><td style=\"padding-right:15px;padding-left:0px\">";
 if(check_normal_user()) {
  display_search_form();
  } else if (check_admin_user()) {
  display_reports_form();
  }
 echo "</td><td style=\"padding-right:5px;padding-left:15px\">";
 if(isset($_SESSION['admin_user'])) {
     echo '&nbsp;';
    } else {
     echo ' <div id="totals">';
     echo '<u>Shopping Status</u><br />';
     echo 'Total Items : '.$_SESSION['items'];
     echo '<br />Total Price : $'.number_format($_SESSION['total_price'],2);
     echo '</div>';
}
 echo "</td></tr></table>";
  if(check_normal_user()) {
 if($customer = get_customer_details($user)) {
  change_pass_form($customer,$theuser);
  } else {
  echo 'You are not authorized to enter users area.';
  echo '<div align="center"><img alt="odie and garfield" src="images/odie_and_garfield.gif" height="200"
width="200" /></div>';
 }
  }
//close table…

do_html_footer();

?>
```

After the administrator reaches this point, he can change his password or log out. Also, the administrator may view a series of reports on clients, pets and the current stock. By viewing such reports the administrator may take various decisions concerning the administration of the virtual pet shop.

The administration user is identified after login by means of the ***$admin_user*** session variable and the *check_admin_user ()* function. This function and the others used by the administrative scripts can be found in the function library administration.php.

Even the administrator chooses to add a new category or new pets or he wants to modify an existing one, he will go to insert_pets_form.php. This script presents the administrator with two forms to fill in and also process the chosen form according to a settable variable, it is verified if the form is filled out and inserted the new data into the database.

The output of insert_pets_form.php is shown in *Figure 15.*



Administration - Forms

Fusce eu urna nec tempor lorem. Integer vestibulum lorem nec tellus. Morbi facilisis sodales. Aenean ut nunc sem, posuere vehicula erat. Integer lorem eros pellentesque ligula. Vivamus nec nisl eros, rhoncus interdum, lacus. Integer tincidunt in, volutpat non, tristique dapibus, libero posuere cubilia Curae; Donec lectus semper sollicitudin. Cras magna neque, vitae wisi accumsan odio non orci. Aliquam ut arcu. Suspendisse vitae erat eget elit eu venenatis consequat. Aliquam eleifend et, porttitor a, bibendum.

New Specie	
Pet` Name	
Pet`s Race	
Species	Dogs
Price	
Race Description	
Pet Description	
Image	

FIGURE 15
This form allows the administrator to enter new pets or species or to modufy them

The ***Category*** field for pets is an HTML SELECT element. The options for this SELECT come from a call to the *get_species ()* function.

When the ***Add Pet*** button is clicked, the insert_pet.php script will be activated. The code for this script is shown in *Listing 26.*

LISTING 26 insert_pets_form.php—This Script Validates the New Pet Data and Puts It into the Database

```
<?php

// include function files for this application
require_once('include_pet_fns.php');
session_start();


// operation type
//----------------
// operations with species
$insert_sp = $_POST['insert_sp'];
$update_sp = $_POST['update_sp'];
$delete_sp = $_GET['delete_sp'];
// operations with pets
```

```
$insert_pet = $_POST['insert_pet'];
$update_pet = $_POST['update_pet'];
$delete_pet = $_GET['delete_pet'];

// variables passed
//-----------------
// passed by species
$spid_sp = $HTTP_POST_VARS['spid_sp'];
$spname = $HTTP_POST_VARS['spname'];

$insert = $_GET['insert'];
$edit_sp = $_GET['edit_sp'];
$specie_id = $_GET['specie_id'];
$sp_id = $HTTP_GET_VARS['spid'];

//passed by pets
$edit_pet = $_GET['edit_pet'];
$petid = $_GET['petid'];
$del_petid = $_GET['del_petid'];
$del_petname = $_GET['del_petname'];
$del_race = $_GET['del_race'];

$oldpetid = $HTTP_POST_VARS['oldpetid'];
$name = $_POST['name'];
$race = $_POST['race'];
$spid = $_POST['spid'];
$price = $_POST['price'];
$description = $_POST['description'];
$about = $_POST['about'];
$pic_name = $_FILES['picture']['name'];
$pic_tmp_name = $_FILES['picture']['tmp_name'];
$pic_type = $_FILES['picture']['type'];

$form_vars_sp1 = array($spname);
$form_vars_sp2 = array($spid_sp, $spname);
$form_vars_pet = array($name, $race, $price);


do_html_header('Administration - Forms');

if (check_admin_user()) {

if(isset($insert_sp)) {
if (filled_out($form_vars_sp1)) {
if(insert_sp($spname)) {
     echo "<b class=\"species\">Category '$spname' was added to the database.</b>";
 } else {
     echo "<b class=\"species\">Category '$spname' could not be added to the database.</b>";
  }
 } else {
   echo '<b class="species">You have not filled out the form. Please try again.</b>';
  }
}

if(isset($update_sp)) {
if (filled_out($form_vars_sp2)) {
if(update_sp($spid_sp, $spname)) {
     echo '<b class="species">Species was updated.</b>';
 } else {
     echo '<b class="species">Species could not be updated.</b>';
     }
  } else {
    echo '<b class="species">You have not filled out the form. Please try again.</b>';
  }
}

if(isset($delete_sp)) {
  if (isset($specie_id)) {
    if(delete_sp($specie_id)) {
      echo '<b class="species">Specie was deleted.</b>';
   } else {
      echo '<b class="species">Specie could not be deleted. This is usually because it is not empty.</b>';
          }
  } else {
    echo '<b class="species">No category specified. Please try again.</b>';
  }
}

if(isset($insert_pet)) {
$speciename = get_the_species($spid);
if (filled_out($form_vars_pet)) {
 if(insert_pet($name, $race, $spid, $price, $description, $about)){
      echo "<b class=\"species\">Pet '".stripslashes($name)."' was added to the database.</b>";
   if(insert_pictures($name, $race, $speciename, $pic_name, $pic_tmp_name, $pic_type)) {
      echo "<b class=\"species\">Picture '".stripslashes($name)."' was added to the database.</b>";
      } else {
      echo "<b class=\"species\">Picture '".stripslashes($name)."' could not be added to the database.</b>";
      }
    } else {
      echo "<b class=\"species\">Pet '".stripslashes($name)."' could not be added to the database.</b>";
          }
  } else {
    echo '<b class="species">You have not filled out the form. Please try again.</b>';
    }
}

if(isset($update_pet)) {
$speciename = get_the_species($spid);
$oldpet = get_pet_details($oldpetid);
if (filled_out($form_vars_pet)) {
   if(update_pet($oldpetid, $name, $race, $spid, $price, $description, $about)) {
      echo '<b class="species">Pet details were updated.</b>';
      } else {
      echo '<b class="species">Pet details could not be updated.</b>';
     }
   if(insert_pictures($name, $race, $speciename, $pic_name, $pic_tmp_name, $pic_type)) {
      echo "<b class=\"species\">Picture '".stripslashes($name)."' was added to the database.</b>";
      delete_picture($oldpet['name'], $speciename);
      } else {
      }
  } else {
    echo '<b class="species">You have not filled out the form. Please try again.</b>';
    }
}

if(isset($delete_pet)) {
$pet_details = get_pet_details($del_petid);
$del_speciename = get_the_species($pet_details['spid']);

if (isset($del_petid)) {
  if(delete_pet($del_petname, $del_speciename, $del_race)) {
      echo '<b class="species">Pet '.$del_name.' was deleted.</b>';
   } else {
      echo '<b class="species">Pet '.$del_name.' could not be deleted.</b>';
      }
   delete_picture($del_petname, $del_speciename);
  } else {
    echo '<b class="species">We could not delete this pet. Please try again.</b>';
  }
}

//start table format…
echo '<img alt="love pets" id="image1" src="images/love_pets.gif" height="150" width="102" />';
?>
//Text…
<?php
 echo "<br />";
 if(isset($edit_sp)) {
   if ($sp_name = get_sp_name($sp_id)) {
    $sp = compact('sp_name', 'sp_id');
    display_sp_form($sp);
  } else {
    echo '<b class="species">Could not retrieve species details.</b>';
   }
 } else {
display_sp_form();
}
echo  '</td>';

echo  '<td width="59%" style="padding-left:15px;padding-right:5px;">';
if(isset($edit_pet)) {
if ($pet = get_pet_details($petid)){
display_pet_form($pet);
} else {
echo '<b class="species">Could not retrieve pet details.</b>';
  }
} else {
display_pet_form();
}
//close table…
} else  {
  echo '<b class="species">You are not authorized to enter the administration area.</b>';
}

do_html_footer();
?>
```

This script calls all the functions neaded for the operations above like *insert_sp (), insert_pet (), update_sp (), update_pet (), delete_sp () and delete_pet ().* This function and the others used by the administrative scripts can be found in the function library administration.php. In addition to adding new species and pets, the administrative user can edit and delete these items. This was implemented by reusing some code. When the administrator clicks the ***Go to main site*** button in the administration menu, he will go to the category index at index.php and can navigate the site in the same way as a regular user, using the same scripts.

There is a difference in the administrative navigation, however: Administrators will see different options based on the fact that they have the registered session variable ***$admin_user***.

Redimensioning images

When images are displayed on page we have the option of using a redimensioning of images loaded by the administrator, relieving him from the responsibility of choosing an image with a proper format which may be displayed without distorsioning the page format.

To this end, the following change is to be made within the *display_pet_details ()* function:

```
        if (@file_exists('Pets/'.$spname.'/'.$pet['race'].'.jpg'))
  {
                $size = GetImageSize('Pets/'.$spname.'/'.$pet['race'].'.jpg');
    if($size[0]>0 && $size[1]>0)
        $pet_race = $pet['race'];
        $path = "Pets/$spname/$pet_race.jpg";
 // echo '<td><img src=\'Pets/'.$spname.'/'.$pet['race'].'.jpg\' border=0 '.$size[3].'></td>';
//echo '<td><img src='.$path.' border=0 '.$size[3]. '></td>';
echo '<td><img src=resize_image.php?image=';
echo urlencode($path);
echo  '&max_width=150&max_height=150 border=0 '.$size[3]. '></td>';
```

where the *resize_image.php* file is displayed in the following listing:

LISTING 27 resize_image.php—This script redimensions images added by user to pre-established dimensions

```
<?php
  $image = $_REQUEST['image'];
  $max_width = $_REQUEST['max_width'];
  $max_height = $_REQUEST['max_height'];

  if (!$max_width)
    $max_width = 80;
  if (!$max_height)
    $max_height = 60;

  $size = GetImageSize($image);
  $width = $size[0];
  $height = $size[1];
  $x_ratio = $max_width / $width;
  $y_ratio = $max_height / $height;

  if ( ($width <= $max_width) && ($height <= $max_height) ) {
    $tn_width = $width;
    $tn_height = $height;
  }
  else if (($x_ratio * $height) < $max_height) {
    $tn_height = ceil($x_ratio * $height);
    $tn_width = $max_width;
  }
  else {
    $tn_width = ceil($y_ratio * $width);
    $tn_height = $max_height;
  }

  $src = ImageCreateFromJpeg($image);
  $dst = ImageCreate($tn_width,$tn_height);
  ImageCopyResized($dst, $src, 0, 0, 0, 0,
     $tn_width,$tn_height,$width,$height);
  header('Content-type: image/jpeg');
  ImageJpeg($dst, null,75);
  ImageDestroy($src);
  ImageDestroy($dst);
?>
```

NOTE:

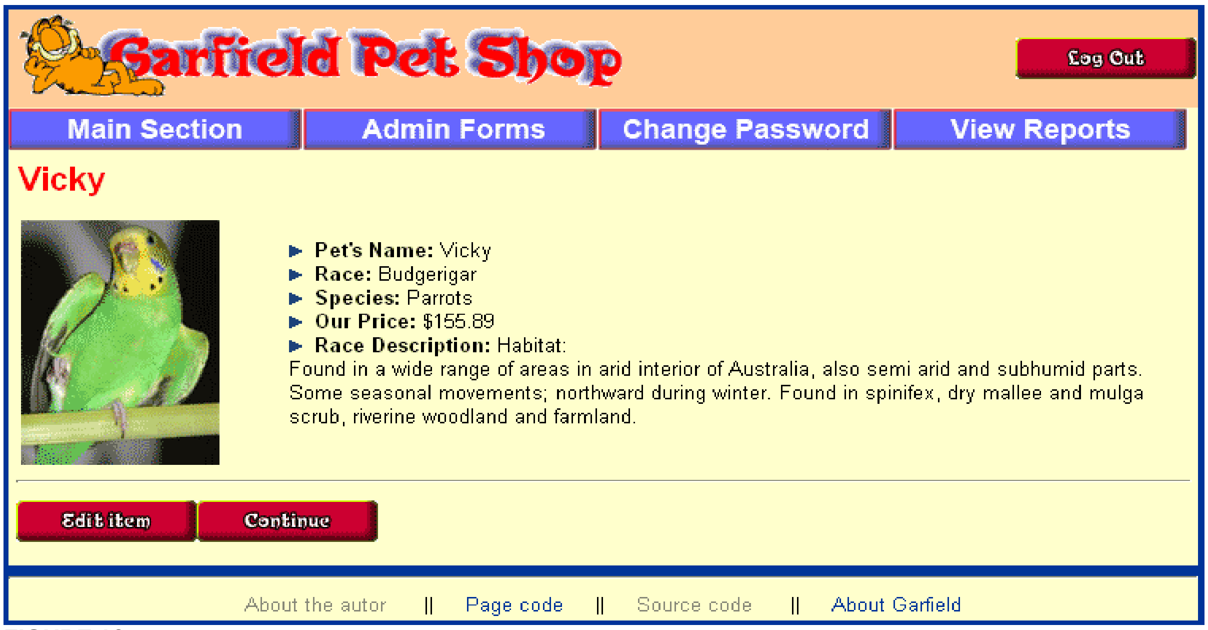
This functionality is not yet implemented, consequently it is use a specialized program such as *IrfanView* for redimensioning and then pictures are uploaded into the relevant directory.

The images are to be inserted in the corresponding directories by using the *insert_pictures ()* function presented in the following listing:

LISTING 28 inert_pictures() from administration.php — This function takes the picture uploaded by the administrator and moves it into the relevant directory

```
function insert_pictures($name, $race, $spname, $picname, $pictmpname, $pictype){
//insert pictures

$pet_name = str_replace(" ","_","$name");
$specie_name = str_replace(" ","_","$spname");

if ((isset($picname) && is_uploaded_file($pictmpname))) {
$type = basename($pictype);
    switch ($type) {
    case 'gif':
    case 'png':
    case 'jpeg':
    case 'pjpeg':   $filename = "images/Pets/$specie_name/$pet_name.jpg";
  if(!move_uploaded_file($pictmpname, $filename)) {
  echo '<b class="species">We could not move picture to destination.</b><br />';
  return false;
  }
  break;
  default:  echo '<b>Invalid picture format: </b>'.$pictype;
  }
        return true;
        } else {  //end inser images
        echo '<b class="species">&nbsp;You did not make any picture upload.&nbsp;</b>';
    return false;
    }
}
```

Also, before the image file is moved to the relevant directory, its extension is checked in order to verify whether the file does contain an image or not.



Vicky

- **Pet's Name:** Vicky
- **Race:** Budgerigar
- **Species:** Parrots
- **Our Price:** $155.89
- **Race Description:** Habitat:

Found in a wide range of areas in arid interior of Australia, also semi arid and subhumid parts. Some seasonal movements; northward during winter. Found in spinifex, dry mallee and mulga scrub, riverine woodland and farmland.

FIGURE 16
The show_pet.php script produces different output for an administrative user.

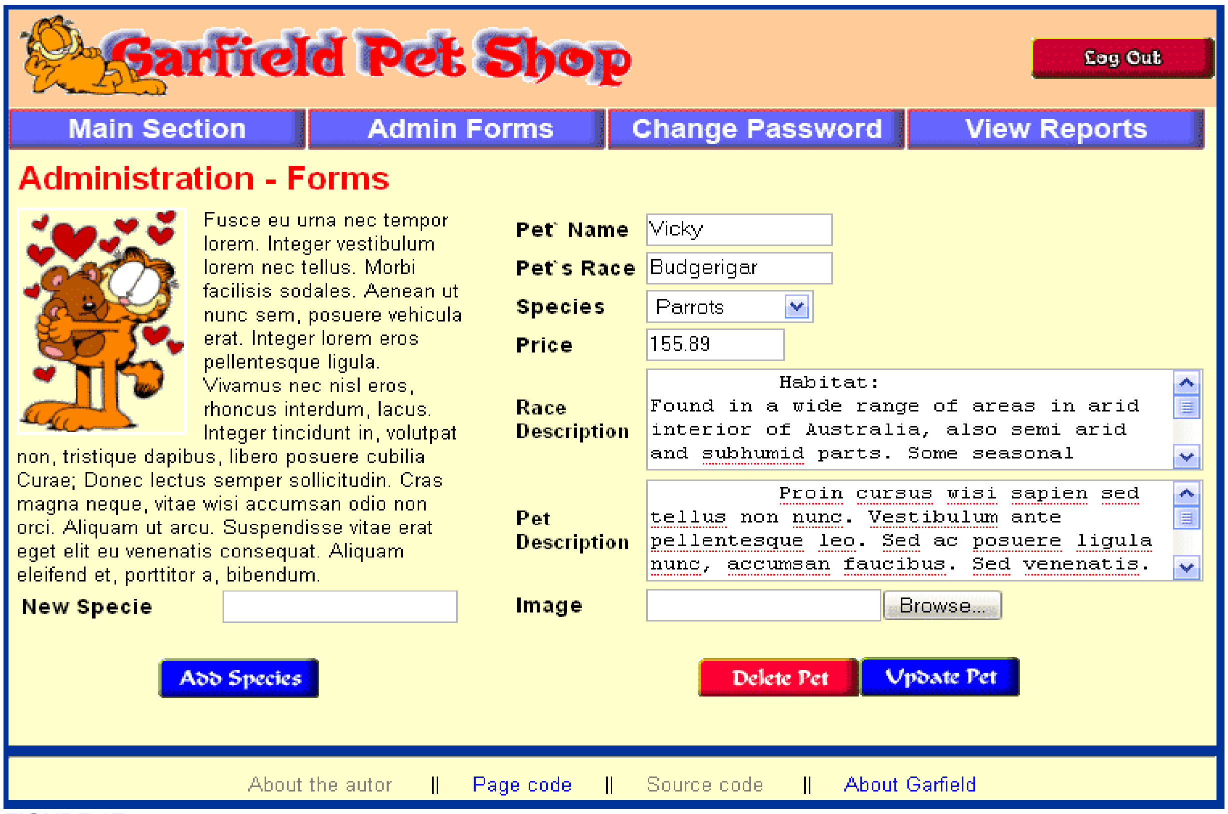
The administrator has access to two new options on this page: ***Edit Item*** and ***Admin Menu***. The ***shopping cart*** in the upper-right corner will disappear—instead, a ***Log Out*** button will appear.

The code for this is here:

```
if( check_admin_user() ) {
   display_button("insert_pets_form.php?edit_pet=on&petid=$petid", 'edit-item', 'Edit Pet');
   display_button($target, 'continue', 'Continue');
 } else {
   display_button("show_cart.php?new=$petid&spname=$spname", 'add-to-cart-back', 'Add '
                 .$pet['name'].' To My Shopping Cart');
   display_button($target, 'continue-shopping', 'Continue Shopping');
 }
```

Looking back at the show_sp.php script, it can be seen that it too has these options built in to it.

If the administrator clicks the ***Edit Item*** button, he will go to the insert_pets_form.php script. The output of this script is shown in *Figure 17*.



Administration - Forms

Fusce eu urna nec tempor lorem. Integer vestibulum lorem nec tellus. Morbi facilisis sodales. Aenean ut nunc sem, posuere vehicula erat. Integer lorem eros pellentesque ligula. Vivamus nec nisl eros, rhoncus interdum, lacus. Integer tincidunt in, volutpat non, tristique dapibus, libero posuere cubilia Curae; Donec lectus semper sollicitudin. Cras magna neque, vitae wisi accumsan odio non orci. Aliquam ut arcu. Suspendisse vitae erat eget elit eu venenatis consequat. Aliquam eleifend et, porttitor a, bibendum.

Pet` Name	Vicky
Pet`s Race	Budgerigar
Species	Parrots
Price	155.89
Race Description	Habitat: Found in a wide range of areas in arid interior of Australia, also semi arid and subhumid parts. Some seasonal
Pet Description	Proin cursus wisi sapien sed tellus non nunc. Vestibulum ante pellentesque leo. Sed ac posuere ligula nunc, accumsan faucibus. Sed venenatis.
Image	
New Specie	

Browse...
Add Species
Delete Pet
Update Pet
About the autor || Page code || Source code || About Garfield

FIGURE 17
The edit_pet_form.php script gives the administrator access to edit pet details or delete a pet
.

This is, in fact, the same form used to get the pet's details in the first place. An option was built into that form to pass in and display existing pet data. The same thing was done with the species form.

LISTING 29 display_pet_form() Function from administration.php—This Form Does Double Duty as an Insertion and Editing Form

```
function display_pet_form($pet = '')
// This displays the pet form.
// form will be displayed with the old data and point to update_pet.php.
{
 // if passed an existing pet, proceed in "edit mode"
  $edit = is_array($pet);
?>
  <form onsubmit="return validAddPet(this)" name="add_pet" method='post'
        action="insert_pets_form.php" enctype="multipart/form-data">
  <table border="0">
  <tr>
    <td class="species">Pet` Name</td>
    <td><input type='text' name='name' size="15" maxlength="20"
         value="<?php echo $edit?$pet['name']:''; ?>"></td>
  </tr>

  <tr>
    <td class="species">Pet`s Race</td>
    <td><input type='text' name='race' size="15" maxlength="35"
         value="<?php echo $edit?$pet['race']:''; ?>"></td>
   </tr>
   <tr>
```

```
      <td class="species">Species</td>
      <td><select name='spid'>
      <?php
         // list of possible species comes from database
         $sp_array = get_species();
         foreach ($sp_array as $thissp) {
              echo '<option value="';
              echo $thissp['spid'];
              echo '"';
              // if existing pet, put in current specie
              if ($edit && $thissp['spid'] == $pet['spid'])
                 echo ' selected';
              echo '>';
              echo $thissp['spname'];
              echo "\n";
         }
      foreach ($sp_array as $thisspecies) {
      echo '<input type="hidden" name="spname" value="'.$thisspecies['spname'].'">';
      }
         ?>
         </select>
       </td>
   </tr>
   <tr>
    <td class="species">Price</td>
    <td><input type='text' name='price' size="10" maxlength="30"
              value="<?php echo $edit?$pet['price']:''; ?>"></td>
   </tr>

   <tr class="species">
     <td>Race Description</td>
     <td><textarea rows="3" cols="40" name='description'>
         <?php echo $edit?$pet['description']:''; ?>
         </textarea></td>
    </tr>
    <tr class="species">
     <td>Pet Description</td>
     <td><textarea rows="3" cols="40" name='about'>
         <?php echo $edit?$pet['about']:''; ?>
         </textarea></td>
    </tr>
              <tr>
              <td class="species">Image</td>
              <td><input name="picture"  type="file" /></td>
              </tr>
    <tr>
    <?php
       echo '<td colspan="2" align="center">';
       if($edit) {
       // we need the old petid to find pet in database
       // if the petid is being updated
       echo '<input type="hidden" name="update_pet" value="on">';
       echo '<input type="hidden" name="oldpetid" value="'.$pet['petid'].'">';
       echo '<input type="hidden" name="petid" value="'.$pet['petid'].'">';
       echo '<input type="hidden" name="race" value="'.$pet['race'].'">';
       echo '<input type="hidden" name="spname" value="'.$thisspecies['spname'].'">';
       $target =
"insert_pets_form.php?delete_pet=on&del_petid=".$pet['petid']."&del_petname=".$pet['name']."&del_race=".
$pet['race'];
       display_button($target, 'new-Delete-Pet', 'Delete Pet');
       display_form_button('new-Update-Pet-back', 'Update Pet');
       } else {
        echo '<input type="hidden" name="insert_pet" value="on">';
        display_form_button('new-Add-Pet-back', 'Add Pet');
    }
       echo '</td>';
  ?>
     </tr>
  </table>
  </form>
<?php
}
```

If in an array containing the pet data has passed, the form will be rendered in edit mode and will fill in the fields with the existing data:

```
<input type=text name=price value="<?=$edit?$pet["price"]:""; ?>">
```

For the edit form there are two submit buttons—one to update the pet, and one to delete it. These call the scripts edit_pet.php and delete_pet.php, which update the database accordingly.

The species versions of these scripts work in much the same way except for one thing. When an administrator tries to delete a species, it will not be deleted if any pets are still in it. This is checked with a database query. This avoids any problems that one might get with deletion anomalies. If a species was deleted that still had pets in it, these pets would become orphans. We wouldn't know what species they were in, and we would have no way of navigating to them! That's the overview of the administration interface

Generating reports

An important characteristic of the administration part is the generation of various reports which may help the administrator to take decisions in accordance to results.
The form for selecting the type of report is simple. It is taken over by the reports.php file presented in the following listing:

LISTING 30 reports.php—This script shall display one out of three types of reports implemented for this variant of the project

```
<?php
  include ('include_pet_fns.php');
  session_start();

        $referer=$_SERVER['HTTP_REFERER'];
        $report_type = $_POST['report_type'];

  $initial_day = $_POST['iday'];
  $final_day = $_POST['fday'];
  $initial_month = $_POST['imonth'];
  $final_month = $_POST['fmonth'];
  $initial_year = $_POST['iyear'];
  $final_year = $_POST['fyear'];

  $i_date = array ("$initial_year","$initial_month","$initial_day");
  $f_date = array ("$final_year","$final_month","$final_day");

        $initial_date = implode('-',$i_date);
        $final_date = implode('-',$f_date);

  if($report_type == 'clients'){
        $some = "Clients";
        } else if($report_type == 'pets'){
        $some = "Pets";
        } else if($report_type == 'stock'){
        $some = "Stock";
        } else {
        $some = "Some";
         }

   do_html_header("$some details");

?>
   <table border="0" cellspacing="0" cellpadding="0" width="600">
   <tr>
   <td width="50%" style="padding-left:165px;padding-right:5px" class="species">Initial date: <?php echo
$initial_date; ?></td>
   <td width="50%" style="padding-left:5px;padding-right:5px" class="species">Final date: <?php echo
$final_date; ?></td>
   </tr>
   </table>
<?php
echo '<div align="center">';
  if($report_type == 'clients'){
        display_clients_details($initial_date, $final_date);
        } else if($report_type == 'pets'){
        display_pets_details($initial_date, $final_date);
        } else if($report_type == 'stock'){
        display_all_pets();
        } else {
        echo '<b class="species">Sorry, we could not give you any report.</b>';
         }
echo '</div>';

        echo '<div align="center">';
         display_button("$referer", 'back', 'Back');
        echo '</div>';

do_html_footer();
?>
```

The client report presents details concerning the client, including the sum spent in order to buy from `*our shop*`. This report is useful for discounts or bonuses destined to clients who have made orders surpassing a certain amount during a given period of time. It may also be used in order to obtain clients' details to the purpose of settling any litigation.

LISTING 31 display_clients_details Function from administration.php—This function displays data of clients who have shopped during a certain period of time.

```
            function display_clients_details($initial_date, $final_date){

            $conn = db_connect();
            $query = "
            SELECT
            c.name name, c.address, c.city , c.state , c.country , c.phone ,
            sum(o.amount) as 'TOTAL AMOUNT', o.date
            FROM  customers c, orders o
            WHERE c.customerid=o.customerid
            AND o.order_status = 'FINAL'
            AND o.date BETWEEN '$initial_date' AND '$final_date'
            GROUP BY name DESC
            ORDER BY sum(o.amount) DESC";
```

```
             $result = mysql_query($query);

            if(mysql_num_rows($result) <1 ) {
                    echo '<b class="species">We have not found eny clients.</b><br /> ';
              return false;
             } else {
               $nr_results = mysql_num_rows($result);
            echo '<div id="the_report">';
                  echo '<table>';
                  echo '<tr>';
                echo '<th>&nbsp;Name&nbsp;</th> <th>&nbsp;Address&nbsp;</th> <th>&nbsp;City&nbsp</th>
<th>&nbsp;State&nbsp;</th> <th>&nbsp;Country&nbsp;</th> <th>&nbsp;Phone&nbsp;</th>
<th>&nbsp;Amount&nbsp;</th> <th>&nbsp;Last Date&nbsp;</th>';
                   echo '</tr>';
                   for($i=0; $i<$nr_results; $i++){
                    $clients= mysql_fetch_assoc($result);
                    echo '<tr>';
                         foreach($clients as $key => $details){
                         echo "<td >&nbsp; $details  &nbsp;</td>";
                         }
                         echo '</tr>';
                         }
                         echo '</table>';
                         echo '</div>';
                         echo '<hr />';
                  }
            }
```

The report on pets presents certain details with regard to animals which have been bought during a certain period of time, the administrator being able to make comparisons between the price of the sale and the present price of the `product` in store, as well as between sold quantities and the ones available within the respective interval of time.

LISTING 32 display_pets_details Function from administration.php—This function displays details about sold pets as well as relevant comparisons

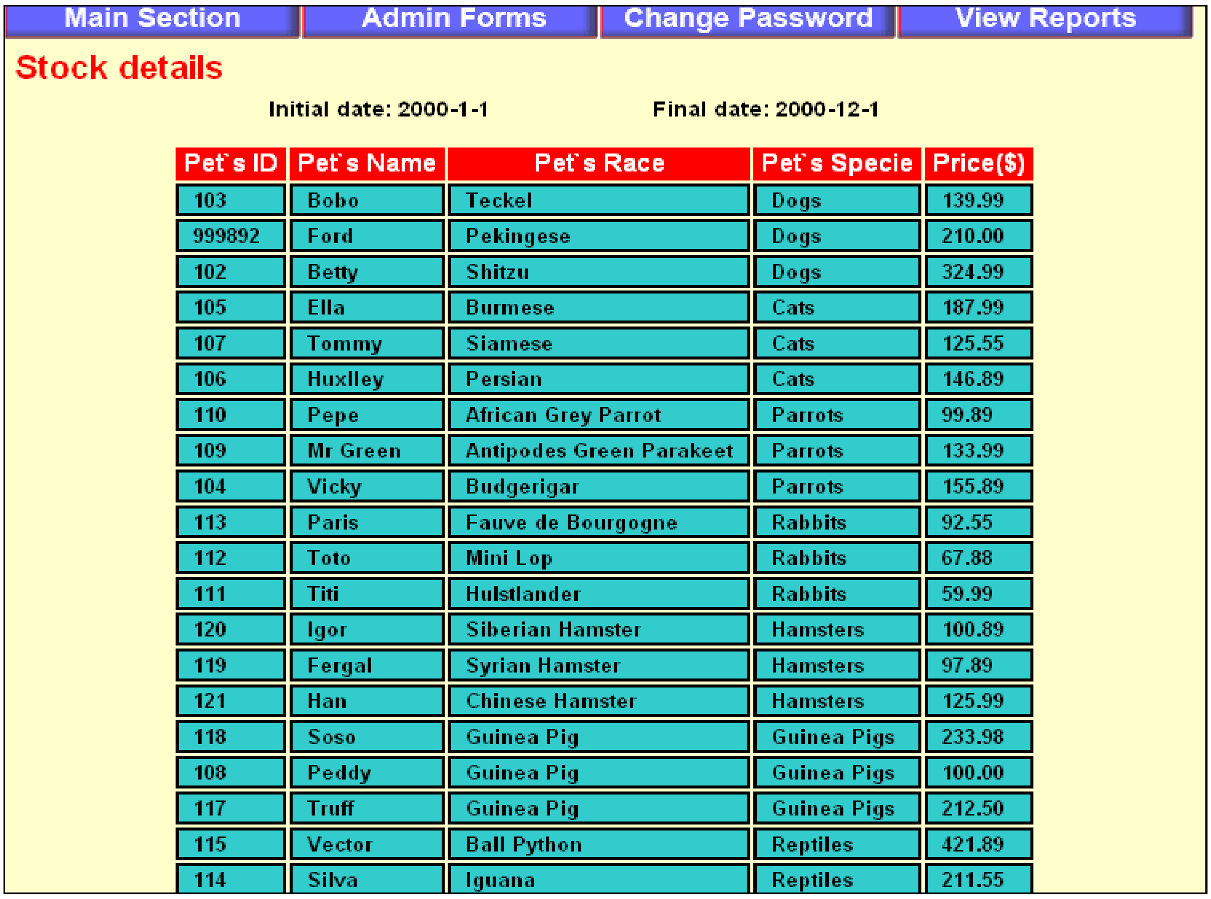
```
function display_pets_details($initial_date, $final_date){

  $conn = db_connect();
$query = "
SELECT p.name NAME, s.spname 'SPECIES NAME', p.race RACE, p.price 'STOCK PRICE', oi.item_price
'SOLD PRICE'
FROM species s, pets p, order_items oi, orders o
WHERE s.spid=p.spid
AND p.petid=oi.petid
AND oi.orderid=o.orderid
AND o.order_status = 'FINAL'
AND o.date BETWEEN '$initial_date' AND '$final_date'
GROUP BY RACE ";
$result = mysql_query($query);

  if(mysql_num_rows($result) <1 ) {
   echo '<b class="species">We have not found any pets.</b><br /> ';
   return false;
  } else {
       $nr_results = mysql_num_rows($result);
echo '<div id="the_report">';
       //start table format…
       echo '<th>&nbsp;Pet`s Name&nbsp;</th> <th>&nbsp;Species Name&nbsp;</th> <th>&nbsp;Pet`s
Race&nbsp;</th> <th>&nbsp;Stock Price($)&nbsp;</th> <th>&nbsp;Sold Price($)&nbsp;</th>  ';
       echo '</tr>';
```

```
         for($i=0; $i<$nr_results; $i++){
          $pets = mysql_fetch_assoc($result);
               echo '<tr>';
               foreach($pets as $key => $details){
               echo "<td>&nbsp $details  &nbsp</td>";
               }
               echo '</tr>';
          }
//close table…
        }
}
```

The „Check Stock" Report displays details of pets presently in store at the virtual pet shop, grouped in accordance to their category/species. Within this variant of the function initial and final data of the report bear no relevance.

LISTING 33 display_all_pets Function from administration.php—This function displays details of pets presently in store

```
function display_all_pets(){

$conn = db_connect();
$query = "
SELECT p.petid ID, p.name NAME, p.race RACE, s.spname 'SPECIES NAME', p.price PRICE
FROM species s, pets p
WHERE s.spid=p.spid
ORDER BY s.spid";
$result = mysql_query($query);

  if( mysql_num_rows($result) <1 ) {
  echo '<b class="species">We have not found any pets.</b><br /> ';
  return false;
  } else {
  $nr_results = mysql_num_rows($result);
 echo '<div id="the_report">';
        //start table format…
        echo '<th>&nbsp;Pet`s ID&nbsp;</th><th>&nbsp;Pet`s Name&nbsp;</th><th>&nbsp;Pet`s
Race&nbsp;</th> <th>&nbsp;Pet`s Specie&nbsp;</th> <th>&nbsp;Price($)&nbsp;</th>';
        echo '</tr>';
         for($i=0; $i<$nr_results; $i++){
          $pets = mysql_fetch_assoc($result);
          echo '<tr>';
               foreach($pets as $key => $details) {
               echo "<td>&nbsp $details  &nbsp</td>";
               }
               echo '</tr>';
               }
               //close table…
        }
}
```



Stock details

Initial date: 2000-1-1 Final date: 2000-12-1

Pet`s ID	Pet`s Name	Pet`s Race	Pet`s Specie	Price($)
103	Bobo	Teckel	Dogs	139.99
999892	Ford	Pekingese	Dogs	210.00
102	Betty	Shitzu	Dogs	324.99
105	Ella	Burmese	Cats	187.99
107	Tommy	Siamese	Cats	125.55
106	Huxlley	Persian	Cats	146.89
110	Pepe	African Grey Parrot	Parrots	99.89
109	Mr Green	Antipodes Green Parakeet	Parrots	133.99
104	Vicky	Budgerigar	Parrots	155.89
113	Paris	Fauve de Bourgogne	Rabbits	92.55
112	Toto	Mini Lop	Rabbits	67.88
111	Titi	Hulstlander	Rabbits	59.99
120	Igor	Siberian Hamster	Hamsters	100.89
119	Fergal	Syrian Hamster	Hamsters	97.89
121	Han	Chinese Hamster	Hamsters	125.99
118	Soso	Guinea Pig	Guinea Pigs	233.98
108	Peddy	Guinea Pig	Guinea Pigs	100.00
117	Truff	Guinea Pig	Guinea Pigs	212.50
115	Vector	Ball Python	Reptiles	421.89
114	Silva	Iguana	Reptiles	211.55

FIGURE 18
List displaying details concerning pets in store at this moment

Extending the Project

The Project shall be extended by the addition of a newsletter function in order to keep in touch with clients, the latter being able to receive thereby various information regarding newly added products or new facilities of the virtual pet shop which may appear.