

SZABADKAI MŰSZAKI SZAKFŐISKOLA
SZABADKA



Web portál készítés

PROJEKTUM

Web programozás tantárgyból

témavezető: Dr. Čović Zlatko
főiskolai tanár
katedra főnök

hallgató[1]: Hodán Valentina
leckekönyv: 12214150
hallgató[2]: Kricskovity Nikoletta
leckekönyv: 12214106
szakirány: Műszaki informatika

Szabadka, 2016

Tartalom:

1. A feladatkör meghatározása	3
2. Kidolgozás	3
2.1 XHTML állományok	3
2.1.1. Elsődleges oldal állományai (main mappa).....	3
2.1.2. Adminisztrációs oldal állománya (admin mappa)	4
2.2 CSS állományok	4
2.3 PHP állományok	4
2.3.1. A „háttér” oldal állományai	4
2.3.2. Elsődleges oldal állományai (main mappa).....	5
2.3.3. Adminisztrációs oldal állományai (admin mappa).....	9
2.4 Javascript állományok	14
3. Az adatbázis struktúrája.....	18
4. A projekt működésének leírása.....	19
4.1. Elsődleges felület (Vendégfogadó felület)	19
4.2. Adminisztrációs felület (Regisztrált felhasználóknak).....	22
Melléklet.....	24
Felhasznált irodalom	24

1. A feladatkör meghatározása

Web portál készítését választottuk témánkul, amelynek megvalósításához alkalmazunk kellett a következő technológiákat: (X)HTML, CSS, JavaScript, AJAX, PHP és MySQL.

Emellett alkalmaztunk még *jQuery* technológiát is.

Így a projektben egy olyan honlap lett létrehozva, melyben az Avon szépségipari cég egyes termékei vannak nyilvántartva. Innen ered a neve: **Avon Cosmetics**. Lehetővé tettük az összes termék megtekintését illetve a köztük való keresést. Másrészt készítettünk egy adminisztrációs oldalt, ahova bejelentkezni vagy regisztrálni kell. Az adminisztrációs felületen új terméket lehet hozzáadni a listához, megváltoztatni vagy törölni lehet azt.

2. Kidolgozás

Ebben a részben ismertetjük a kód részeit, magyarázattal. Minden kód megtalálható a Mellékletben feltüntetett helyen.

2.1 XHTML állományok

2.1.1. Elsődleges oldal állományai (main mappa)

aboutus.xhtml

Az indexoldal betöltésekor megjelenő elsődleges lap szöveges tartalmát foglalja magában `<div></div>`-ek között. Címsor jelölőelemeket: `<h1></h1>`, `<h2></h2>` illetve paragrafust jelölőelemeket: `<p></p>` tartalmaz. Ezenkívül új sort jelölő jelölőelemet is tartalmaz. (`
`)

contact.xhtml

Amikor a *Contact* föltre kattint a felhasználó, indexoldal tartalma ebből az állományból lesz betöltve. Szöveges tartalmat illetve a következő jelölő elemeket foglalja magában: Gyökér jelölőeleme: `<div></div>`. Ezenkívül még címsor jelölőelemeket: `<h1></h1>`, `<h2></h2>` illetve paragrafust jelölő elemeket: `<p></p>` is tartalmaz. Új sort jelölő (`
`) és vízszintes vonalat író jelölőelemet (`<hr/>`) tartalmaz még.

search.xhtml

```
<div>
  <script type="text/javascript" src="../js/forajax.js"></script>
  <h1>Search in the product names</h1>
  <form method="get">
    <label for="search">Keyword: </label><input id="search" type="text"
name="search" />
    <div id="suggest"></div>
    <input type="hidden" name="op" value="search-result"/><br /><br />
    <input class="btn" type="submit" name="submit" value="search"/>
  </form>
</div>
```

Gyökér jelölőeleme: `<div></div>`. Tartalmaz egy külső script fájlt, egy címsor, illetve egy űrlap jelölőelemet(`<form></form>`). Az űrlap fogad egy szöveges értéket, `<input/>` az mezőn

keresztül, melyet továbbítani fog az index oldalra. Egy másik `<input/>` mezővel, ami nem látható a felhasználó számára, jelezzük az index oldalra, hogy ezt az űrlapot melyik állománynak kell feldolgoznia. A későbbiekben lesz ismertetve, a 2.3.2.-es alcím alatt az az index állomány, amely kezeli az egész oldalt.

2.1.2. Adminisztrációs oldal állománya (admin mappa)

home.xhtml

Az adminisztrációs indexoldal betöltésekor annak szöveges tartalmát foglalja magában. Gyökér jelölőeleme: `<div></div>`. Címsor jelölőelemeket: `<h1></h1>`, illetve paragrafust jelölőelemeket: `<p></p>` tartalmaz. Tartalmaz még leíró listát jelző jelölőelemeket: `<dl></dl>`. A leíró lista címét: `<dt></dt>`, míg a leíró lista tartalmát a: `<dd></dd>` elemek jelzik. Ezenkívül új sort jelölő jelölőelemet is tartalmaz. (`
`)

2.2 CSS állományok

mainstyle.css

A *main* mappában található elemek, oldalak stílusát határozza meg. Ezt a stílust látja először a felhasználó mikor megnyitja az oldalt.

sliderstyle.css

Az *index.php* oldalon található önműködő képnézegető stílusát határozza meg.

adminstyle.css

Az *admin* mappában található elemek, oldalak stílusát adja. Ezt a stílust akkor látja a felhasználó mikor bejelentkezik az adminisztrációs felületre.

2.3 PHP állományok

2.3.1. A „háttér” oldal állományai

index.php

Ez az állomány található a legmagasabb szinten. Csupán annyit tesz, hogy a felhasználót belépteti a *main* mappába, ahol maga a főoldal található.

db_config.php

Az adatbázishoz való kapcsolódást teszi lehetővé. Azért van külön állományban ez a kód, mert több másik állománynak van szüksége arra, hogy az adatbázishoz kapcsolódjon. Ez megkönnyíti a változtatások elvégzését, ha például, másik adatbázishoz szeretnénk kapcsolódni. Egy biztonsági feltételnek feleltet minden olyan állományt, amely kapcsolódni próbál az adatbázishoz, és csak akkor hozza létre a kapcsolatot, ha az állomány megfelelt.

```
if (defined('VARIABLE') AND VARIABLE=='jrUmr1bHRevghskwpLdpoxkksqe')
```

2.3.2. Elsődleges oldal állományai (main mappa)

index.php

Fej jelölőelemében tartalmazza a honlap címét, meta-adatokat, a linkelt stílus állományait (*mainstyle.css*, *sliderstyle.css*), illetve a külső script fájlok hivatkozását. Az első, az interneten fellelhető jQuery állomány, melyre a *slider.js* jQuery-t tartalmazó külső JavaScript állománynak van szüksége.

```
<head>
  <title>Avon Cosmetics</title>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
  <meta http-equiv="content-language" content="en"/>
  <link type="text/css" rel="stylesheet" href="../css/mainstyle.css"/>
  <link href="../css/sliderstyle.css" rel="stylesheet" type="text/css"/>
  <script type="text/javascript"
src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.2/jquery.min.js">
</script>
  <script type="text/javascript" src="../js/slider.js"></script>
</head>
```

A test jelölőelem, amely maga az ablak, fejléce tartalmazza a képeket a képnézegetőhöz, illetve az oldalon megjelenő címsort.

```
<body>
<div id="header">
  <div class="slideshow">
    <ul class="slider">
      <li></li>
      <li></li>
      <li></li>
      <li></li>
      <li></li>
      <li></li>
      <li></li>
      <li></li>
      <li></li>
      <li></li>
    </ul>
  </div>
  <h1>Avon Cosmetics</h1>
</div>
```

A fejléc alatt található egy navigációs panel, melyen keresztül elérhetjük az oldal különböző pontjait, miközben végig az index oldalon maradunk. Ezt egy *op* változó segítségével teszünk lehetővé, melyet az alább látható php kód résszel kezelünk.

```
<div id="navigation">
  <ol>
    <li><a href="index.php">About us</a></li>
    <li><a href="index.php?op=contact">Contact</a></li>
    <li><a href="index.php?op=products">Products</a></li>
    <li><a href="index.php?op=search">Search</a></li>
    <li><a href="index.php?op=login">Login</a></li>
    <li><a href="index.php?op=signup">Sign Up</a></li>
  </ol>
</div>
```

Az adatbázishoz kapcsolódunk az alábbi módon:

```
<?php
define("VARIABLE", "jrUmlbHRevghskwpLdpoxkksqe");
require("../db_config.php");
```

Deklaráljuk a változót, amelyet a navigációhoz használunk, majd teszteljük, hogy milyen értéket kapott, mert attól függően választódik ki, mely oldal jelenik meg a felhasználónak.

```
$op= "aboutus";

if(isset($_GET['op']))
    $op= mysqli_real_escape_string($connection, $_GET['op']);

switch($op)
{
    case "aboutus";
    default: include("aboutus.xhtml");
        break;

    case "contact": include("contact.xhtml");
        break;

    case "login": include("login.php");
        break;

    case "products": include("products.php");
        break;

    case "signup":
        include("signup.php");
        break;

    case "search": include("search.xhtml");
        break;

    case "search-result": include("search.php");
        break;
}
?>
```

products.php

Gyökér jelölőeleme: `<div></div>`. És a többi html elemet php kódon belül hozzuk létre. MySQL lekérdezést küldünk php-vel az adatbázishoz, mellyel lekérjük a *Products* tábla tartalmát, melyet később kilistázunk táblázatba rendezve.

```
$sql = "SELECT * FROM products ORDER BY price DESC";

$result = mysqli_query($connection, $sql) or
die(mysqli_error($connection));

echo "<table border=\"1\" align=\"center\">";
echo "<tr><th>Product
name</th><th>Image</th><th>Description</th><th>Price</th></tr>";
```

```

if(mysqli_num_rows($result)>0)
{
    while ($record = mysqli_fetch_array($result))
        echo "<tr><td>$record[productname]</td><td><img
src=\"images/$record[image]\" alt=\"$record[productname]\" width=\"100\"
/></td>
        <td>
width='400'$record[description]</td><td>$record[price]</td></tr>";
    }
else
    echo "There is no data in the database!";

echo "</table>";

```

search.php

Nem tartalmaz html jelölő elemeket, bár mikor lefut a php kód, létrehoz html kódot. Fogad egy kulcs szavat és adatbázisból kikeresi a kulcs szónak leginkább megfelelő eredményeket. Majd táblázatban kiírja.

```

<?php
//define("VARIABLE","jrUmr1bHRevghskwpLdpoxkksqe");
//require('db_config.php');

$search = mysqli_real_escape_string($connection,$_GET['search']);

echo "<p>The keyword which you searched for, was: <b>".$search."</b>. The
results: </p>";

$sql = "SELECT * FROM products
        WHERE Binary productname LIKE '%$search%'";

$result = mysqli_query($connection, $sql) or
die(mysqli_error($connection));

echo "<table border=\"1\" align=\"center\">";
echo "<tr><th>Product
name</th><th>Image</th><th>Description</th><th>Price</th></tr>";

if (mysqli_num_rows($result)>0)
{
    while ($row=mysqli_fetch_array($result,MYSQLI_BOTH))
        echo "<tr><td>".$row[1]."</td><td><img src=\"images/\".$row[2].\"\"
alt=\"\".$row[1].\"\" width=\"100\"
/></td><td>".$row[3]."</td><td>".$row[4]."</td>";
    mysqli_free_result($result);
}
else
    echo "<tr><td colspan=\"4\">There is no match for \"$search\" in the
database! </td></tr> ";

echo "</table>";
mysqli_close($connection);
?>

```

ajax.php

AJAX lekérdezést a fogad, és az adatbázisból kapcsolódva lekéri azokat az elemeket, amelyek megfelelnek leginkább a küldött adatnak. Majd kiírja, és ezt kapja meg *forajax.js* script állomány, hogy feldolgozza.

```
<?php
define("VARIABLE","jrUmlbHRevghskwpLdpoxkksqe");
require ("../db_config.php");
$keyword=mysqli_real_escape_string($connection,$_GET['keyword']);
if(empty(trim($keyword))){
    return "";
}
$sql="SELECT productname,id_product from products WHERE productname LIKE
'$keyword%'";
$res=mysqli_query($connection,$sql);
echo"<table>";
while($row=mysqli_fetch_array($res)){
    echo"<tr><td id='$row[1]' onclick='putin($row[1])'>$row[0]</td></tr>";
}
echo "</table>";
```

login.php

Tartalma egy HTML bejelentkezési űrlap, illetve egy php-n keresztül beágyazott ellenőrző php állomány. Ezenkívül tartalmaz még egy feltételt, hogy abban az esetben, amennyiben a felhasználó egyszer bejelentkezett már az adminisztrációs felületre, de valamiért elhagyta azt kijelentkezés nélkül, abban az esetben beléphet, újabb bejelentkezés nélkül.(Biztonsági rés, melyet sütik használatával ki lehetne küszöbölni.)

```
<?php
include ('verify.php'); //beagyazzuk az ellenorzo fajlt
if (isset($_SESSION['username']) != ''){
    header('Location:../admin');
}

?>
<div id="loginform" class="loginform">
    <script type="text/javascript" src="../js/functionlogin.js"></script>
    <div id="error"><?php echo $error; ?></div><br/>
    <form id="login" name="login" method="post">
        <h2>Login</h2>
        <label for="username">Username:</label><br/><input id="username"
name="username" type="text" /><br><br>
        <label for="password">Password:</label><br/><input id="password"
name="password" type="password" /><br><br>
        <input class="btn" name="loginsb" value="Login" type="submit"/>
    </form>
</div>
```

verify.php

Ez egy beágyazott fájl, melyet a bejelentkezési űrlap használ, hogy ellenőrizze: a felhasználó megtalálható-e az adatbázisban, vagyis regisztrált tag-e. Amennyiben nem az, nem

engedélyezi, hogy az adott felhasználó hozzáférjen az adminisztrációs felülethez. Ezt Session-ök alkalmazásával oldottuk meg.

```
if(mysqli_num_rows($result) == 1){
    $_SESSION['username'] = $username; //Session inicializalasa
    header("Location:../admin"); //atiranyitas egy masik oldalra
}else{
    $error = "Incorrect username or password.";
}
```

signup.php

Tartalma egy HTML bejelentkezési űrlap, illetve egy php-vel végrehajtott ellenőrzés és adatbázis-kezelés. A kód fogadja az űrlap által küldött adatokat, majd leellenőrzi, hogy a megadott felhasználónév és email cím létezik-e.

```
$sql1 = "SELECT email FROM people WHERE email='$email'";
$sql2 = "SELECT username FROM people WHERE username='$username'";
$result1 = mysqli_query($connection,$sql1); ...
$row1 = mysqli_fetch_array($result1,MYSQLI_ASSOC); ...

if(mysqli_num_rows($result1) == 1){
    $msg = "Sorry... This email already exists...";
}elseif(mysqli_num_rows($result2) == 1){
    $msg = "Sorry... This username already exists...";
}
else{
    $insql = "INSERT INTO people (username, email, password)
              VALUES ('$username','$email', '$md5pass')";
    $query = mysqli_query($connection, $insql);
    if($query){
        $msg = "Thank you! You are now registered. You can login now.";
    }
}
```

Amennyiben nem létezik, beírja az adatbázisba és innentől kezdve lehet azzal a felhasználónévvel és jelszóval bejelentkezni.

2.3.3. Adminisztrációs oldal állományai (admin mappa)

index.php

Közel teljesen egyezik a 2.3.2. címsor alatt ismertetett *index.php* állománnyal. Amiben különbözik: még mielőtt deklarálva van a HTML dokumentum tartalma az alábbi sor van feltüntetve:

```
<?php
//session_start();
define("VARIABLE", "jrUmr1bHRevghwskwpLdpoxkksqe");

include("check.php");
require("functions.php");
?>
```

A *VARIABLE* konstans akkor fontos, mikor kapcsolódni próbál az adatbázishoz az oldal. A *functions.php* beágyazása pedig akkor lesz fontos mikor pl. a törlési opció lesz kiválasztva.

Azonban a *check.php* abból a szempontból fontos, hogy ő ellenőrzi le majd, hogy a felhasználó bejelentkezett-e. De erről lesz szó a későbbiekben.

Fej jelölőelem tartalma is szintén majdnem teljesen ugyanaz, csupán különbözik a stílusban melyre hivatkozik az oldal, illetve három script állományra hivatkozik, melyből az első szintén az online elérhető jQuery könyvtár. A másik kettőről pedig szintén a későbbiekben lesz szó bővebben.

```
<link type="text/css" rel="stylesheet" href="../../css/adminstyle.css"/>
<script type="text/javascript"
src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.2/jquery.min.js"></s
cript>
<script type="text/javascript" src="../../js/jquery.js"></script>
<script type="text/javascript" src="../../js/admin.js"></script>
</head>
```

Ami szintén különbség az adminisztrációs felületen, az a navigációban található, ahol egy session segítségével üdvözölhetjük a bejelentkezett felhasználót, a nevén.

```
<li><a href="index.php?op=logout">Logout</a></li>
<li id="user" >Hello <?php echo $_SESSION['username']; ?>!</li>
```

check.php

Tehát ez az állomány az alábbi módon, Session segítségével, ellenőrzi a felhasználót, az adatbázishoz kapcsolódva és onnan kérve le az adatokat.

```
<?php
include ('../db_config.php');
session_start();

$username_check = $_SESSION['username'];
$ses_sql = mysqli_query($connection, "SELECT username FROM people WHERE
username='$_username_check'");
$row = mysqli_fetch_array($ses_sql,MYSQLI_ASSOC);

$login_user = $row['username'];
if(!isset($username_check)) {
    header("Location:../main");
}
```

Amennyiben pedig nincs a Session változóban eltárolva a felhasználó neve, abban az esetben visszairányítják az elsődleges honlap részre.

new.php

Tartalmaz egy HTML űrlapot, új termék bevitelére az adatbázisba. Kép feltöltésére is van lehetőség. Az adatokat az *add_new.php* fogadja és dolgozza fel, majd visszaküld egy üzenetet ennek az állománynak, hogy megtörtént-e az adatbevitel, vagy valami hibatörtént.

```
<div>
<h1>Add a new product</h1>
<?php
$productname="";
$description="";
$price=0;

if(isset($_GET['message']) AND $_GET['message']=="error") {
```

```

$productname=$_GET["productname"];
$description=$_GET['description'];
$price=$_GET['price'];
echo "You must fill all fields and choose an image to upload!";
}

if (isset($_GET['message']) AND $_GET['message']=="ok"){
    echo "A new product was successfully entered!";
}

?>
    <form action="add_new.php" method="post" enctype="multipart/form-data">
        <label for="productname">Name of the product: </label><input
value="<?php echo $productname; ?>" type="text" name="productname"
id="productname" /><br /><br />
        <label for="image">Image: </label><input id="image" type="file"
name="image" /><br /><br />
        <label for="description">Description: </label><br /><textarea
id="description" maxlength="100" cols="20" rows="5" name="description"
><?php echo $description; ?></textarea><br /><br />
        <label for="price">Price: </label><input type="number" name="price"
id="price" value="<?php echo $price; ?>" /><br /><br />
        <input class="btn" type="submit" name="sb" value="save"/>
        <input class="btn" type="reset" name="rb" value="cancel"/>

    </form>
</div>

```

add_new.php

Fogadja azokat az adatokat, melyeket a *new.php* küld neki, majd feldolgozza őket. Kapcsolódik az adatbázishoz és beágyazza a *functions.php*-t, melyet a képek feldolgozásához igényel.

```

define("VARIABLE", "jrUmr1bHRevghskwpLdpoxkksqe");
require("../db_config.php");
require("functions.php");

```

Megnézi, hogy volt-e bevétel a küldő oldalról, és abban az esetben, ha valamely mezőt üresen találja, visszaküld egy hibaüzenetet.

```

...
if(fileUploaded()){ $image = "set";}

if(isset($_POST['description']))
    $description = mysqli_real_escape_string($connection,
$_POST['description']);

if(isset($_POST['price']))
    $price = mysqli_real_escape_string($connection, $_POST['price']);

if(empty($productname) OR empty($image) OR empty($description) OR
empty($price)){
    header("Location:index.php?op=new&productname=$productname
&description=$description&price=$price&message=error");
    exit();
}

```

Amennyiben úgy találta, hogy minden mező ki lett töltve, és kép is lett kiválasztva eltárolja a képet, majd kicsit módosítja a nevét, hogy könnyebben eltárolhassa.

```
else{
    $file = $_FILES['image']['name'];
    $file_loc = $_FILES['image']['tmp_name'];
    $folder="./main/images/";
    $new_file_name = strtolower($file);
    $final_file = str_replace(' ', '-', $new_file_name);
```

Ekkor megpróbálja feltölteni, és ha sikerül, akkor végrehajtja a lekérdezést, mellyel feltölti az adatbázis az új adattal. És visszajelzi, hogy sikeres volt a lekérdezés.

```
if(move_uploaded_file($file_loc,$folder.$final_file)) {
    $sql = "INSERT INTO products (productname,image,description,price)
VALUES ('$productname','$final_file','$description','$price')";
    $result = mysqli_query($connection, $sql) or
die(mysqli_error($connection));
    mysqli_free_result($result);
    mysqli_close($connection);

    header("Location:index.php?op=new&productname=$productname
&description=$description&price=$price&message=ok");
    exit();
}
```

functions.php

Két függvényt tartalmaz. Az egyikkel törli az adatbázisból a kiválasztott terméket. Míg a másikkal leellenőrzi, hogy a, jelen esetben, kép fel lett-e töltve.

```
<?php
function deleteProduct($id_product){
    global $connection;
    $sql = "DELETE FROM products WHERE id_product='$id_product'";
    $result = mysqli_query($connection,$sql) or
die(mysqli_error($connection));
}

function fileUploaded()
{
    if(empty($_FILES)) {
        return false;
    }
    $file = $_FILES['image'];

    if(!file_exists($_FILES['image']['tmp_name']) ||
!is_uploaded_file($_FILES['image']['tmp_name'])) {
        return false;
    }
    return true;
}??>
```

list.php

Ez az állomány hasonlóan működik, mint a *products.php*, annyi különbséggel, hogy van egy ráadás oszlop az elején a sorszámoknak, és van egy a végén a műveleteknek (*update*, *delete*). A kódnak csupán az eltárolási része változott, így módon:

```
if (mysqli_num_rows($result)>0)
{
    $no=1;
    echo '<table align="center">';
    echo '<tr><th>No.</th><th>Product name</th><th>Product image</th><th  
class="description">Description</th><th>Price</th><th>Operation</th></tr>';
    while ($record=mysqli_fetch_array($result,MYSQLI_BOTH))
    {
        echo "<tr>";

        echo '<td>'.$no.'</td><td  
class="show">'.$record['productname'].'</td>  
<td></td>  
<td class="description">'.$record['description'].'</td>  
<td>'.$record['price'].'</td>';

        echo '<td>  
<a  
href="index.php?op=delete_product&id_product='.$record['id_product'].'"  
>delete</a> |  
<a  
href="index.php?op=update_product&id_product='.$record['id_product'].'"  
>update</a>  
</td>';
        echo "</tr>";
        $no++;
    }
    echo '</table>';
}
```

update.php

Ez egy egyszerű HTML űrlapot tartalmazó dokumentum, melynek a legördülő listáját egy php parancs segítségével hoztuk létre, az adatbázis adataiból.

```
echo '<option value="choose">choose</option>';
while ($record=mysqli_fetch_array($result,MYSQLI_BOTH))
{
    echo '<option value="'.$record['id_product'].'">  
' . $record['productname'] . '</option>';
}
```

Végrehajtva az *update_product.php* által lesz, az alábbi sor végett:

```
echo ' <input type="hidden" name="op" value="update_product" />';
```

update_product.php

Ebben az állományban hasonló űrlapot hoztunk létre, mint amilyen a *new.php* állományban található. Az űrlapot a *change_product.php* állomány dolgozza fel.

change_product.php

Mivel az *update_product.php* állomány is ad lehetőséget képek feltöltésére, ezért ebben az állományban is le kell ellenőrizni, hogy lett-e kép kiválasztva. Bár itt nincs kikötve, hogy kötelező a kép kiválasztása/cseréje. Ezért módosul az else ág, ha minden feltétel megvan az adatkód módosításához.

```
if(fileUploaded()){
//végrehajtódik a képfeltöltés és az adatbázis többi elemének módosítása is
//ahogy az az add_new.php állományban látható

}
}else{
//a képfeltöltés nem történik az adatbázison belül, csupán a többi elem
módosul.
$sql = "UPDATE products SET
productname='$productname',description='$description',price='$price'
WHERE id_product='$id_product'
";
$result = mysqli_query($connection, $sql) or
die(mysqli_error($connection));
```

delete.php

Ez az állomány közel teljesen megegyezik az *update.php* állománnyal. Csupán annyi a különbség, hogy az index oldalra, más paramétert küld vissza az *op* változóval.

```
echo '<input type="hidden" name="op" value="delete_product" />';
```

logout.php

Végrehajtja az adott Session megsemmisítését, így kiléptetve a felhasználót az adminisztrációs felületről.

```
<?php
//session start();
if(session_destroy()){
header("Location:../main");
}
```

2.4 Javascript állományok

admin.js

Csupán annyit tesz, hogy 30 másodpercenként változtatja az admin felület háttér színét.

```
setInterval(function(){ document.body.style.backgroundColor='#77081e' },
30000);
setInterval(function(){ document.body.style.backgroundColor='#4f0010' },
60000);
```

functionlogin.js

Ellenőrzi, hogy a bejelentkezésnél a mezők értéke megfelel-e egyes feltételeknek.

Ez a kód rész ellenőrzi a felhasználó név mezőjét. Először megnézi, hogy van-e értéke, majd a hosszúságát, hogy ne legyen túl rövid, se túl hosszú. Ha hosszabb a kelleténél, akkor nem tehető be az adatbázisba. Ezenkívül megnézi, hogy tartalmaz-e üres helyeket, szóközöket.

Addig pirossan fog jelezni, míg a feltételek bármelyike is igaz. Ha mind hamis, akkor az else kerül végrehajtásra és a mező színe zöldre vált. Persze csak akkor engedi tovább a felhasználót, ha az összes mező zöld, ebbe beletartozik a felhasználónév (username) és a jelszó (password).

```
if(user.value.trim() == null || user.value.trim() == ''){
    user.style.backgroundColor = "#f00";
    text += "You must enter the username!<br />";
    b_user = false;
}else if(user.value.trim().length < 4 || user.value.trim().length > 30){
    user.style.backgroundColor = "#f00";
    text += "The username must be between 4 and 30 characters.<br />";
    b_user = false;
}else{
    user.style.backgroundColor = "#0f0";
    text += "";
    b_user = true;
}
```

functions/signup.js

A functions/signup-pal is ugyan ez a helyzet, annyi különbséggel, hogy itt az e-mail címet is ellenőrzi. Megnézi, hogy a mező ki van-e töltve, a hosszúsága akkora, hogy elérjen az adatbázisban és van-e benne a megfelelő helyen at jel(@) és pont. Csak akkor kap zöld utat, ha minden feltétel hamis, így az *else* hajtódik végre és a mező zöld lesz.

```
if( email.value.trim() == null || email.value.trim() == ''){
    email.style.backgroundColor = "#f00";
    text += "You must enter an email address!<br />";
    b_email = false;
}else if(email.value.trim().length < 4 || email.value.trim().length > 35){
    email.style.backgroundColor = "#f00";
    text += "The email must be between 4 and 35 characters.<br />";
    b_email = false;
}else if(!rex.test(email.value)){
    email.style.backgroundColor = "#f00";
    text += "The email's format is incorrect.<br />";
    b_email = false;
}
else{
    email.style.backgroundColor = "#0f0";
    text += "";
    b_email = true;
}
```

slider.js

jQuery-t használ a képek megjelenítéséhez. Ez a kód rész egy automatikus diavetítést hoz létre, melyben a képek maguktól váltogatják egymást a lista szerint, ami a *main/index.php*-ben van leírva. Minden kép megjelenése és eltűnése 1 másodpercig tart, ez között pedig 3 másodpercet áll.

Az első függvényben először változókba elmentjük, hogy mennyi képről van szó, melyikekről, kell egy számláló és egy tömb is. A tömbbe be lesznek téve a képek, majd mikor a feltétel már hamis, lenullázza a számlálót és meghívja *SlideShow()* függvényt, ami egyenként meg fogja jeleníteni a képeket, így hozva létre a diavetítést.

```
$ (function() {
    var slides=$( '.slider>li' );
    var slideCount=0;
    var totalSlides=slides.length; //10
    var slideCash=[];

    (function preloader(){
        if(slideCount<totalSlides){
            slideCash[slideCount]=new Image();

slideCash[slideCount].src=slides.eq(slideCount).find('img').attr('src');
            slideCash[slideCount].onload=function(){
                slideCount++;
                preloader();
            }
        }
        else{
            slideCount=0;
            SlideShow();
        }
    })();

    function SlideShow(){
slides.eq(slideCount).fadeIn(1000).delay(3000).fadeOut(1000,function(){
        slideCount < totalSlides-1 ? slideCount ++ : slideCount = 0;
        SlideShow();
    });
    }
});
```

forajax.js

Az AJAX kérelem kezelésére hoztuk létre. Feladata, hogy a *search.xhtml* oldalon keresési ajánlatot tegyen az adatbázis segítségével.

Mikor betöltődik az ablak, akkor hívódik meg az alábbi kódrészlet, amely esemény figyelőt tesz a kereső bemeneti mezőre.

```
window.addEventListener('load', function() {
    document.getElementById('search').addEventListener('keyup', Ajax);
});
```

Amikor megtörténik a billentyűlenyomás, majd –elengedés, az alábbi függvény hívódik meg, amely lebonyolítja a háttérben az AJAX kérelmet.

```
function Ajax() {
    var keyword = document.getElementById('search').value;

    var xmlhttp;
    if (window.XMLHttpRequest) { // code for IE7+, Firefox, Chrome, Opera,
Safari
        xmlhttp = new XMLHttpRequest();
```



```

    }
    else { // code for IE6, IE5
        xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
    }
    xmlhttp.onreadystatechange = function () {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
            document.getElementById("suggest").innerHTML =
xmlhttp.responseText;
        }
    }
    xmlhttp.open("GET", "ajax.php?keyword=" + keyword, true);
    xmlhttp.send();
}

```

Így lekéri az adatbázisból, a leütött billentyű(k)nek megfelelő terméket/termékeket. Az alábbi függvény pedig lehetővé teszi, hogy ha rákattint a felhasználó a felkínált lehetőségek egyikére, akkor azt betegye a kereső mezőbe.

```

function putin(id) {
    document.getElementById('search').value =
document.getElementById(id).innerHTML;
    document.getElementById('suggest').innerHTML=" ";
}

```

jquery.js

jQuery-t hajt végre az adminisztrációs felületen.

Az alábbi kódrészlet bemutatja, hogy a *list.php* oldalon, hogyan jeleníthetjük meg illetve tüntethetjük el a leírását, ha rákattintunk a termék nevére.

//az adott lehetoseg nevere kattintva lecsusztathatjuk/felcsusztathatjuk a bovebb leirast, akar egy drop down listat

```

var jq = $.noConflict();

jq(document).ready(function() {
    jq("#show").click(function() {
        jq("#description").toggle("slow");
    });
});

```

Az alábbi kód részlet pedig azt mutatja meg, hogy az *admin/index.php* oldalon felsorolt leírások nevére kattintva lecsúsztathatjuk/felcsúsztathatjuk a bővebb leírást, akárcsak egy legördülő listát

```

jq("#new").click(function() {
    jq("#descriptionnew").slideToggle("slow");
});
jq("#list").click(function() {
    jq("#descriptionlist").slideToggle("slow");
});
jq("#update").click(function() {
    jq("#descriptionupdate").slideToggle("slow");
});
jq("#delete").click(function() {
    jq("#descriptiondelete").slideToggle("slow");
});
});

```

3. Az adatbázis struktúrája

Az adatbázisunk két táblából áll, melyek nincsenek egymással kapcsolatban (ahogy az az 1. ábrán látható is).

products	people
id_product INT (11)	id_user INT (11)
productname CHAR(30)	username CHAR(30)
image VARCHAR(100)	email VARCHAR(50)
description CHAR(100)	password CHAR(50)
price DOUBLE(5,2)	
Indexes	Indexes

1. ábra – az adatbázis tartalma: a Products és a People táblák struktúrája

A 2. és 3. ábra pedig az adatbázis tartalmát mutatja be:

id_user	username	email	password
1	AdminNiki	nikoletta9553@hotmail.com	f04e5afb5355f7243aa0b17cdfa57cd0
2	AdminValentina	valentinahodan@yahoo.com	2a3733d1a02f5ee2eaeecbd2e79ef2dc
3	test	test@test.com	098f6bcd4621d373cade4e832627b4f6

2. ábra – az adatbázis tartalma: a People tábla tartalma

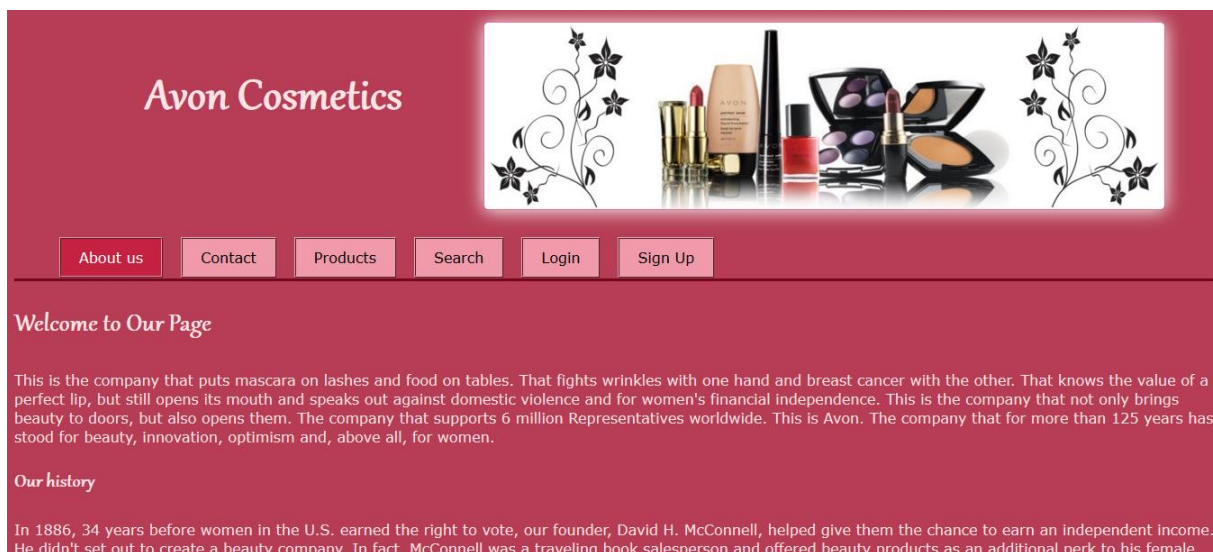
id_product	productname	image	description	price
1	Lipstick	ruzs.jpg	It is used for coloring the lips...	5.99
2	Powder	puder.jpg	It is used for smoothing the ...	10.99
3	Mascara	szempillaspiral.jpg	It is used to style the eyes, ...	6.99
4	Nailpolish	koromlakk.jpg	It is used to color and decorate ...	4.99
5	Eyeshadow	szemhejfestek.jpg	It is used to style the eyes ...	8.99
6	Lip gloss	szajfeny.jpg	It gives shine to the lips.	3.99
7	Blush	pirosito.jpg	It is used to redden the cheeks.	6.55
8	Foundation	alapozo.jpg	It is applied to the face to ...	9.98
9	Eye pencil	szemceruza.jpg	It is used to define the eyes.	3.99
10	Eyeliner	szemkonturceruza.jpg	It is used to define the eyes.	4.50
11	Corrector	korrektorstift.jpg	It is used to mask dark circles, ...	5.56

3. ábra – az adatbázis tartalma: a Products tábla tartalma

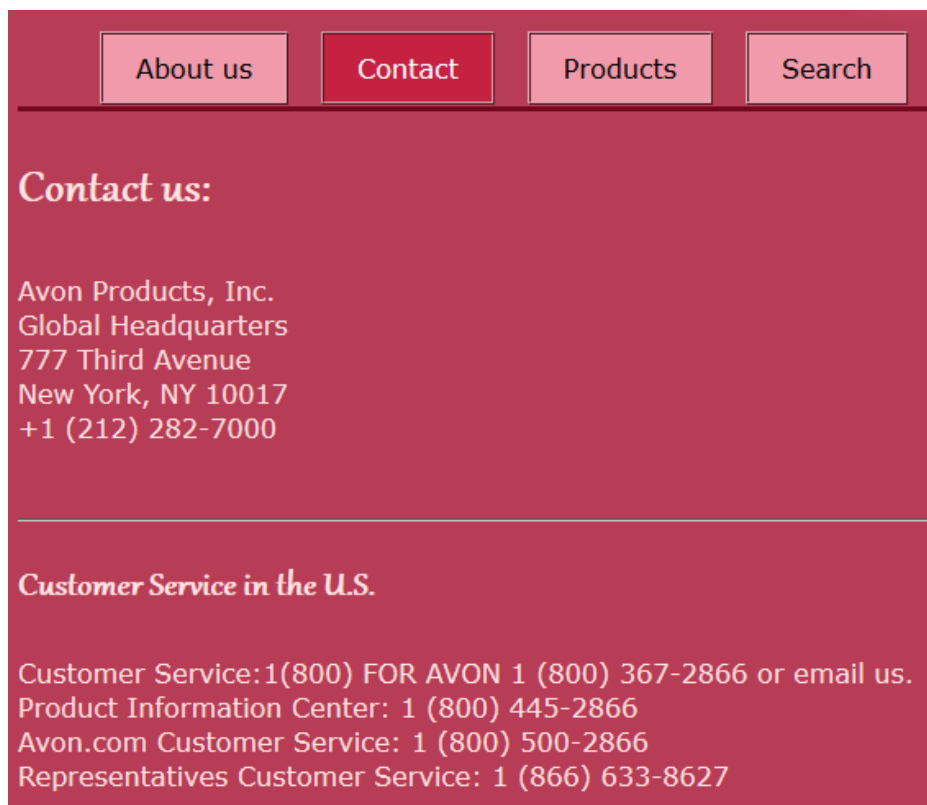
4. A projekt működésének leírása

4.1. Elsődleges felület (Vendégfogadó felület)

Az **Avon Cosmetics** kezdőoldalán (*About Us*) meglehet tekinteni a vállalat leírását és történelmét, hogy hogyan, mikor illetve miért jött létre.



A *Contact* gombra kattintva a felhasználó az elérhetőségeket és kapcsolat-felvételi lehetőségeket találhatja meg.




A *Products* gomb a cég termékeit jeleníti meg az adatbázisból egy táblázat keretén belül, ahol fel van tüntetve a termék neve, képe, leírása és ára. Mellékesen megjegyezendő, hogy a termékek az árak szerint vannak rendezve, csökkenő sorrendben.

Products


Search

Login

Sign Up

Product name	Image	Description	Price
Powder		It is used for smoothing the face, and hiding blemishes. It can be used everyday, or occasionally.	10.99
			

Ezután következik a keresés, ami a termékneve alapján történik. Mikor a felhasználó elkezd beírni a keresett árut, alatta máris megjelennek a lehetséges termékek, amelyekre rákattintva az bekerül az üres mezőbe és a *search* gombra kattintva az oldal kikeresi a keresett terméket/termékeket. Az oldal ekkor kiírja a keresőmezőbe beírt szót, ami alapján lett keresve a termék és egy táblázatba rendezi az eredményt, az esetleges találatokkal, ahol minden megtalálható róluk.

The keyword which you searched for, was: Eyeshadow . The results:			
Product name	Image	Description	Price
Eyeshadow		It is used to style the eyes by coloring the eyelids. It is good for everyday use.	8.99

Ha a felhasználó esetleg olyan szót ír be, ami nem található meg adatbázisban, akkor a táblában megjelenik a figyelmeztetés, miszerint nincs egyezés, mivel az adatbázis nem tartalmaz ilyen terméket.

Product name	Image	Description	Price
There is no match for "db" in the database!			

A következő opció a *Login*, ahol a felhasználó bejelentkezhet az oldalra azonban, ha helytelen felhasználó nevet vagy jelszót ad meg, akkor kiírja a hibát és újra lehet próbálkozni. Abban az esetben, ha egyik vagy mindkét mezőt kitöltetlenül hagyva próbálnánk bejelentkezni szintén egy hibaüzenet figyelmeztet, hogy a tovább lépés nem engedélyezett.

Navigation: Login Sign Up

Login

Username:

Password:

Login

Incorrect username or password.

Login

Username:

Password:

Login

És egyben az utolsó gomb a *Sign Up*, amely segítségével a felhasználó regisztrálhat az oldalra és a adatai bekerülnek az adatbázisba, így legközelebb már csak bejelentkeznie kell. Mikor rákattint a *Sign Up* gombra megjelenik a szöveg, melyben értesül, hogy a regisztráció sikeres volt-e.

Navigation: Login Sign Up

Sign Up

Username:

Email:

Password:

Sign Up

Thank you! You are now registered. You can login now.

Sign Up

Username:

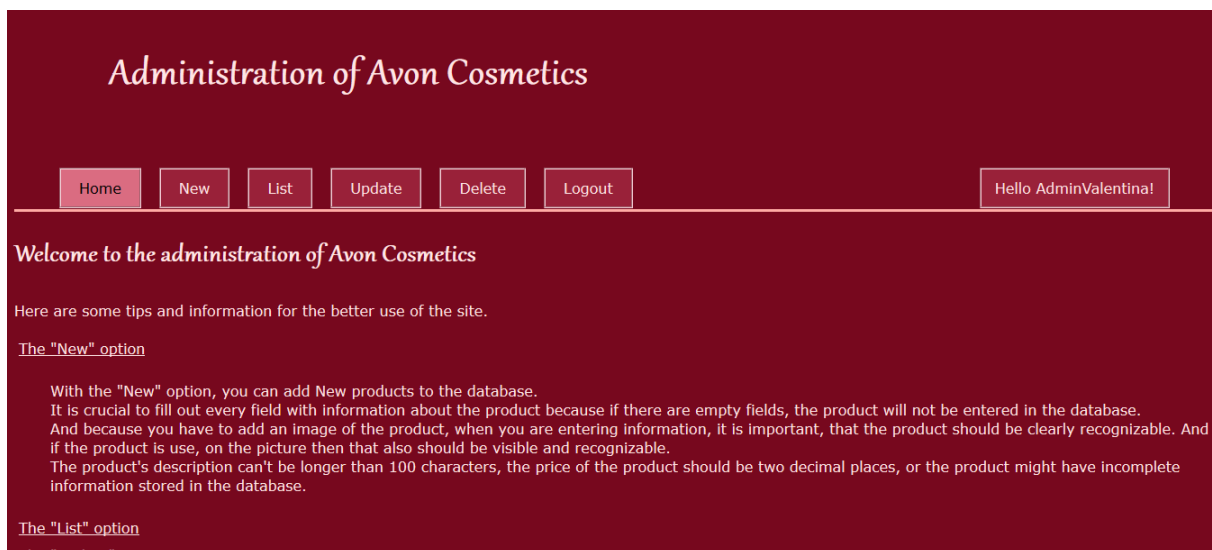
Email:

Password:

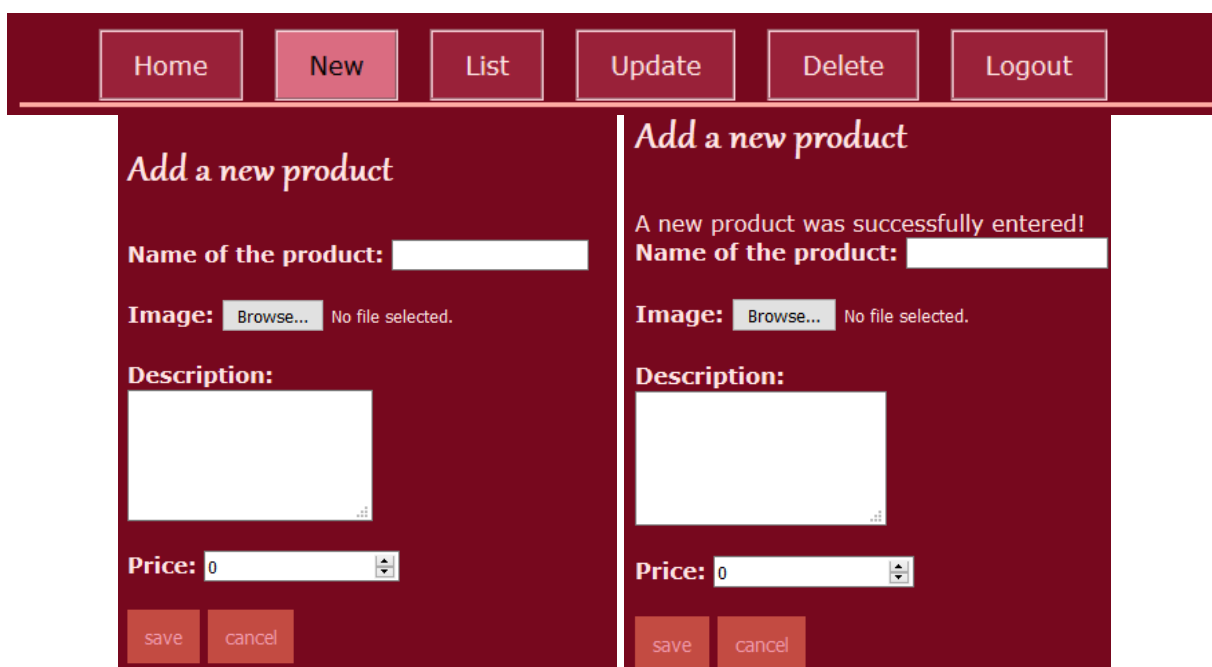
Sign Up

4.2. Adminisztrációs felület (Regisztrált felhasználóknak)



Mikor sikeresen bejelentkezett a felhasználó az **Administration of Avon Cosmetics** oldalra jut, ahol a kezdőoldalon találni lehet egy leírást miként működnek az adatbázist módosító parancsgombok, mint a New, List, Update, Delete és Logout.



A *New* gombra kattintva a felhasználó új terméket adhat a listához, mely bekerül az adatbázisba. Ha a felhasználó valamit rosszul ír be esetleg kihagy egy mezőt, addig nem kerül az adatbázisba a termék, míg mindent ki nem javított vagy ki nem töltött. A *save* gombbal történik meg az adatok elmentése. A sikeres bevitelt kiírja az oldal és már készen is áll az új adatok beírására. Ha *cancel* gombra kattint a felhasználó, akkor az addig kitöltött mezők törlődnek.



A következő opció a *List*, ahol megjelenik az adatbázisban lévő összes termék. Ez esetben egy plusz oszloppal, melyben megváltoztatni vagy törölni lehet a terméket, csak rá kell kattintani. Abban az esetben viszont, hogy ha bármelyik termék nevére kattintunk, előhívhatjuk, majd újra elrejtethetjük a termék leírásának oszlopát.

Home	New	List	Update	Delete	Logout
No.	Product name	Product image	Price	Operation	
1.	Blush		1.99	delete update	
2.	Corrector		5.99	delete update	

Az *Update* gombra kattintva, akár csak a *List* opció keretén belül az *update* paranccsal, meg lehet változtatni a termék adatait. Csak ki kell választani egy legördülő menüből a változtatni kívánt terméket. Ekkor megjelennek azok az adatok, amelyeket meg lehet változtatni. Megnyomva *change* gombot elmenti a változást és kiírja, hogy a termék sikeresen meg lett változtatva.

Home	New	List	Update	Delete	Logout
------	-----	------	--------	--------	--------

List of products - for update

choose product:

choose

choose

Blush

Corrector

Eye pencil

Eyeliner

Eyeshadow

Foundation

Lip gloss

Lipstick

Mascara

Nailpolish


Powder

send

Update product

Name of a product: Blush

Image: No file selected.

Current image: 

Description: It is used to redden the cheeks.


Price: 1.99

Update product

Product was successfully updated!

Name of a product: Blush with mirror

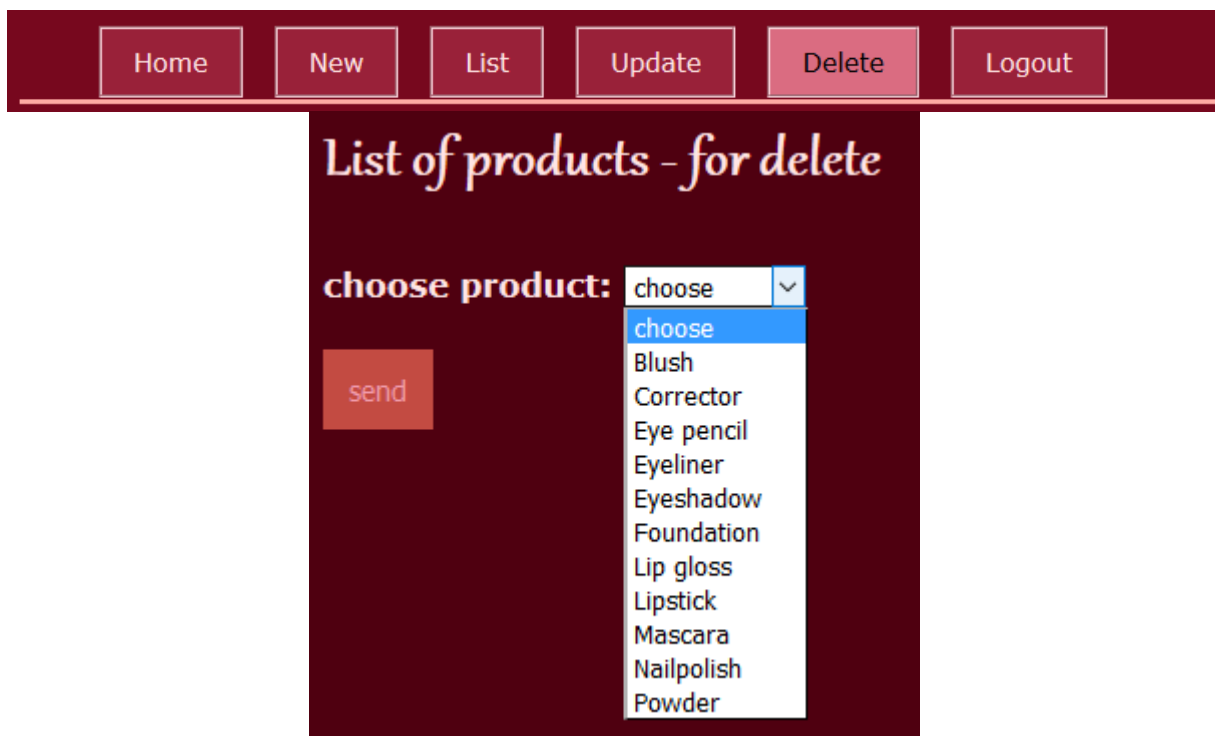
Image: No file selected.

Current image: 

Description: It's used to redden the cheeks.

Price: 5.00

A *Delete* gombra kattintva megjelenik egy olyan lehetőség, mint az *Update* gomb alatt, hogy legördülő menüből választhatjuk ki a törölni kívánt terméket. A törlés azonnali és végleges így csak gondos odafigyeléssel ajánlott alkalmazni ezt az opciót, különben csak abban az esetben lesz a véletlenül törölt adat újra az adatbázisban, ha *New* opcióban újra bevisszük az adatait. A törlés után a felhasználó visszakérül a termékek listájához. Ez igaz a *List* opcióban is előforduló *delete*-re is.



Ha a *Logout* gombra kattint a felhasználó, akkor kijelentkezik az adminisztrációs felületről és az Avon Cosmetics kezdőoldalára kerül.



Melléklet

CD

Felhasznált irodalom

[W1] File Upload Check - <http://stackoverflow.com/questions/946418/how-to-check-if-user-uploaded-a-file-in-php>

[W2] jQuery - http://www.w3schools.com/jquery/jquery_slide.asp

[W3] PHP AJAX Live Search - http://www.w3schools.com/php/php_ajax_livesearch.asp