Tampereen ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma
Digimedia / Ohjelmistotuotanto
Jari Ketolainen

Opinnäytetyö

Verkkokaupan arkkitehtuuri ja toteutus

Case: Mango Hotel

Työn ohjaaja Lehtori Harri Hakonen
Työn tilaaja Oy Anivik Ltd, Mango Hotel, Tampere
Tampere 04/2010

Tampereen ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma, Digimedia / Ohjelmistotuotanto

Tekijä	Jari Ketolainen
Työn nimi	Verkkokaupan arkkitehtuuri ja toteutus Case: Mango Hotel
Sivumäärä	56
Valmistumisaika	Huhtikuu 2010
Työn ohjaaja	Lehtori Harri Hakonen
Työn tilaaja	Oy Anivik Ltd, Mango Hotel, Tampere

TIIVISTELMÄ

Opinnäytetyön toimeksiantajana toimi tamperelainen hotellialan yritys Mango Hotel. Yrityksellä oli ollut käytössä valmis verkkokaupparatkaisu vaatteiden, tavaroiden ja asusteiden myyntiin, mutta sovellus osoittautui vaikeakäyttöiseksi, huonosti Mango Hotellin tarpeisiin mukautuvaksi ja sitä oli ylläpitäjän hankala päivittää.

Opinnäytetyön tarkoituksena oli suunnitella ja toteuttaa aivan uusi verkkokauppasovellus Mango Hotellin käyttöön käyttämättä tai muokkaamatta mitään valmista verkkokaupparatkaisua. Näin verkkokaupasta saataisiin toimeksiantajan toiveiden ja vaatimusten mukainen, sekä ylläpitäjälle helposti hallittava.

Verkkokauppaan toteutettiin asiakkaille ja ylläpitäjälle erilliset sovellukset. Verkkokauppasovelluksessa asiakas voi selata tuotteita ja tehdä tilauksia. Hallintasovelluksessa ylläpitäjä voi hallita verkkokaupan tuotteita ja tuotekategorioita, sekä tarkastella asiakkaiden tekemiä tilauksia ja muokata verkkokaupan asetuksia.

Työ toteutettiin käyttämällä PHP-ohjelmointiympäristöä sekä MySQL-tietokantaa. Myös JavaScript-komentosarjakieli ja sen ohjelmointikirjasto jQuery olivat hyvin tärkeässä osassa verkkokaupan kehityksessä.

Opinnäytetyössä käydään läpi työssä käytetyt tekniikat, verkkokaupan käyttämän relaatiotietokannan suunnittelu ja toteutus, sekä miten itse verkkokauppasovellus ja ylläpitäjän hallintasovellus toimivat.

Lopputuloksena toteutettua verkkokauppajärjestelmää tullaan vielä kehittämään eteenpäin tulevaisuudessa. Tällä hetkellä se kuitenkin tarjoaa helpon ja nopean ratkaisun asiakkaille tuotetilausten tekemiseen, sekä ylläpitäjälle tuotteiden ja tilausten hallitsemiseen.

Avainsanat: verkkokauppa, sisällönhallinta, tietokanta, PHP, MySQL, JavaScript, jQuery

TAMK University of Applied Sciences
Business Information Systems, Digital Media / Software Development

Writer	Jari Ketolainen
Thesis	E-Commerce Architecture and Implementation Case: Mango Hotel
Pages	56
Graduation time	April 2010
Thesis Supervisor	Lecturer Harri Hakonen
Co-operating Company	Oy Anivik Ltd, Mango Hotel, Tampere

ABSTRACT

This thesis is based on a commission by Mango Hotel company. The company had been using a predefined web store application for selling clothes, items and accessories but the application had turned out to be difficult to use, it did not adapt well enough to serve Mango Hotel's needs and it was troublesome to update by the web store admin.

The main purpose of this thesis was to design and implement a whole new web store for Mango Hotel without using or modifying any already defined web store solution. By using this method, the web store would carry out Mango Hotel's expectations and requirements and it would be easy to manage for admin.

Standalone applications were implemented for customers and admin. In the web store, a customer can browse products and make orders. Admin can manage products and product categories in the management application. In the management application, the admin can also view customers' orders and modify the web store's settings.

The task was implemented by using PHP programming and MySQL database. JavaScript scripting language and JavaScript jQuery programming library were also in a very important role when the web store was developed.

In this thesis I examine the technologies used, web store's relation database design and implementation and how the customer's and admin's standalone applications work.

The created application will be developed in future. Right now it offers an easy and a fast solution for the customers to make orders and for the admin to manage products and orders.

Keywords: web store, e-commerce, content management, data base, PHP, MySQL, JavaScript, jQuery

Sisällysluettelo

1 Johdanto

Verkko-ostaminen on kasvanut huomattavasti koko 2000-luvun ajan Internetin yleistymisen vuoksi, ja se on entistä tutumpi asia myös suomalaiselle kuluttajalle. Smilehouse Oy:n (2010) tekemän tutkimuksen mukaan suomalaiset tekivät taantumavuodesta huolimatta entistä enemmän verkko-ostoksia. Nykyään ostokset on helppo tehdä verkossa ja ostetut tuotteet saadaan tilattua kotiovelle asti ilman järin suuria postimaksuja. Tämä auttaa esimerkiksi haja-asutusalueiden ihmisiä, joilla lähimpään kauppaan saattaa olla matkaa useita kymmeniä kilometrejä.

Verkosta on mahdollista tilata ruokia, juomia, vaatteita ja tavaroita, eivätkä tuotehinnat välttämättä nouse normaalien päivittäistavarakauppojen asettamien hintojen yli, vaan päinvastoin ne saattavat olla jopa huomattavasti halvempia. Verkkokaupan kasvun ohella myös kuluttajien vaatimukset kaupoille ovat kiristyneet, sillä rahan käytössä ollaan entistä varovaisempia.

Hyvin toimiva verkkokauppa säästää aikaa ja vaivaa niin asiakkaalta kuin kauppiaaltakin. Asiakas pystyy tilaamaan tuotteet vaivattomasti miltä tahansa päätteeltä ja mihin vuorokaudenaikaan tahansa. Myyjä käsittelee verkkokauppaa ja sen tuotteita erillisen hallintasivun kautta ja tarkastelee asiakkaiden tilauksia. Kummassakin tapauksessa työ tapahtuu nopeasti, helposti ja resursseja säästämällä, vaikkapa kotikoneelta.

Opinnäytetyöni toimeksiantaja on Tampereen Hatanpäällä sijaitseva hotellialan yritys Mango Hotel ja yhteyshenkilönä hotellin yksi perustajajäsen Saini Ranjit. Mango Hotelli on valmistunut vuonna 2005 ja se sisältää 26 huonetta. Yrityksen omien sanojen mukaan hotellissa yhdistyvät itämainen tunnelma, viihtyvyys ja kustannuksiltaan edullinen hotelliyö. Hotelli toimii itsepalveluperiaatteella ja hotellihuonehuonevarauksia voidaan tehdä heidän kotisivuiltaan osoitteesta www.mangohotel.fi. Sivujen kautta asiakkaat pääsevät myös hotellin verkkokauppaan tekemään ostoksia.

Mango Hotel käytti ennen tämän työn valmistumista valmista verkkokauppapohjaa vaatteidensa, tavaroidensa ja asusteidensa myyntiin Internetissä. Heidän verkkokauppansa sisältää kymmeniä erilaisia tuotteita, ja esimerkiksi vaatteilla ja asusteilla on hyvin useita erilaisia koko- ja värivaihtoehtoja. Mango Hotel kuitenkin totesi käytössä olleesta verkkokaupasta, että verkkokauppapohjan ulkoasu ei

vastannut yrityksen vaatimuksia, se oli vaikeasti käytettävä, eikä sivujen sisältöä pystynyt muokkaamaan helposti.

Tavoitteenani oli suunnitella ja toteuttaa Mango Hotellille uusi Internetissä toimiva verkkokauppa, joka vastaisi yrityksen toiveita. Verkkokauppa päätettiin toteuttaa alusta alkaen itse käyttämättä tai muokkaamatta mitään valmista verkkokauppasovelluksen pohjaa. Tällä varmistettiin, että sivujen ulkoasu vastaisi mahdollisimman hyvin toimeksiantajan toiveita ja että sivujen käyttäminen ja päivittäminen olisi nopeaa ja helppoa.

Verkkokauppaan toteutettiin erilliset sivut asiakkaille ja sivujen ylläpitäjille. Asiakassivuilla pystytään muun muassa selaamaan tuotekategorioita ja tuotteita sekä tekemään tilauksia. Erillisellä hallintasivulla sivujen ylläpitäjä voi lisätä, muokata ja poistaa tuotekategorioita sekä tuotteita, tarkastella asiakkaiden tilauksia, muokata kaupan tietoja ja lisätä sivuille uusia käyttäjiä ylläpitäjiksi.

Mango Hotellilla entuudestaan olleeseen MySQL-tietokantaan lisättiin omat taulut muun muassa kaupan tuotekategorioille, tuotteille ja tilauksille. Sivujen ohjelmointikielenä käytettiin PHP-ohjelmointikieltä, joka soveltuu dynaamisten websivujen toteuttamiseen erinomaisesti. Asiakas- ja ylläpitäjäsivujen rakenne toteutettiin W3C:n standardien mukaisella XHTML-merkintäkielellä sekä CSS-tyyliohjeilla. Sivujen dynaamisen toiminnallisuuden lisäämiseksi käytettiin JavaScript-komentosarjakieltä sekä sen jQuery-kirjastoa.

Opinnäytetyöni alussa käyn läpi verkkokauppasovelluksessa käytetyt tekniikat. Kolmannessa luvussa kerron miten verkkokauppa käyttää relaatiotietokantaa hyödykseen ja miten sen tietokantataulut ovat yhteydessä toisiinsa. Neljännessä ja viidennessä luvussa käyn läpi yksityiskohtaisesti, miten verkkokaupan asiakas- ja ylläpitäjäsivut on rakennettu ja miten ne toimivat. Työn lopussa pohdin, miten olen onnistunut työssäni ja miten verkkokauppaa voi kehittää tulevaisuudessa.

2 Käytetyt tekniikat

2.1 (X)HTML

HTML (Hypertext Markup Language) on avoimesti standardoitu kuvauskieli, josta erityisesti web-sivut rakentuvat, ja jota jokainen selainohjelma osaa tulkita omien tapojensa mukaan (Linjama 2001, 12).

XHTML (eXtensible Hypertext Markup Language) HTML:stä laajennettu ja kehitetty versio, joka noudattelee tarkasti XML-kielen (eXtensible Markup Language) rakenteellista dokumentin merkintätapaa (Linjama 2001, 9). XHTML-kielen määritelmiä ylläpitää kansainvälinen yritysten ja yhteisöjen yhteenliittymä W3C (World Wide Web Consortium). Koska W3C ylläpitää ja kehittää WWW-standardeja, on sen tarjoamia online-dokumentteja käytetty hyvin paljon apuna ja lähteinä tässä työssä. W3C:n dokumentteja voidaan pitää ajankohtaisina ja luotettavina.

XHTML eroaa HTML:stä sen tiukemmilla muotoilusäännöillä. Eroja ovat muun muassa pienten kirjaimien pakotettu käyttö kaikissa tunnisteissa, elementtien attribuuttien pakolliset arvot ja niiden merkitseminen lainausmerkeillä, sekä elementtien pakollinen sulkeminen. Dokumentissa tulee olla myös viittaus käytettyyn XML-määrittelyyn. (XHTML 2009.)

Kuten Linjama (2001, 130) toteaa, tulisi HTML-dokumenteissa aina käyttää XHTML-määrityksiä. Tämä siksi, että nykyään web-sivuja on mahdollista tarkastella myös muilla laitteilla, kuin pelkästään perinteisellä tietokoneella. Tästä syystä on erityisen tärkeää, että HTML-dokumentit sisältävät standardia XML-merkintätapaa ja metatietoja sekä ovat laajennettavissa erilaisin toiminnallisin moduulein. Tämä varmistaa sen, että sivut näkyvät juuri niin kuin niiden tekijä on halunnut, riippumatta laitteesta, jolla niitä selataan.

Olen samaa mieltä Linjaman kanssa XHTML-määrityksistä ja niiden käytöstä, sillä tämäkin verkkokauppa tuli suunnitella toimimaan kaikilla eri selaimilla ja päätteillä, ja sen varmistamiseksi käytin työssä juuri XHTML-määrityksiä. Linjama ei kuitenkaan mainitse, että aina eivät edes standardit XHTML-määritykset auta sivuja näkymään oikein. Hyvin vanhoilla selaimilla on erittäin huono tuki uusille standardeille WWW-määrityksille, mistä johtuen sivut saattavat näyttää hieman erilaisilta, kuin uusimmilla selaimilla. Vanhojen selaimien käyttö on kuitenkin koko ajan

ollut laskussa, kuten seuraavasta W3C:n ylläpitämästä tilastosta voidaan huomata (Taulukko 1).

2010	IE8	IE7	IE6	Firefox	Chrome	Safari	Opera
February	14.7%	11.0%	9.6%	46.5%	11.6%	3.8%	2.1%
January	14.3%	11.7%	10.2%	46.3%	10.8%	3.7%	2.2%

Taulukko 1. W3C:n tilasto suosituimmista web-selaimista alkuvuonna 2010 (W3 Schools 2010a).

XHTML-dokumentit tulisi aina validoida W3C:n kehittämässä merkistön tarkastuspalvelussa, millä varmistetaan virheetön XHTML-rakenne ja sivujen toimiminen oikein eri selaimilla ja päätteillä. Myös tämän työn XHTML-dokumentit on tarkastettu kyseisellä tarkastuspalvelulla. Validointi on mahdollista suorittaa ilmaiseksi Internetissä osoitteessa http://validator.w3.org.

XHTML-kielen erottaa lähdekoodista

Lähdekoodia (Koodiesimerkki 1) tarkastelemalla XHTML-dokumentin tunnistaa heti alussa määritellyllä XML-esittelyllä, joka kertoo, mitä XML-versiota dokumentti käyttää. Tämän jälkeen on esitelty käytettävä dokumenttityyppi (DTD, Document Type Definition). Dokumenttityyppi määrittelee, miten elementit ja attribuutit voivat ilmetä dokumentissa. Varsinainen HTML-osuus alkaa <html>-elementistä, jolle on annettu attribuutiksi dokumentin käyttämä XML-nimiavaruus. (W3C XHTML, 2002.)

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title> Web-sivu esimerkki </title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <link href="css/styles.css" type="text/css" media="screen" />
</head>
<body>
</body>
</html>
```

Koodiesimerkki 1. XHTML-dokumentin lähdekoodi.

Ennen käyttäjälle näkyvää osaa on määritelty <head>-elementti ja tämän osion sisään sivun otsikko <title>. Näiden kahden elementin jälkeen on <meta>-elementille annettu attribuutit, jotka kertovat tiedon dokumentin sisältämistä meta-

tiedoista ja sen käyttämästä merkistökoodauksesta. Meta-elementteihin voidaan myös määritellä eri tietoja esimerkiksi hakukoneita varten.

Ennen <head>-elementin sulkemista on määritelty dokumentin käyttämä erillinen tyylitiedosto, josta lisää seuraavassa alaluvussa. Varsinainen käyttäjälle näkyvä osuus aloitetaan <body>-elementistä.

2.2 CSS

CSS (Cascading Style Sheets) on tyyliohjejärjestelmä, joka on kehitetty WWW-dokumenteille. Sen yleisin käyttötarkoitus on määritellä ulkoasu ja sen esitystapa web-sivuille, jotka on kirjoitettu HTML- ja XHTML–kielillä, mutta sitä voidaan myös hyödyntää missä tahansa XML-kieltä sisältävässä dokumentissa (Cascading Style Sheets 2010.)

CSS-tyyli pystytään yhdistämään haluttuun HTML-dokumenttiin kolmella eri tapaa. Ensimmäinen tapa on kirjoittaa tyylisääntö suoraan haluttuun HTML-elementtiin style-parametrilla. Esimerkiksi asemointilohkon muodostavan div-elementin taustaväri vaihdetaan seuraavalla tavalla: <div style=”background-color: #000000;”>. Tämä on kuitenkin hankala ja hidas tapa, sillä sääntö pitää kirjoittaa jokaiseen elementtiin erikseen, mikä muuttaa dokumentin päivittämisen hankalaksi.

Toinen tapa on sisällyttää tyylisäännöt HTML-dokumentin <head>-osaan <style>-elementin sisään. Tämä tapa toimii kuitenkin vain niissä dokumenteissa, joihin tyylisäännöt on sisällytetty, ja jos dokumentteja on useita, hankaloituu päivittäminen jälleen.

Kolmas tapa on käyttää erillistä CSS-tyylitiedostoa, jota on myös tässä työssä käytetty. Erillinen tyylitiedosto liitetään HTML-dokumentin <head>-osioon <link>-elementin sisään. Erillisen tyylitiedoston käyttäminen on suositeltavaa, sillä tällöin tyylit toimivat jokaisessa HTML-dokumentissa ja dokumenttien päivittäminen on hyvin yksinkertaista ja nopeaa.

Tyylitiedostossa voidaan määritellä kaikille elementeille samat tyylimääritykset tai yksilöidä määritykset lisäämällä HTML-elementtiin id- tai class-attribuutti, esimerkiksi . Id-attribuutti on uniikki määritys ja sille määri-

telty tyyli esiintyy vain kyseisen id:n omaavassa elementissä, kun taas class-attribuuttia voidaan käyttää useammissa toistuvissa elementeissä. Uniikille -elementille voidaan esimerkiksi määritellä taustakuva, joka toistuu automaattisesti y-akselin suuntaisesti (Koodiesimerkki 2).

```
#content_text
{
    width: 210px;
    height: auto;
    background-image: url(../images/background.jpg);
    background-repeat: repeat-y;
}
```

Koodiesimerkki 2. Uniikin -elementin tyylimääritykset CSS-tiedostossa.

2.3 PHP

PHP (Hypertext Preprocessor) on vuonna 1995 syntynyt dynaamisten web-sivujen luontiin tarkoitettu ohjelmointikieli. PHP on avoimeen lähdekoodiin perustuva komentosarjakieli ja se ajetaan aina, kun WWW-palvelin lähettää web-sivun selaimelle. Nykyisin PHP on johtavassa asemassa dynaamisten web-sivujen luontiin tarkoitetuissa ohjelmointikielissä (PHP 2009).

Kuten Heinisuo ja Rauta (2007, 13) toteavat, PHP-koodi suoritetaan aina palvelimen puolella, ja näin ollen sitä ei pystytä ajamaan esimerkiksi suoraan tietokoneen levyltä. Tästä syystä PHP-koodia sisältävät sivut näyttävät selaimessa vain itse ohjelman tulostuksen eivätkä sen lähdekoodia.

PHP-kielestä on kehitteillä versio 6, jonka julkaisuajankohtaa ei ole määritelty. Tällä hetkellä uusin versio on PHP5, jota on myös tässä työssä käytetty.

PHP5-versio toi mukanaan uusia parannuksia olemassa oleviin toiminnallisuuksiin sekä piirteitä, jotka ovat yleensä yhdistetty oliopohjaisiin ja kypsiin ohjelmointikieliin. Versio viisi sisältää muun muassa huomattavasti versio neljästä kehittyneemmät olio-ohjelmointipohjaiset ominaisuudet, poikkeusten hallinnan, kehittyneemmän merkkijonojen hallinnan, sekä SimpleXML-laajennuksen XML-määrittelyiden jäsentämiseen ja käsittelyyn. (Gilmore 2005, 4.)

PHP sisältää kaikki ohjelmointikielille tyypilliset rakenteet, kuten esimerkiksi muuttujat, funktiot, for- ja while-silmukat, if-ehtolauseet sekä olio-ohjelmointiin kuuluvat luokat. Se sisältää myös paljon valmiita funktioita, joilla esimerkiksi tietokantojen käsittely onnistuu hyvin yksinkertaisesti. PHP on myös laajennettavissa erilaisten ohjelmakirjastojen avulla. (Heinisuo ym. (2007, 13.)

Tässä työssä ei ole käytetty PHP:n oliopohjaista ohjelmointia, vaan verkkokauppasovellus on toteutettu proseduraalisella ohjelmoinnilla. Tämä siksi, että työn tekijällä ei ollut hirveästi kokemusta oliopohjaisesta ohjelmoinnista, jota oltaisiin voitu hyödyntää verkkokauppasovelluksen toteuttamisessa.

PHP-ohjelmointikielessä koodi voidaan kirjoittaa suoraan HTML-dokumenttien sisään mihin tahansa kohtaan HTML-koodia. Jos koodissa on virheitä, lopettaa selain sivun avaamisen siihen kohtaan dokumenttia, missä virhe ilmenee. PHP-koodi aloitetaan aina <?php–merkinnällä ja lopetetaan ?>-merkintään (Koodiesimerkki 3).

```
<body>
<div>
<?php
// Funktio
function isExampleString($example)
{
    $stringEsim = $example;
    if (is_string($stringEsim)) {
        echo "Esimerkki on String";
        return TRUE;
    } else {
        echo "Esimerkki ei ole String";
        return FALSE;
    }
}
// Funktion ajo
isExampleString($testi = "Testataan");
?>
</div>
</body>
```

Koodiesimerkki 3. Esimerkki PHP-ohjelmoinnista. Esimerkissä tarkastetaan PHP:n valmiilla funktiolla is_string(), että onko funktiolle syötettävä muuttuja merkkijono.

2.3.1 Evästeet

Eväste (cookie) on pieni pala tietoa, jonka web-selain tallentaa verkkopalvelun käyttäjän tietokoneelle. Ne ovat nimettyjä ja niihin voidaan tallentaa hyvin pieni määrä tekstimuotoista dataa. Evästeiden käytössä on omat haittapuolensa, sillä käyttäjä voi kytkeä selaimesta evästetuen pois päältä tai poistaa omatoimisesti evästeet tietokoneelta. (Heinisuo ym. (2007, 271.)

Koska evästeet ovat aina yhteydessä yhteen palveluun ja niillä on tietty voimassaoloaika, voidaan niihin tallentaa käyttäjätietoja joko tilapäisesti kyseessä olevan istunnon ajaksi tai pidemmäksi ajaksi, jolloin evästettä voidaan käyttää uudelleen myöhemmässä istunnossa (Gilmore 2005, 382).

Evästeet luodaan ja poistetaan PHP:n setcookie()-funktiolla ja niiden tietoja voidaan tallentaa erillisiin muuttujiin käsittelyä varten (Koodiesimerkki 4).

```php
<?php
// Luo evästeen nimi, jonka arvo on Jari
setcookie("nimi", "Jari");
// Tämä eväste on voimassa seuraavat 10 minuuttia
setcookie("auto", "BMW", time() + 60 * 10);

// Haetaan evästeen tieto muuttujaan
$auto = $_COOKIE["auto"];
// Tulostetaan evästeen tieto
echo "{$nimi}n auto on $auto";

// Poistetaan evästeet
setcookie("nimi", "");
// Aika asetetaan menneisyyteen, jotta se toimii
// kaikissa selaimissa
setcookie("auto", "", time() - 60 * 10);
?>
```

Koodiesimerkki 4. Luodaan evästeet “nimi” ja “auto”, joille annetaan arvoksi “Jari” ja “BMW”.

2.3.2 Istunnot

Istuntojen käyttäminen perustuu siihen, että web-sivun käyttäjälle luodaan yksilöllinen ID-tunnus (Identifier), joka tallennetaan evästeeseen tai kuljetetaan URL:n (Uniform Resource Identifier) mukana. ID-tunnus tunnetaan yleisesti SID-tunnuksena (Session Identifier) (Gilmore 2005, 382). ID-tunnusta vastaavat tie-

dostot, jotka sisältävät tiedot istunnossa käytetyistä muuttujista, tallentuvat palvelimelle.

Jos SID-tunnus on tallennettu evästeeseen, haetaan se aina tarvittaessa käyttäjän selatessa sivua, ja sen mukana voidaan toimittaa muita tekstialkioita yhdistettynä kyseiseen istuntotunnukseen. Tällöin käyttäjän tietoja ei tarvitse tallentaa hänen käyttämälleen tietokoneelle, vaan ne voidaan noutaa esimerkiksi erillisestä tietokannasta määritellyn SID-tunnuksen perusteella. (Gilmore 2005, 382.)

SID-tunnus on myös mahdollista kuljettaa URL-osoitteen mukana, jolloin sen kulkeutuminen palvelussa automaattisesti sivulta toiselle onnistuu huolimatta siitä, onko käyttäjä estänyt selaimestaan evästeiden käytön. Tätä tapaa kutsutaan URL-osoitteiden uudelleenkirjoituksena. Tässä tekniikassa on myös omat haittapuolensa, sillä jos käyttäjä poistuu verkkosivulta, ei SID-tunnuksen lisääminen URL-osoitteeseen jatku. (Gilmore 2005, 382.)

Istunto luodaan tai sitä jatketaan PHP:ssa session_start()-funktion avulla. Istunnon muuttujia käsitellään ja luetaan $_SESSION-taulukon kautta. Istuntomuuttujat poistetaan unset()-funktiolla ja koko istunto tuhotaan session_destroy()-funktiolla (Koodiesimerkki 5).

```php
<?php
// Aloitetaan istunto
session_start();
// Määritetään istuntomuuttujat
$_SESSION["nimi"] = "Jari";
$_SESSION["auto"] = "BMW";

// Luetaan istuntomuuttujat
$nimi = $_SESSION["nimi"];
$auto = $_SESSION["auto"];
// Tulostetaan tiedot
echo "{$nimi}n auto on $auto";

// Poistetaan istuntomuuttujat
unset($_SESSION["nimi"]);
unset($_SESSION["auto"]);
// Tuhotaan sessio
session_destroy();
?>
```

Koodiesimerkki 5. Luodaan istunto ja määritellään istuntomuuttujat "Jari" ja "BMW".

2.4 Relaatiotietokanta

Tietokanta tulkitaan relaatiotietokannaksi silloin, kun siihen on liitettynä yksi tai useampi relaatio. Relaatiotietokannat ovat erittäin hyviä suurten, luettelomaisten tietomäärien varastoinnissa sekä ylläpitämisessä. Tietokantoihin voidaan tehdä lisäys-, poisto-, päivitys- ja hakutoimintoja, mutta myös laajennettujen tietokanta-operaatioiden käyttö on mahdollista relaatiotietokannan ositusten ja relaatioiden ansiosta.

Relaatiotietokannassa relaatioksi kutsutut tiedot esitetään tauluina (table). Taulun yhtä riviä kutsutaan tietueeksi (record) ja taulun jokaisella rivillä on myös yhtä monta kenttää (field), jotka muodostavat tietueiden attribuutit. Tietueiden kenttiin liitetään kyseisen attribuutin arvoalue sekä arvon puuttumisen symboli (Ekonoja, Lahtonen & Mäntylä 2004). Jokaisella rivillä täytyy olla uniikki perusavain, joka vastaa jotakin reaalimaailman kohdetta.

Relaatiotietokannasta voidaan hakea mikä tahansa yksittäinen tieto ilmoittamalla taulun nimi, perusavaimen kentän nimi ja avaimen arvo, sekä haettavan tiedon kentän nimi. Tietokannoista voidaan noutaa tietoa myös lukemattomilla muilla tavoilla. Relaatiotietokannoista tieto haetaan tiedon nimien ja arvojen perusteella, eikä koskaan sen sijainnin tai järjestyksen mukaan. (Ekonoja ym. 2004.)

Johtuen jatkuvasta ohjelmistokäyttöliittymien ja tietokantojen kyselykielien kehityksestä, ovat relaatiotietokannat nykyään liitettävissä joustavasti lähes minkä tahansa sovellusohjelmatyypin kanssa.

Tämän työn verkkokaupassa on käytetty relaatiotietokantaa kaupan tarvitsemien tietojen tallentamiseen. Tietokannasta haetaan tietoja verkkokaupan käyttöön SQL-kyselykielellä (Structured Query Language) ja kieltä hallitaan seuraavassa kappaleessa esiteltävällä MySQL-hallintajärjestelmällä.

2.5 MySQL

MySQL:n kehittäjänä toimii ruotsalainen MySQL AB. Se on saatavissa vapaalla ohjelmistolisenssillä ja on nykyään ylivoimaisesti eniten käytetty tietokantojen hallintajärjestelmä, jota on myös tässä työssä käytetty (Heinisuo ym. (2007, 38).

MySQL on helppo asentaa ja sen käyttöönotto onnistuu ilman erityisiä tietoja ja taitoja tietokannoista. Hallintajärjestelmästä löytyy muun muassa monipuolinen komentokanta, monen käyttäjän yhtäaikainen tuki sekä aidosti monisäikeinen toteutus. Kuten myös Heinisuo ja Rauta (2007, 39) toteavat, mielestäni MySQL on erittäin hyvä vaihtoehto erilaisten WWW-palvelujen tietokannaksi, sillä se on ilmainen ja yksinkertainen käyttää. Sen käyttöön on myös saatavilla monipuolinen ohjelmistodokumentaatio, sekä hyvin paljon muita hyödyllisiä lisäominaisuuksia.

PHP sisältää oman ohjelmointirajapinnan (API, Application Programming Interface) MySQL:lle, jota on myös tässä työssä käytetty. Rajapinta tarjoaa yli 45 eri funktiota ja sen avulla tietoja voidaan noutaa, lisätä, päivittää ja poistaa tietokannasta, sekä tehdä erilaisia hallinnollisia toimintoja tietokantaan (Koodiesimerkki 6) (Gilmore 2005, 617-618).

```php
<?php
/* Muodostetaan yhteys MySQL-palvelimeen ja valitaan tietokanta */
$connect = @mysql_connect("server","user","secret")
        or die("Ei voida yhdistää palvelimeen");
/* Tee kysely kantaan */
$query = "SELECT pd_name, pd_price
          FROM tbl_product
          WHERE pd_id = 101
          ORDER BY pd_name ASC";
$result = mysql_db_query("tietokanta", $query);
/* Suljetaan yhteys */
mysql_close();
?>
```

Koodiesimerkki 6. Yhteyden muodostaminen tietokantaan ja yksinkertaisen SQL-haun tekeminen PHP:n MySQL API-rajapinnan komentojen avulla.

2.6 JavaScript

JavaScript on komentosarjakieli, jonka tärkein ominaisuus on lisätä WWW-sivuille dynaamisia ominaisuuksia. Sen alkuperäinen kehittäjä on Netscape Communications Corporation ja se julkaistiin yhtiön vuonna 1995 markkinoille tuomassa Netscape Navigator 2 –selaimessa. JavaScriptia käytetään pääasiallisesti web-selainympäristössä, mutta se sopii käytettäväksi myös esimerkiksi web-palvelimiin. (Peltomäki 2000, 2.)

Koska JavaScripti on tulkattava kieli, se usein luokitellaan skriptikieliin, toisin kuin oikeat olio-ohjelmointikielet. Skriptikielien käyttökohteina ovat erilaiset sovellustoimintojen laajennukset. Skriptin pituus ei ole vakio, vaan se voi vaihdella yhden rivin ja kokonaisen sovelluksen välillä. JavaScript suoritetaan aina selaimessa tai muussa sitä tukevassa sovelluksessa ja toisin kuin useimmat sovellusohjelmointikielet, JavaScriptia ei tarvitse kääntää ennen suorittamista. Tämä mahdollistaa nopean ja helpon ohjelmistokehityksen. (Peltomäki 2000, 6.)

JavaScript-skripti aloitetaan aina <script>-elementillä, joka sijoitetaan HTML-dokumentissa <head>- tai <body>-osioon, riippuen skriptin toiminnasta (Koodiesimerkki 7). Skripti on myös mahdollista hakea erillisestä tiedostosta, mihin viitataan <script>-elementin sisällä src–attribuutilla.

Koska W3C:n validaattori pitää JavaScript-koodia merkkidatana, tulee se merkata XHTML-yhteensopivaksi CDATA-merkinnällä. Tämän merkinnän ansiosta JavaScript-koodi jätetään huomioimatta tarkastaessa dokumenttia. (W3 Schools 2010b.)

```
<html>
<head>
    <title>JavaScript-esimerkki</title>
</head>
<body>
<script type="text/javascript">
<![CDATA[
function printText()
{
    document.write("Tämä on JavaScriptia");
}
    printText();
]]>
</script>
</body>
</html>
```

Koodiesimerkki 7. Yksinkertainen JavaScript esimerkki.

Tällä hetkellä uusin vakaa versio JavaScriptista on 1.8.1 vuodelta 2009. 2000-luvun aikana JavaScript on päivittynyt useita kertoja ja saanut päivityksien mukana muun muassa vaikutteita Python-ohjelmointikielestä, tuet iteraattoreille ja JSON:lle (*JavaScript Object Notation*) sekä lisäkomentoja taulukoiden käsittelemiseen. (JavaScript 2010.)

2.7 JavaScript jQuery-kirjasto

JavaScript-kirjastot ovat joukko valmiita työkaluja ja oikopolkuja, jotka mahdollistavat helpomman tavan ohjelmoida JavaScripteihin pohjautuvia sovelluksia, erityisesti AJAX:lle (Asynchronous JavaScript And XML) ja muille web-pohjaisille teknologioille (JavaScript library 2010). Kirjastot mahdollistavat muun muassa yhteensopivuuden muiden sovellusten kanssa ja käyttöliittymäkomponenttien tekemisen. Niiden avulla voidaan myös olla yhteydessä DOM-ohjelmointirajapintaan (Document Object Model) ja käsitellä sitä (Snook, Gustafson, Langridge & Webb (2007, 81).

Useat valmiit työkalut, joita JavaScript-kirjastot sisältävät, tuottavat suuren määrän koodirivejä ja kasvattavat kirjaston tiedostokokoa. Tästä syystä kirjastojen kehittäjät ovat helpottaneet ohjelmoijien työtä rakentamalla kirjastot koostumaan moduuleista. Näiden moduulien ansiosta ohjelmoija voi valita kirjastoista vain ne ominaisuudet, joita hän tarvitsee, ja näin ollen koodirivien määrä pysyy minimissään. (Snook ym. (2007, 81.)

Lähes jokainen JavaScript-kirjasto on julkaistu avoimen lähdekoodin lisenssillä, joten ne ovat vapaasti käytettävissä, levitettävissä ja muokattavissa (JavaScript library 2010).

JQuery on myös yksi JavaScriptin avoimeen lähdekoodiin perustuvista kirjastoista. Lindleyn (2009, 2) mukaan ohjelmoijan tulisi valita jQuery-kirjasto, koska se on ilmainen, sen tiedostokoko on erittäin pieni ja huomioi eri selainten erot, jolloin käyttäjän ei itse tarvitse tehdä sitä. JQuery on myös hyvin suosittu, minkä ansiosta sen käyttäjäyhteisössä on paljon kielen kehittymisen edistäjiä ja uusien ohjelmoijien avustajia.

Lindleyn kirjoittama kirja ”JQuery Cookbook” (2009) on yksi uusimpia jQuery-kirjastosta kertovista kirjoista, joka tarjosi tarkempaa tietoa kielestä kuin esimer-

kiksi jQueryn omat verkkosivut (www.jquery.com). Tästä syystä valitsin tämän kirjan yhdeksi lähteeksi tehdessäni tätä työtä.

JQueryyn on saatavilla myös suuri määrä liitännäisiä, plugineita, jotka mahdollistavat kirjaston muuntautumisen hyvin useaan eri käyttötarkoitukseen. Nämä pluginit ovat vapaasti saatavilla Internetissä osoitteessa http://plugins.jquery.com/.

Yksi jQueryn suurimmista hyödyistä on mahdollisuus viitata suoraan HTML-koodin elementteihin ilman erillistä dokumentin elementtivalitsinta (Koodiesimerkki 8). Elementin sisältöä voidaan muokata, sille voidaan määritellä uusi attribuutti sekä tämän arvo ja se voidaan piilottaa käyttäjältä koska tahansa napin painalluksella.

```
<head>
<script src="jquery-folder/jquery-latest.js"></script>
<script type="text/javascript">
// Kun dokumentin lataus on valmis..
$(document).ready(function(){
    // ..tarkastetaan, painaako käyttäjä linkkiä
    $(".nappi").click(function () {
        // Piilotetaan <p>-elementti ja sen sisältö
        $("p").hide("slow");
    });
});
</script>
</head>
<body>
  <a class="nappi" href="#">Piilota</a>
  <p>Tämä teksti piiloitetaan</p>
</body>
```

Koodiesimerkki 8. Jos käyttäjä painaa linkkiä, koko <p>-elementti ja sen sisältö piilotetaan käyttäjältä.

3 Relaatiotietokannan toteutus

3.1 Relaatiotietokannan suunnittelu

3.1.1 Alkuasetelma

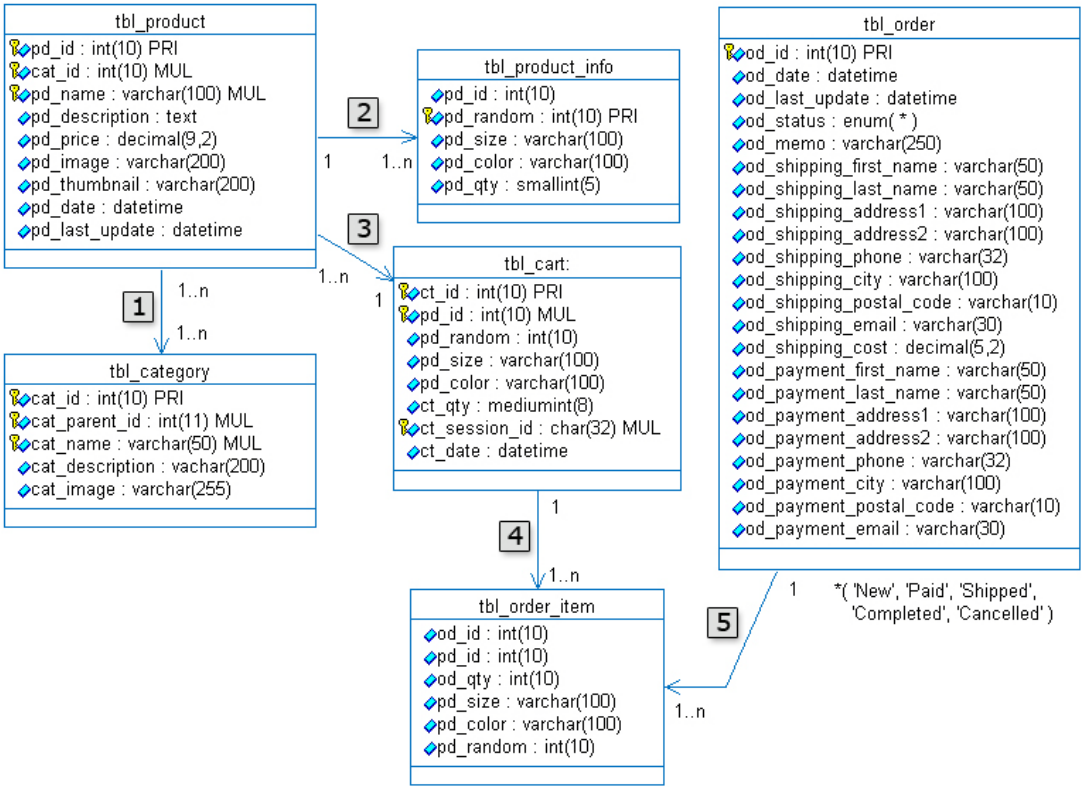
Kuten moni muukin Internetin verkkokauppa, myös tässä työssä tehty verkkokauppa käyttää relaatiotietokantaa hallitakseen suuria määriä asiakas- ja tuotetietoja sekä muita kaupalle tärkeitä tietoja.

Ennen ohjelmointityön aloittamista, kaupalle suunniteltiin relaatiotietokanta kattamaan kaikkia kaupan vaatimia tarpeita. Tämä etukäteen suunnittelu helpotti myös erityisen paljon itse ohjelmointityötä, sillä kaupan rakenteesta sai huomattavasti selkeämmän kuvan tietokantasuunnittelun avulla.

Suunnittelun jälkeen Mango Hotellin verkkokaupalle päädyttiin luomaan tietokantaan yhdeksän erilaista taulua, joista jokaisella on oma yksilöllinen tehtävä. Nämä taulut ja niiden yhteydet toisiinsa esitellään seuraavissa alaluvuissa.

3.1.2 Tuotetaulut

Tuotteisiin liittyviä tauluja luotiin verkkokaupalle kuusi kappaletta (Kuva 1). Tuotehierarkian huipulla on tuotekategoriataulu. Kategorioille luotiin oma taulu, tbl_category, mihin voidaan tallentaa pää- ja alakategorioita. Taululle on määritelty perusavain (primary key), joka yksilöi taulun sisältämät tietueet. Perusavaimen lisäksi kahdelle taulun kentälle määriteltiin moniavain (multiple key), joka mahdollistaa samanlaisten kenttien esiintymisen taulussa.



Kuva 1. Verkkokaupan tuotetaulujen rakenne ja tietotyypit sekä niiden yhteydet toisiinsa. Tietueavaimet on merkitty tietueiden perään PRI- (Primary Key) ja MUL (Multiple Key) –merkinnöillä.

Kaupan tuotteet tallennetaan tbl_product–tauluun. Tuotetaulu sisältää perustiedot tuotteesta, kuten esimerkiksi tuotenimen ja -hinnan sekä viitteen tuotekuvaan. Koska yhdellä tuotteella voi olla useita eri koko- ja värivaihtoehtoja, tallennetaan sen tarkemmat tuotetiedot tbl_product_info–tauluun, josta tietoja voidaan hakea uniikin ID:n (pd_id) perusteella.

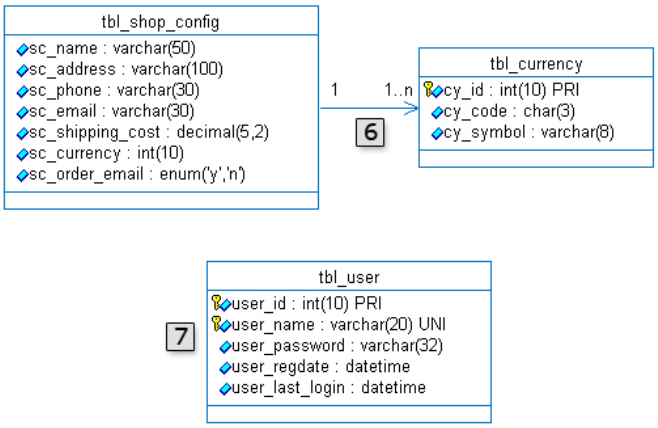
Jos verkkokaupan käyttäjä haluaa tilata jonkin tuotteen, lisää hän tämän ostoskoriin. Ostoskorille on määritelty oma taulu, tbl_cart, mihin tallennetaan väliaikaisesti tuotetiedot ja joka on yksilöity sen hetkisen istunnon ID:llä (ct_session_id).

Käyttäjällä on mahdollista muokata ostoskorin sisältöä poistamalla tuotteita tai lisäämällä niiden määrää.

Kun käyttäjä päättää tilata ostoskorissa olevat tuotteet, pyydetään hän täyttämään tilaustiedot, jonka jälkeen hänet ohjataan tilausvahvistussivulle. Jos käyttäjä jatkaa maksuvaiheeseen, poistetaan tuotteet ostoskorista (tbl_cart–taulusta), ja ne lisätään tbl_order_item–tauluun. Onnistuneen tilauksen jälkeen käyttäjän tilaustiedot tallennetaan tbl_order-tauluun, mistä on viittaus tilattuihin tuotteisiin.

3.1.3 Ylläpitäjän taulut

Mango Hotellin verkkokaupan ylläpitäjällä on mahdollista hallita kaikkia edellisessä kappaleessa mainittuja tauluja erillisen ylläpitäjäsivun kautta. Hänellä on myös mahdollisuus hallita verkkokaupan asetuksia ja ylläpitäjäkäyttäjiä, joiden tiedoille on luotu yhteensä kolme taulua (Kuva 2).



Kuva 2. Verkkokaupan asetusten ja ylläpitäjäkäyttäjien taulut. Tietueavaimet on merkitty tietueiden perään PRI- (Primary Key) ja UNI (Unique Key) –merkinnöillä.

Verkkokaupan asetukset on tallennettu tbl_shop_config–tauluun. Tämä taulu sisältää muun muassa Mango Hotellin osoitteen, puhelinnumeron ja sähköpostiosoitteen, sekä verkkokaupan postimaksun suuruuden ja sen käyttämän valuuttamuodon tiedot. Nämä kaikki tiedot ovat ylläpitäjän muokattavissa. Asetuksista voidaan myös määritellä, että lähetetäänkö ylläpitäjälle ilmoitus sähköpostiin uusista tilauksista.

Verkkokaupan käyttämälle valuutalle on luotu oma taulu tbl_currency. Valuuttatauluun voidaan tallentaa useampi eri valuuttamuoto ja valuutan tunnus, jotka voidaan ottaa käyttöön koska tahansa verkkokaupan asetuksista. Taulu luotiin verkkokaupan tulevaisuuden laajentumista silmällä pitäen, sillä tällä hetkellä verkkokauppa ei tue kuin yhden valuutan näyttämistä kerrallaan.

Ylläpitäjäkäyttäjien tiedot on tallennettu tbl_user-tauluun. Tähän hyvin yksinkertaiseen tauluun on tallennettu tiedot sivuston ylläpitäjistä, heidän salasanoistaan sekä kirjautumisajoista. Ylläpitäjäkäyttäjän nimi on määritelty uniikilla avaimella (unique key), mistä johtuen kahta samannimistä käyttäjää ei voi esiintyä taulussa. Jotta käyttäjien salasanat eivät päätyisi vääriin käsiin, ne salataan (enkryptataan) tauluun.

3.1.4 Taulujen relaatiot

Tietokannan taulujen relaatiot on merkitty kuvissa 1 ja 2 numeroilla ja ne ovat seuraavat:

1. Yksi tai useampi tuote kuuluu yhteen tai useampaan tuotekategoriaan.
2. Yhdellä tuotteella voi olla yksi tai useampi tarkempi tuotemääritys (koko, väri ja varastosaldo).
3. Yksi tai useampi tuote voi kuulua vain yhteen ostoskoriin.
4. Yhteen ostoskoriin voi kuulua yksi tai useampi tilattu tuote.
5. Yksi tilaus voi sisältää yhden tai useamman tilatun tuotteen.
6. Verkkokaupan asetukset voivat sisältää yhden tai useamman valuuttamuodon.
7. Verkkokaupan ylläpitäjäkäyttäjät -taululla ei ole relaatiota.

3.2 Relaatiotietokannan toteutus

Verkkokaupan käyttämä relaatiotietokanta luotiin Mango Hotellin valmiiksi hankkimaan tietokantaan. Edellisessä alaluvussa esitellyt taulut luotiin kantaan käsin syöttämällä ja seuraavaksi esittelen tarkemmin, miten ostoskorin tietoja sisältävä taulu luotiin tietokantaan (Koodiesimerkki 9).

```
CREATE TABLE `tbl_cart`
(
  `ct_id` int(10) unsigned NOT NULL auto_increment,
  `pd_id` int(10) unsigned NOT NULL default '0',
  `pd_random` int(10) unsigned NOT NULL,
  `pd_size` varchar(100) default '',
  `pd_color` varchar(100) default '',
  `ct_qty` mediumint(8) unsigned NOT NULL default '1',
  `ct_session_id` char(32) NOT NULL default '',
  `ct_date` datetime NOT NULL default '0000-00-00 00:00:00',
  PRIMARY KEY  (`ct_id`),
  KEY `pd_id` (`pd_id`),
  KEY `ct_session_id` (`ct_session_id`)
)
ENGINE=MyISAM AUTO_INCREMENT=58 ;
```

Koodiesimerkki 9. Ostoskorin tietoja sisältävän taulun ja sen tietueiden luonti.

Taulun luonti tietokantaan toteutetaan CREATE TABLE –komennolla, jonka jälkeen sulkumerkkien sisälle määritellään halutut tietueet ja niiden tietotyypit. Taulun sekä tietueiden nimet voivat olla mitä tahansa, mutta hyvä tapa on käyttää selkeitä englanninkielisiä nimiä tai lyhenteitä, jotta kuka tahansa taulua tarkasteleva ymmärtäisi sen sisällön.

Tarkastellaan seuraavaksi sulkujen sisältä ensimmäistä riviä. Ensimmäisellä rivillä luodaan tietue nimeltä ct_id, joka on kokonaislukutyyppinen, ja yksilöi jokaisen ostoskoriin lisätyn tuotteen. Int(10) unsigned määrittelee, että tietueeseen voidaan tallentaa vain positiivinen arvo väliltä 0-4294967295. NOT NULL –määre ilmoittaa, että syötetty arvo ei voi olla tyhjä. Ennen rivin päättymistä pilkkuun, on tietueelle asetettu auto_increment–määre, joka on MySQL:n oma SQL-standardista poikkeava tapa määritellä tietue sellaiseksi, että jokaiselle uudelle riville luodaan automaattisesti uusi kokonaisluku, kun ostoskoriin lisätään uusi tuote. Tällöin rivit automaattisesti numeroituvat muotoon 1, 2, 3 ja niin edelleen.

Muita tietotyyppejä esimerkissä ovat muun muassa tekstityypit varchar ja char sekä aikatyyppi datetime. Default–määritys määrittelee arvon, joka annetaan automaattisesti kentälle, mikäli se ei saa mitään syötettyä arvoa.

Ennen sulkumerkkien sulkemista, määritellään tietueille avaimet. Kokonaislukutyyppinen tietue ct_id on määritelty perusavaimella (PRIMARY KEY), jolloin se saa aina yksilöllisen arvon. Kaksi muuta avainta (KEY) sallivat samanlaisten arvojen sisältymisen pd_id ja ct_session_id –tietueilla.

Luotu taulu on määritelty käyttämään taulutyyppiä MyISAM, joka mahdollistaa muun muassa sen, että taulu pystyy tallentamaan enemmän tietoa pienempään tallennustilaan. Gilmoren (2005, 566) mukaan MyISAM-taulutyypit ovat myös käyttöjärjestelmästä riippumattomia, jolloin niitä voidaan käyttää niin Windows- kuin Linux-palvelimellakin.

Lopuksi on määritelty, että kaikkien auto_increment-määrettä käyttävien tietotyyppien automaattisesti luodut kokonaisluvut alkavat numerosta 58.

4 Verkkokauppasovelluksen toteutus

4.1 Sivuston rakenne

Mango Hotellin verkkokauppasovelluksesta haluttiin hyvin selkeä ja helppokäyttöinen, mutta myös ilmeeltään tuore sivu. Sivut toteutettiin hyvin yksinkertaisilla grafiikoilla, joilla säästetään pitkiä latausaikoja ja tehdään sivuista selkeät.

Verkkokaupan rakenne on toteutettu pääosin HTML:n div-elementeillä, joiden sijainti ja sisältömääritykset on tehty CSS-tyylitiedostojen avulla. Nämä elementit luovat asemointilohkoja, joilla voidaan rajata ne HTML-elementit, joihin halutaan kohdistaa erilaisia määrityksiä, esimerkiksi koko-, sijainti- tai värimäärityksiä. Näin sivuston ulkoasu, esimerkiksi kuvat, on voitu jakaa palasiin, jotka on sijoiteltu div-elementteihin ja määritelty oikeisiin paikkoihin CSS-komentojen avulla.

Suurin osa sivujen toiminnallisista osista on ohjelmoitu PHP:lla, esimerkiksi ostoskorin toiminta ja tuotteiden esille tuonti, mutta myös JavaScriptia ja jQuerya käytettiin pitkälti, muun muassa lomakkeiden tarkastuksiin ja sivujen dynaamisuuden lisäämiseen.

Käyttäjä voi liikkua kaupassa ylä- ja alapalkista löytyvien linkkien kautta ja hän voi selata tuotteita vasemmalta löytyvän tuotekategorialistan avulla (Kuva 3). Tuotekategorialistan alta löytyy verkkokauppaan lisätty uusin tuote ja sivuston oikeasta reunasta miniostoskori. Sivustolla on myös pikalinkkejä, esimerkiksi tuotehaku yläpalkin navigointilinkkien vieressä, sekä ostoskärry, josta pääsee tarkastelemaan ostoskorin sisältöä. Eri sivujen sisällöt ladataan aina sivun keskellä olevaan pääsisältökenttään.

Kuva 3. Mango Hotellin verkkokauppasovelluksen ulkoasu ja sen etusivu.

Include()–funktion käyttö

Koska sivulla on hyvin paljon elementtejä, joiden sisältö ei muutu käyttäjän selailun aikana, on ne eritelty erillisiin tiedostoihin, ja niiden lataamiseen on käytetty PHP:n funktiota include() (Koodiesimerkki 10). Tämä funktion avulla minkä tahansa ennalta määrätyn tiedoston tiedot voidaan ladata mihin tahansa kohtaan HTML-koodia. Esimerkiksi miniostoskori ja tuotekategorialista pysyvät aina käyttäjän näkyvissä, vaikka tämä navigoisi linkistä toiseen.

```
<?php
    include("include/header.php");
    include("include/header_content.php");
?>
```

Koodiesimerkki 10. Include()-funktion käyttö.

Include()–funktion käyttö säästää hyvin paljon sivuston latausaikaa, sillä esimerkiksi sivuston ulkoasu ladataan vain kerran käyttäjän tullessa sivustolle ensimmäisen kerran eikä enää uudestaan tämän siirtyessä sivulta toiselle. Erillisten tie-

dostojen käyttö helpottaa myös sivujen päivittämistä paljon, sillä yhden tiedoston muokkaamisella voidaan vaikuttaa koko sivuston toimintaan.

Include()–funktion lisäksi sivulla käytetään require_once()–funktiota, joka lataa tärkeimmät PHP-funktiot käyttöön heti käyttäjän saapuessa sivulle. Koska sivuston tärkeimmät asetukset, muun muassa tietokantayhteyden luominen, on määritelty erillisessä konfiguraatiotiedostossa, require_once()–funktiota on hyvä käyttää includen sijaan. Require_once()-funktio lataa nimensä mukaan tiedostot vain kerran, jolloin esimerkiksi tietokantayhteyttä ei avata joka kerta uudelleen, kun sivu ladataan uudestaan.

4.2 Sivuston toiminta

Verkkokauppasovelluksessa käyttäjä voi selata tuotekategorioita ja tuotteita, lisätä tuotteita ostoskoriin ja tehdä tuotetilauksia. Verkkokauppasovellus sisältää myös sivut Mango Hotellin yhteystiedoille, tuotehaulle sekä palautelomakkeelle. Tuotehaun ja palautelomakkeen toimintaa ei tässä työssä käsitellä tarkemmin.

Kun käyttäjä saapuu Mango Hotellin verkkokauppaan, hänelle luodaan automaattisesti uusi SID-tunnus session_start()-funktiolla. Tästä tunnuksesta tulee hänen yksilöllinen ostajatunnuksensa, joka on voimassa niin kauan, kun käyttäjä pysyy verkkokaupan sivuilla tai tyhjentää selaimestaan evästeet. Tunnusta käytetään myös yksilöimään hänen ostoskorinsa.

Suurin osa sivun tiedoista kuljetetaan URL-osoitteen mukana. Jos esimerkiksi käyttäjä selaa tuotteita, haetaan tuotteen yksilöivä tuote-ID URL-osoitteesta PHP:n superglobaalimuuttujalla $_GET (yleisesti GET). Kun tieto on haettu, sitä verrataan tietokannasta löytyviin tuotteisiin ja löydetty tuote palautetaan käyttäjän nähtäväksi.

Jos linkki avaa sivun “http://www.esimerkki.fi/catalog.php?product=225”, catalog.php–sivu vastaanottaa osoitteessa määritellyn muuttujan $_GET[‘product’], jonka arvo on merkkijono “225”. Saatua merkkijonoa voidaan käsitellä edelleen ja verrata tietokannasta löytyviin tuotteisiin. Sivuilla on käytetty myös toista superglobaalimuuttujaa $_POST (yleisesti POST), millä tietoja voidaan lähettää eteenpäin.

Lähes jokaisessa verkkokaupan tietokantaa käsittelevässä funktiossa käytetään PHP:n MySQL API -rajapintaa. Kun tietokantaan on tehty tarvittava SQL-haku, sen tulos käsitellään mysql_query()–funktiolla, joka on vastuussa kyselyn suorittamisesta. Se palauttaa arvon TRUE (1) tai FALSE (0) riippuen kyselyn onnistumisesta ja sen tarkoituksesta.

Jos tietokantakyselyn tulee palauttaa esimerkiksi tietue tuotetiedoista, sitä voidaan käsitellä eteenpäin. Saatu tietue syötetään mysql_fetch_assoc()–funktiolle, joka palauttaa assosiatiivisen taulukon ja esittelee avaimen kentän nimenä ja sen arvon kentän sisältönä. Taulukko käsitellään vielä extract()–funktiolla, joka tekee sille annetusta assosiatiivisesta taulukosta avaimia vastaavat muuttujat. Tämän jälkeen tulostamalla kentän nimi saadaan sen sisältö esille. Esimerkissä (Koodiesimerkki 11) extract()-funktion avulla tietokannasta haettu tuotenimi ja tuotehinta voidaan tulostaa suoraan muuttujanimillä.

```php
<?php
    // Haetaan URL-osoitteesta tuote-ID
    $productId = $_GET['p'];
    $sql = "SELECT pd_name, pd_price
            FROM tbl_product
            WHERE pd_id = $productId";

    $result = mysql_query($sql) or die(mysql_error());
    $row    = mysql_fetch_assoc($result);
    extract($row);
    // Tulostetaan
    echo "Tuotenimi on $pd_name ja hinta $pd_price";
?>
```

Koodiesimerkki 11. Haettu tuotenimi ja tuotehinta voidaan tulostaa suoraan muuttujanimillä.

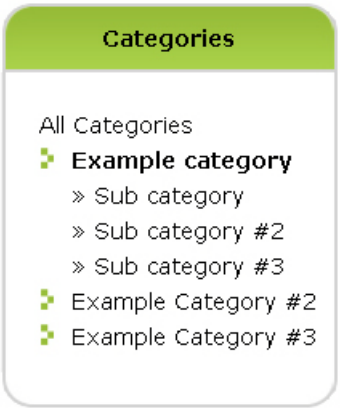
4.3 Tuotekategoriat

Tuotekategorioiden tiedot on tallennettu tietokantaan ja ne on yksilöity kahdella eri tapaa: cat_id antaa kategorialle yksilöllisen ID:n ja cat_parent_id kertoo, onko kategoria pääkategoria, eli sen alle kuuluu alakategorioita. Sivujen ylläpitäjä ei voi lisätä tuotteita pääkategorioiden alle, joten jokaisella pääkategorialla on vähintään yksi alakategoria.

Kategorian yksilöivä ID (cat_id) kulkee selaimen URL-osoitteen mukana aina kun käyttäjä klikkaa kategorialistasta haluamaansa linkkiä. Kategoria näytetään URL-

osoitteessa muodossa “c=123”, jossa luku määrää kategorianumeron. Numero haetaan GET-muuttujaan ja otetaan tämän jälkeen käsittelyyn. Tämän avulla tuotekategoriat tietokannasta hakeva funktio osaa aina määritellä oikeat tuotteet oikeiden tuotekategorioiden alle ja listata ne käyttäjän näkyville.

Tuotekategoriat on jäsennelty hyvin selkeästi ja käyttäjän sen hetkinen sijainti kategoriassa näytetään tummennetulla tekstillä navigointipalkissa (Kuva 4). Pääkategorian alakategoriat avautuvat käyttäjän näkyviin vasta tämän klikatessa pääkategorian linkkiä. Kategorioiden sisältämät tuotteet avautuvat keskelle sivua.



Kuva 4. Tuotekategorioiden jäsentely. Alakategoriat tulevat näkyviin vasta käyttäjän painaessa pääkategorian linkkiä.

4.4 Tuotteet

Kuten tuotekategorioilla, myös tuotteilla on ne yksilöivä tuote-ID, joka kulkee URL-osoitteen mukana. Tuotteen ID esitetään osoitteessa aina tuotekategorian ID:n jälkeen p-kirjaimen arvolla, esimerkiksi “catalog.php?c=123&p=321”. Näin ollen osoitteesta voidaan hakea tuotekategorian ja siihen kuuluvan tuotteen tiedot GET-muuttujalla.

Tarkemmissa tuotetiedoissa käyttäjälle näytetään tuotekuva, tuotteen selostus, tuotteen hinta, tuotteen koko- ja väritiedot, sekä jokaisen koko- ja väritiedon sen hetkinen varastosaldo. Tuotekuvaa painamalla avataan jQuerylla toteutettu kuvaikkuna, joka näyttää tuotekuvan suurennettuna.

Tuotekuva, tuotteen selostus ja tuotehinta ovat tallennettu tietokantaan tbl_product-tauluun, mutta tuotteen koko- ja väritiedot sekä varastosaldo hae-

taan tbl_product_info–taulusta. Tämä siksi, että jokaisella tuotteella voi olla niin monta eri koko- tai värivaihtoehtoa, joiden varastomäärät vaihtelevat, että niiden tallentaminen samaan tauluun muiden tuotetietojen kanssa kasvattaisi taulun kokoa huomattavasti ja tekisi siitä sekavan.

Tuotteiden koko-, väri- ja varastotiedot haetaan erillisestä taulusta viittaamalla tuotteen ID-numeroon, mikä tallennetaan automaattisesti tbl_product_info-tauluun, kun ylläpitäjä lisää verkkokauppaan uuden tuotteen. Lisäksi jokaiselle tietueelle, joka sisältää tuotteen koon, värin ja saldon, on määritelty yksilöivä ID (pd_random).

Koko- ja värivaihtoehdon valinta

Kun käyttäjä tarkastelee tuotteen tarkempia tietoja, voi hän valita esimerkiksi t-paidan koon ja värin erillisestä alasvetovalikosta (Kuva 5). Alasvetovalikkoon haetaan tietokannasta kyseessä olevan t-paidan saatavilla olevat tiedot. Kun käyttäjä on valinnut alasvetovalikosta esimerkiksi koon XL, jonka väri on sininen, tulee esiin erillinen div-elementti, josta voidaan valita haluttu kappalemäärä kyseiselle valinnalle.

Kuva 5. Käyttäjälle näkyvä sivu tarkemmista tuotetiedoista. Alasvetovalikosta voidaan valita haluttu tuotekoko ja –väri, jonka jälkeen tuodaan esiin kappalemäärävalitsin.

Alasvetovalikon toiminta on toteutettu JavaScriptin jQuery-kirjaston avulla. Kirjaston valmiit funktiot slideUp() ja slideDown() tuovat sulavasti esiin erillisen kappalemäärävalitsimen, kun valikosta valitaan haluttu tuotekoko. Sivua ei ladata uudestaan, vaan kappalevalitsin on piilotettu käyttäjän näkyvistä ja se tuodaan aina esiin tai piilotetaan, kun valikossa tehdään valinta.

Alasvetovalikon jokaiseen valittavaan HTML-elementtiin (option-elementti), mitkä näyttävät käyttäjälle saatavilla olevat koko- ja värivaihtoehdot, on asetettu tuotekoon varastosaldot. Nämä varastosaldot haetaan PHP:n avulla tietokannasta ja asetetaan alasvetovalikon jokaisen valinnan kohdalle class–attribuuttiin. Kun käyttäjä valitsee haluamansa koon alasvetovalikosta, jQuery hakee valinnan attribuutista varastosaldon ja tallentaa sen muistiin.

Varastosaldo tallennetaan muistiin siksi, ettei käyttäjä voi tilata esimerkiksi viittä kappaletta sinisiä t-paitoja, jos niitä on varastossa vain neljä kappaletta. Kun käyttäjä painaa plusnappia valitessaan kappalemäärää siniselle t-paidalle, lisääntyy numero nappien välissä. Valittua kappalemäärää verrataan muistissa olevaan varastosaldoon, ja kun se ylitetään, tuodaan slideDown()–funktiolla käyttäjälle esiin pahoitteluviesti varastosaldon ylittymisestä (Kuva 6).

Myös tuotteen koko-, väri- ja saldotiedot yksilöivä ID, pd_random, on tallennettu option-elementtiin, mutta value–attribuuttiin. ID tallennetaan ostoskoriin, kun sinne lisätään tuote ja se varmistaa, ettei ostoskori lajittele esimerkiksi valkoista ja mustaa t-paitaa samaksi tuotteeksi.

Kun käyttäjä on valinnut tuotteesta halutun koko- ja värikombinaation, painaa hän Add To Cart –nappia, jolloin tuote lisätään ostoskoriin.

Kuva 6. Pahoitteluviesti on tuotu esiin, koska valittua tuotetta ei ole varastossa kuin yksi kappale.

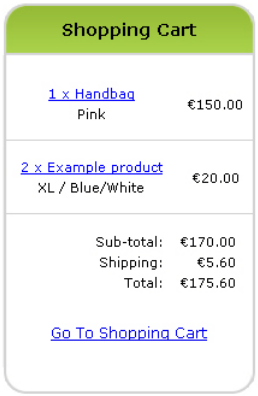
4.5 Ostoskorit

Käyttäjän selatessa verkkokaupan sivulle, luodaan hänelle eväste, joka sisältää yksilöllisen SID-tunnuksen. Tämä tunnus saadaan käsittelyyn PHP:n session_id()–funktiolla. SID-tunnus lisätään aina automaattisesti ostoskorin tietokantatauluun jokaisen tilatun tuotteen tietueelle, mikä yksilöi ostoskorin ja sen tuotteet käyttäjälle. Käyttäjän selainohjelma vanhentaa SID-tunnuksen sisältämän evästeen automaattisesti tai käyttäjä voi poistaa sen itse tyhjentämällä selaimensa evästeet.

Jos eväste poistetaan tai se vanhenee, tyhjentyvät ostoskorit automaattisesti, koska SID-tunnus vaihtuu. Ostoskorin sisältö jää kuitenkin tietokantaan tbl_cart–tauluun, koska ei voida määritellä, milloin käyttäjän yksilöity SID-tunnus poistuu käytöstä, ja näin ollen poistaa taulusta käyttäjän hylkäämät tuotteet. Tuotteet poistetaan taulusta seuraavan kerran, kun kuka tahansa uusi käyttäjä lisää ostoskoriin tuotteita.

Kun verkkokaupan käyttäjä haluaa tilata tuotteen, lisätään se miniostoskoriin sekä varsinaiseen ostoskoriin. Näiden kahden ostoskorin tiedot haetaan samasta tietokantataulusta, mutta niiden toiminta on hieman erilaista.

Jotta käyttäjä voisi helposti tarkastella tekemiään tuotevalintoja, näkyvät hänen ostoskorin sisältämät tuotteet miniostoskorissa sivun oikealla reunalla (Kuva 7). Miniostoskorissa näkyvät tuotteet ja niiden kappalemäärät sekä koko-, väri- ja hintatiedot. Ostoskori laskee myös käyttäjälle automaattisesti valittujen tuotteiden yhteishinnan sekä tulevan laskun summan postimaksun kanssa.



Kuva 7. Miniostoskorin näkymä.

Varsinainen ostoskori löytyy oman linkin takaa sivuston yläpalkista. Tästä ostoskorista käyttäjä näkee tuotteet kuvan kera ja voi valita, haluaako hän poistaa tai lisätä niiden kappalemääriä (Kuva 8). Kappalemääriä ei pysty lisäämään yli varastosaldon, mutta jos näin tapahtuu, käyttäjälle näytetään pahoitteluviesti varastosaldon ylityksestä. Jos tuotteen kappalemäärää vähennetään ja määrä alittaa yhden kappaleen, poistetaan tuote kokonaan ostoskorista. Näin koko ostoskori voidaan tyhjentää tuotteista. Tästä ostoskorista löytyy myös nappi, josta käyttäjä pääsee seuraavaan vaiheeseen, eli täyttämään tilaustiedot ja tilaamaan tuotteet.

Kuva 8. Varsinainen ostoskori.

4.6 Tuotteen tilaaminen

4.6.1 Tilaustietojen kirjaaminen

Kun käyttäjä päättää tilata valitsemansa tuotteet, siirtyy hän tilaustietojen täyttämiseen. Tilaustiedoissa häneltä kysytään erikseen tilaajan ja tilauksen maksajan henkilö- ja osoitetiedot, jonka jälkeen käyttäjän tulee valita maksuvaihtoehto, millä tilaus maksetaan. Oletuksena tilaukset maksetaan luottokortilla. Kun vaaditut kentät on täytetty, voi käyttäjä jättää vielä vapaamuotoisen kommentin liittyen tilaukseen, ennen kuin siirtyy seuraavalle sivulle täytettyjen tietojen oikeellisuuden varmistukseen.

Kaikki täytettävän tilaustietolomakkeen pakolliset kentät tarkastetaan JavaScriptillä. Tarkastuksessa kentät käydään läpi ja varmistetaan, ettei käyttäjä ole jättänyt pakollisia kenttiä tyhjiksi. Kun käyttäjä haluaa jatkaa seuraavaan vaiheeseen tilaustietojen täyttämisen jälkeen ja painaa siirtymisnappia, lähettää lomake tarkastuspyynnön JavaScriptille. Tämä tarkastuspyyntö lähetetään HTML-lomakkeen mukana form-elementissä attribuutilla onsubmit=”return funktio();”. Esimerkissä (Koodiesimerkki 12) kutsutaan funktiota checkShippingAndPaymentInfo(), joka

palauttaa ponnahdusikkunan varoitustekstillä, jos käyttäjä on jättänyt tyhjäksi vaadittuja kenttiä. Skripti tarkastaa kentän tyhjyyden isEmpty()–funktiolla.

```
function checkShippingAndPaymentInfo()
{
    with (window.document.frmCheckout) {
        if (isEmpty(txtShippingFirstName, 'Enter first name')) {
            return false;
        } else if (isEmpty(txtShippingLastName, 'Enter last name')) {
            return false;
        } else if (isEmpty(txtShippingAddress1, 'Enter shipping address')) {
            return false;
        ......
        } else {
            return true;
        }
}
```

Koodiesimerkki 12. Tilaustietokenttien tarkastus.

Maksuvaihtoehdot

Käyttäjällä on valittavana kolme eri maksutapaa: luottokortti, Sampo-pankin etukäteismaksu tai postimaksu. Luottokorttimaksupalvelu on toteutettu Luottokunnan tarjoaman valmiiksi toteutetun maksupalvelun avulla. Jos käyttäjä valitsee maksutavaksi luottokortin, ohjataan hänet erilliselle luottokunnan suojaamalle maksusivulle, missä hän voi maksaa tilauksensa syöttämällä luottokortin tiedot.

Sampo-pankin maksupalvelu ohjaa käyttäjän pankin omalle verkkopankkisivulle, missä käyttäjä voi maksaa tilauksensa pankin verkkotunnuksien avulla. Luottokunnan ja Sampo-pankin maksutavat edellyttävät erillisten HTML-lomakkeiden lisäämistä verkkokauppaan ja ne ovat saatavilla yritysten kautta. Maksutapojen ja lomakkeiden toimintaa ei käsitellä tässä työssä tarkemmin.

Postimaksulla maksaessa käyttäjän tilaamat tuotteet lähetetään verkkokaupasta ostajaa lähimpänä olevaan postiin. Tuotteiden lähettämisestä vastaa verkkokaupan ylläpitäjä. Tuotteet noudetaan postista tilauksen suuruutta vastaavaa summaa vastaan.

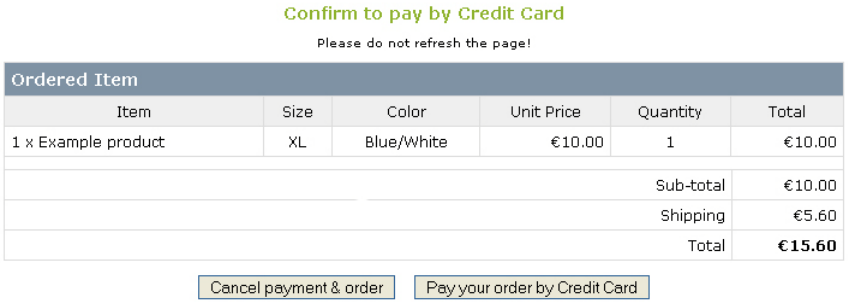
Tilaustietojen täyttämisen voi peruuttaa ja palata takaisin verkkokauppaan, esimerkiksi lisäämään tuotteita ostoskoriin, kunhan käyttäjä ei ole ehtinyt tilaustietojen varmennussivulta eteenpäin, jolloin ostoskori tyhjennetään.

4.6.2 Tilauksen käsittely

Onnistunut tilaus

Verkkokaupan käyttäjän tekemä tilaus tallennetaan tietokantaan kahteen eri tauluun. Käyttäjän antamat tilaustiedot tallennetaan tbl_order–tauluun ja hänen tilaamansa tuotteet tbl_order_item-tauluun. Kaupan ylläpitäjä näkee erillisellä ylläpitäjäsivulla uudet tilaukset, jotka sisältävät kaikki käyttäjän antamat tiedot ja tilatut tuotteet.

Maksuvaihtoehdon ollessa luottokortti tai etukäteismaksu, ohjataan käyttäjä vielä tilaustietovahvistussivun jälkeen varmennussivulle (Kuva 9), mistä hän pääsee siirtymään joko Luottokunnan tai Sampo-pankin tarjoamalle erilliselle maksusivulle. Tällä varmistetaan, että käyttäjä on varmasti valinnut oikean maksuvaihtoehdon ja on tietoinen tilaamistaan tuotteista ja niiden hinnoista.



Kuva 9. Tilauksen maksutavan ja loppusumman varmennussivu.

Varmennussivulle saavuttaessa tilatut tuotteet poistetaan ostoskorista ja tietokannasta tbl_cart–taulusta. Tämän jälkeen tuotteet lisätään tbl_order_item–tauluun ja käyttäjän syöttämät tilaustiedot lisätään tbl_order–tauluun.

Jos käyttäjä siirtyy Luottokunnan tai Sampo-pankin erilliseen maksupalveluun ja maksaa tuotteensa, ohjataan hänet maksun jälkeen takaisin verkkokauppaan onnistunut tilaus –sivulle.

Kun maksuvaihtoehdoksi on valittu postimaksu ja käyttäjä hyväksyy tilauksensa, poistetaan tuotteet ostoskorista heti tilaustietojen varmennussivun jälkeen ja ne lisätään tietokantaan. Tässä vaihtoehdossa ei siirrytä enää erilliselle maksunvarmentamissivulle, vaan käyttäjä ohjataan onnistunut tilaus –sivulle.

Peruutettu tai epäonnistunut tilaus

Aiottu tilaus voidaan peruuttaa missä kohtaa tahansa varmennus- tai maksuvaihetta. Jokaisessa tapauksessa käyttäjä ohjataan verkkokaupan sivulle, jossa kerrotaan, että tilaus on peruutettu tai sen tekemisessä on ilmennyt virhe.

Jos tilaus peruutetaan tai sen tekemisessä on ilmennyt virhe, kaikki käyttäjän antamat tilaustiedot ja tilatut tuotteet poistetaan tietokannasta. Tuotteet lisätään takaisin tuotetauluihin, jolloin ne ovat jälleen verkkokaupassa tilattavissa.

Myös Luottokunnan ja Sampo-pankin palveluissa käyttäjä ohjataan takaisin verkkokauppaan, jos hän päättää peruuttaa tilauksensa esimerkiksi luottokortin tietojen täyttämisen yhteydessä.

Tilaussähköpostin lähettäminen

Onnistuneen tilauksen jälkeen käyttäjälle lähetetään sähköposti hänen tekemästään tilauksesta. Sivuston ylläpitäjä saa myös sähköpostiviestin, mikäli hän on verkkokaupan asetuksista asettanut toimimaan sähköpostin lähetyksen ylläpitäjälle aina uuden tilauksen saapuessa.

Nämä sähköpostit lähetetään automaattisesti, kun käyttäjä on ohjattu onnistunut tilaus –sivulle. Sähköposti lähetetään PHP:n mail()-funktiolla (Koodiesimerkki 13), mihin määritetään automaattisesti tilaajan sähköpostiosoite, joka saadaan tietokantaan tallennetuista tilaustiedoista.

Tuotteiden tilaaja saa sähköpostissa tiedot tilauksen ajankohdasta, tilattujen tuotteiden tiedot ja hinnat, tiedot hänen täyttämistään tilaustiedoista sekä yleisiä ohjeita riippuen hänen valitsemastaan maksutavasta. Sivuston ylläpitäjä saa sähköpostissa suoran linkin ylläpitosivulle juuri tilattuun tuotteeseen.

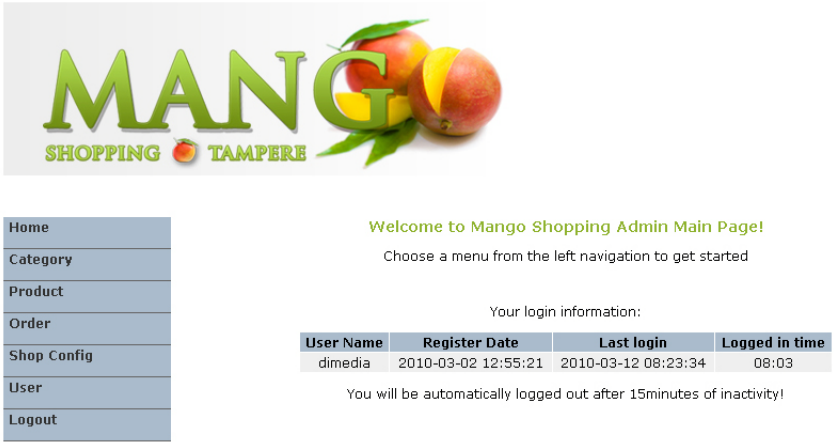
```php
<?php
$to      = 'matti.meikalainen@esimerkki.com';
$subject = 'Viestin aihe';
$message = 'Viestin sisältö';
$headers = 'From: maija.meikalainen@esimerkki.com' . "\r\n" .
    'Reply-To: maija.meikalainen@esimerkki.com' . "\r\n" .
    'X-Mailer: PHP/' . phpversion();

mail($to, $subject, $message, $headers);
?>
```

Koodiesimerkki 13. Sähköpostin lähetysesimerkki PHP:n avulla.

5 Hallintasovelluksen toteutus

5.1 Sivuston rakenne

Mango Hotellin verkkokaupan ylläpitäjän sivut ovat ulkoasultaan hyvin karsitut ja yksinkertaiset. Sivuston vasemmassa reunassa ovat kaikki sivun navigointilinkit toteutettuna CSS-tyyleillä (Kuva 10). Oikealle puolelle navigointilinkkejä avataan aina linkkien sisältö. Kuten verkkokauppasovellus, myös hallintasovellus on toteutettu pääosin HTML:n div-elementeillä, mutta siihen on myös käytetty hyvin paljon taulukointia. Hallintasovelluksen ohjelmointi on toteutettu PHP:lla ja JavaScriptilla.



Kuva 10. Hallintasovelluksen etusivunäkymä.

Ylläpitäjäsivut käyttävät yhtä sivupohjaa, johon kaikki tieto ladataan käyttäjän linkkipainallusten mukaan. Tietojen lataamiseen käytetään PHP:n funktiota require_once(), jotta sivujen käyttämiä kymmeniä eri funktioita ei ladattaisi aina uudestaan.

Jokaisella hallintasovelluksen sivulla käytetään eri JavaScript-tiedostoja, mitkä sisältävät muun muassa funktioita lomakkeiden tarkastuksiin. Funktioita on jokaisessa tiedostossa hyvin paljon ja ne sisältävät samoja tarkastusmenetelmiä. Tästä syystä tiedostoja ei ladata heti kerralla käyttöön, vaan jokainen tiedosto ladataan käyttöön vain silloin, kun sitä tarvitsevalle sivulle navigoidaan. Tällä vältetään ristiriitaisuuksien ilmeneminen.

5.2 Sivuston toiminta

Hallintasovelluksessa verkkokaupan ylläpitäjä voi lisätä, muokata ja poistaa tuotekategorioita ja tuotteita, sekä tarkastella verkkokaupan asiakkaiden tekemiä tilauksia ja muokata niiden statuksia. Ylläpitäjällä on myös mahdollisuus muokata verkkokaupan asetuksia sekä hallita ylläpitäjäkäyttäjiä.

Hallintasovelluksessa suurin osa tiedoista kuljetetaan URL-osoitteen mukana. Osoitetta tulkitaan GET- ja POST-muuttujien avulla, jotta voidaan esimerkiksi määritellä, minkä tuotteen tietoja ylläpitäjä haluaa lähteä päivittämään.

Kuten alaluvussa 3.1 kerrotaan, muun muassa tietokantayhteyksien avaamiseen tarvittavat käyttäjätunnukset löytyvät yksittäisestä konfiguraatiotiedostosta. Tämä konfiguraatiotiedosto sisältää yleiset asetukset ja määritelmät koko verkkokaupan toiminnalle, mutta suurin osa tiedoston asetuksista on kuitenkin määritelty itse hallintasovellukselle.

Konfiguraatiotiedostossa on määritelty verkkokaupan alin dokumenttikansio PHP:n komennolla $_SERVER['DOCUMENT_ROOT']. $_SERVER on taulukko, joka sisältää tiedot sivun ylätunnisteista, sen poluista ja ajettavien skriptien sijainneista. Palvelin luo nämä tiedot automaattisesti. DOCUMENT_ROOT määrittelee, missä kansiossa sille määritelty skriptitiedosto ajetaan. Tässä tapauksessa siis konfiguraatiotiedosto. (PHP.NET 2010.)

Yleisiä asetuksia on määritelty define()-funktiolla. Tämä funktio määrittelee nimetyn vakioasetuksen. Funktion avulla on määritelty esimerkiksi palvelimelle ladattavien tuotekuvien maksimikoot sekä niiden määränpääkansio, mihin kuvat tallennetaan (Koodiesimerkki 14). Define() on hyvin hyödyllinen funktio, sillä sen avulla asetetut vakiot toimivat missä tahansa sivustolla, jos itse define-määrityksen sisältävä tiedosto on ladattu käyttöön.

```
<?php
define('PRODUCT_IMAGE_DIR',  'images/product/');
define('MAX_PRODUCT_IMAGE_WIDTH', 800);
?>
```

Koodiesimerkki 14. Define()-funktiolla tehdyt määritykset.

Erikoismerkkien muuntaminen

Yksi erittäin tärkeä toimenpide pelkästään sivuston turvallisuuden kannalta on kääntää verkkokaupassa esiintyvät erikoismerkit oikein. Erikoismerkkejä ovat puolilainausmerkit (‘), lainausmerkit (“), kenoviivat (\) ja tyhjät arvot (null). Koska erikoismerkkejä käytetään muun muassa merkkijonojen erottajina ohjelmoinnissa sekä kirjoitetussa tekstissä, tulee ne erottaa toisistaan ohjelmointivirheiden välttämiseksi.

Virheiden välttämiseksi konfiguraatiotiedostoon on tehty erikoismerkkitarkastaja, joka tekee erikoismerkkien kääntämisen, eli koodinvaihdon, jos sille on tarvetta. Koodinvaihdossa erikoismerkkien eteen lisätään kenoviiva (\), eli näin ollen lainausmerkistä tulisi \”–muotoinen. Tarkastus on esitelty seuraavassa koodiesimerkissä (Koodiesimerkki 15) ja selitetty yksityiskohtaisesti sen jälkeen.

```
<?php
if (!get_magic_quotes_gpc()) {
    if (isset($_POST)) {
        foreach ($_POST as $key => $value) {
            $_POST[$key] =  trim(addslashes($value));
        }
    }

    if (isset($_GET)) {
        foreach ($_GET as $key => $value) {
            $_GET[$key] = trim(addslashes($value));
        }
    }
}
?>
```

Koodiesimerkki 15. Erikoismerkkien koodinvaihto.

Koodiesimerkissä tarkastetaan ensimmäiseksi, onko palvelimella otettu magic_quotes_gpc()–funktio (Get, Post, Cookie, yleisesti GPC) käyttöön. GPC tekee koodinvaihdon automaattisesti muuttujille, jotka haetaan GET- ja POST-superglobaalimuuttujilla tai evästeistä. GPC:n käyttö tarkastetaan get_magic_quotes_gpc()–tarkastusfunktiolla, joka palauttaa tiedon GPC:n tilasta palvelimella.

Jos magic_quotes_gpc()-funktion käyttö on käännetty palvelimelta pois päältä, tehdään erikoismerkkien koodinvaihto toisella tavalla. Kaikki GET- ja POST-muuttujilla lähetetyt merkkijonot käsitellään trim()- ja addslashes()–funktioilla. Trim()-funktio poistaa merkkijonoista kaikki tyhjät välilyönnit sekä muut erikoismerkit, esimerkiksi rivinvaihdon merkin “\n”. Addslashes()-funktio lisää erikois-

merkkien eteen kenoviivan, eli se tekee saman toiminnan, kuin magic_quotes_gpc()-funktio.

Esitellyllä tarkastusmenetelmällä saataisiin esimerkiksi lause “Jari sanoi, “pidän ohjelmoinnista”” käännettyä muotoon “Jari sanoi, \”pidän ohjelmoinnista\””. Näin ollen PHP osaisi tulostaa jälkimmäisen tekstin oikein, kun taas ensimmäisessä tekstissä se ei ymmärtäisi kahta lainausmerkkiä peräkkäin ja antaisi virheilmoituksen.

Virheiden välttämiseksi on erikoismerkkien käsittely hyvin tärkeää tehdä aina ohjelmoitaessa web-pohjaisia ja erityisesti tietokantaa käyttäviä sovelluksia. Erikoismerkkien käsittelyllä minimoidaan myös erilaiset tietoturvariskit, esimerkiksi SQL-injektiot. SQL-injektio on tekniikka, jota käytetään tietoturva-aukkojen hyödyntämiseen tietokantapohjaisissa sovelluksissa. Lisätietoja sen toiminnasta ja ehkäisemisestä löytyy osoitteesta http://wiki.mureakuha.com/wiki/SQL-injektio.

5.3 Sovellukseen kirjautuminen

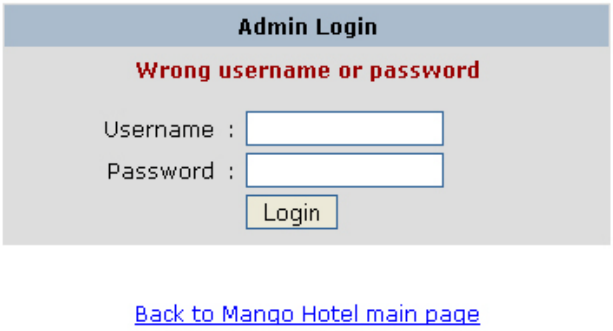
Hallintasovellukseen kirjautuminen sallitaan vain niiltä käyttäjiltä, joille on luotu ylläpitäjätunnukset. Sovellukseen kirjaudutaan erilliseltä sivulta, jossa käyttäjältä kysytään käyttäjätunnus ja salasana.

Sisään kirjautuessa käyttäjätunnusta ja salasanaa verrataan tietokannasta löytyvän käyttäjätaulun (tbl_user) tietueisiin ja etsitään niille vastaavuutta (Koodiesimerkki 16). Jos vastaavuudet löytyvät, tallennetaan löytyneeltä riviltä käyttäjän yksilöllinen ID (user_id) muistiin ja asetetaan se $_SESSION-taulukkoon. Tämän jälkeen käyttäjä ohjataan hallintasovelluksen etusivulle.

```php
<?php
function doLogin()
{
    // Jos sivulla esiintyy virheilmoitus, tallennetaan se muuttujaan.
    $errorMessage = '';
    $userName = $_POST['txtUserName'];
    $password = $_POST['txtPassword'];
    // Tallennetaan kirjautumisaika istuntoon.
    $_SESSION['login_time'] = time();
    // Tarkastetaan, ettei käyttäjätunnus ja salasana -kentät ole tyhjiä.
    if ($userName == '') {
        $errorMessage = 'You must enter your username';
    } else if ($password == '') {
        $errorMessage = 'You must enter the password';
    } else {
        // Tarkastetaan, löytyykö tietokannasta vastaavuudet käyttäjätunnukselle ja salasanalle.
        $sql = "SELECT user_id
                FROM tbl_user
                WHERE user_name = '$userName' AND user_password = PASSWORD('$password')";
        $result = dbQuery($sql);
        // Jos vastaavuudet löytyvät, tallennetaan käyttäjän ID istuntomuuttujaan.
        if (dbNumRows($result) == 1) {
            $row = dbFetchAssoc($result);
            $_SESSION['plaincart_user_id'] = $row['user_id'];

            // Tallennetaan käyttäjän kirjautumisaika tietokantaan.
            $sql = "UPDATE tbl_user
                    SET user_last_login = NOW()
                    WHERE user_id = '{$row['user_id']}'";
            dbQuery($sql);
            // Kun käyttäjätunnus ja salasana ovat täsmänneet, ohjataan käyttäjä
            // hallintasivun etusivulle. Jos käyttäjä on ollut kirjautuneena aikaisemmin,
            // ohjataan tämä edelliselle sivulle, mistä uloskirjautuminen on tapahtunut.
            if (isset($_SESSION['login_return_url'])) {
                header('Location: ' . $_SESSION['login_return_url']);
                exit;
            } else {
                header('Location: index.php');
                exit;
            }
        } else {
            // Annetaan virheilmoitus.
            $errorMessage = 'Wrong username or password';
        }
    }
    // Ei virheilmoituksia.
    return $errorMessage;
}
?>
```

Koodiesimerkki 16. Hallintasovellukseen kirjautuminen.

Mikäli käyttäjätunnusta ja salasanaa vastaavia tunnuksia ei tietokannasta löydy, ilmoitetaan siitä käyttäjälle virheilmoituksella ja estetään kirjautuminen (Kuva 11).



Kuva 11. Käyttäjän kirjautuminen hallintasovellukseen on estetty.

Hallintasovelluksen etusivulta käyttäjä voi tarkastella milloin hänen tunnuksensa on rekisteröity, milloin hän on viimeksi kirjautunut sisään, ja kuinka kauan hän on tällä hetkellä ollut kirjautuneena sovellukseen. Jos hallintasovelluksessa ei liikuta sivulta toiselle tai tehdä mitään muutoksia, kirjaa sovellus käyttäjän automaattisesti ulos 15 minuutin toimettoman ajan jälkeen.

Istuntojen luominen

Sisään kirjautuessa käyttäjälle luodaan neljä uutta sessiomuuttujaa, jos hän ei ole vieraillut sivulla aikaisemmin. Näistä yksi on edellä mainittu user_id, joka tallennetaan $_SESSION-taulukkoon nimellä plaincart_user_id. Muut uudet sessiomuuttujat ovat nimeltään last_login, last_reload ja login_return_url.

Jotta voitaisiin tarkastella käyttäjän kirjautumisaikaa, tallennetaan viimeisin kirjautumisaika last_login–sessioon ja viimeisin sivunlataus last_reload–sessioon. Näille sessioille haetaan sen hetkinen kellonaika palvelimelta PHP:n date()–funktiolla. Viimeisin kirjautumisaika pysyy vakiona, mutta viimeisin sivunlataus vaihtelee käyttäjän liikkumisen ja hänen tekemien sivulatausten mukaan. Sivulatauksen ja kirjautumisajan erotuksella saadaan tietoon käyttäjän toimeton aika hallintasovelluksessa, jolloin hänet kirjataan ulos, jos aika ylittää 15 minuuttia.

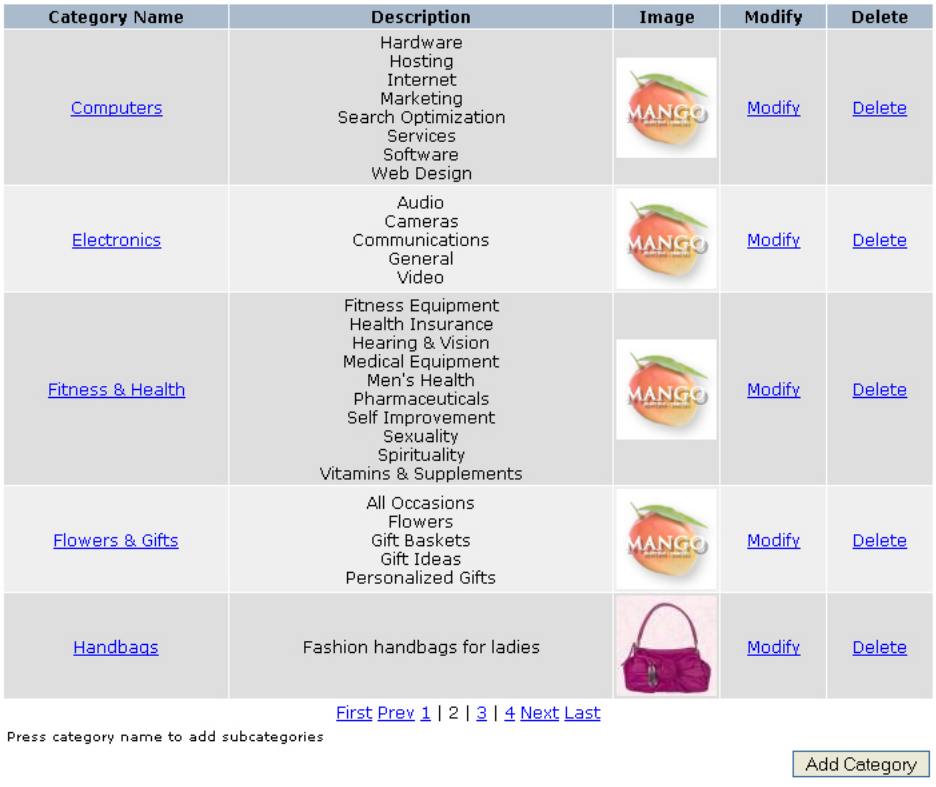
Ylläpitäjän viimeisin sivu, missä hän on vieraillut hallintasovelluksessa, tallennetaan paluuosoite–sessioon (login_return_url). Kun hallintasovellukseen kirjaudutaan, tarkastetaan paluuosoite-sessiosta, onko käyttäjä vieraillut aikaisemmin sovelluksessa. Jos näin on, ohjataan hänet sivulle, jolta uloskirjautuminen tapahtui.

Lukuun ottamatta paluuosoite-sessiota, kaikki sessiot nollataan kun käyttäjä kirjataan ulos automaattisesti tai kun hän kirjautuu itse ulos. Sovelluksessa tarkastetaan myös, löytyykö käyttäjän yksilöivä tunnus plaincart_user_id $_SESSION-taulukosta. Jos tunnusta ei löydy, ohjataan käyttäjä kirjautumissivulle. Tällä varmistetaan, ettei hallintasovellukseen voida palata takaisin selaimen takaisinnapilla, kun ylläpitäjäkäyttäjä on kirjautunut ulos sovelluksesta.

5.4 Hallinnointisivut

5.4.1 Kategoriasivu

Kategoriasivulla sivujen ylläpitäjä voi lisätä, muokata ja poistaa verkkokaupan tuotekategorioita. Jokaiselle uudelle tuotekategorialle on luotava alakategoria, jotta sille voidaan lisätä tuotteita. Tuotekategoriat listautuvat automaattisesti käyttäjälle, kun tämä navigoi kategoriasivulle (Kuva 12).

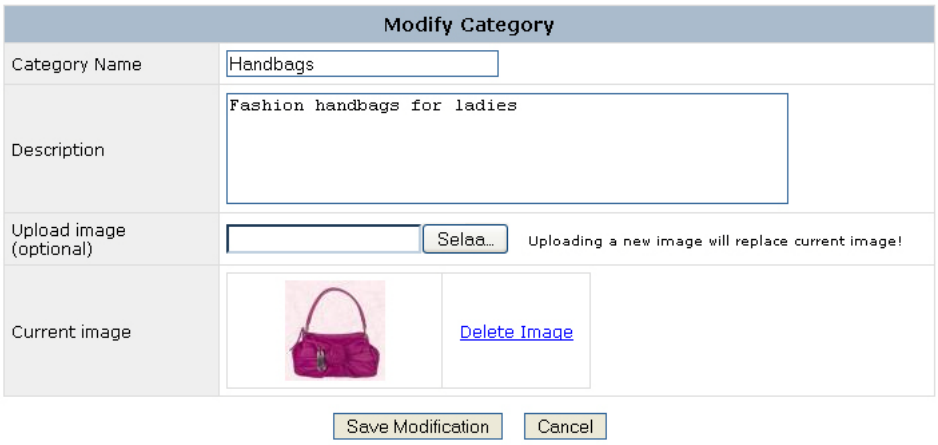


Kuva 12. Tuotekategorioiden listaus hallintasovelluksessa.

Ylläpitäjän lisäämät kategoriat lisätään tietokantaan tbl_category-tauluun. Lisäys ja muokkaus tapahtuvat erillisiltä sivuilta, joissa käyttäjää pyydetään täyttämään vaaditut kentät. Kategorian lisäyksessä ylläpitäjän tulee syöttää kategorialle nimi ja kategoriaselostus. Kategoriaselostus ei näy asiakkaalle, mutta se luotiin helpottamaan sivujen ylläpitämistä. Kategorialistauksessa kategoriaselostus näkyy ylläpitäjälle ja helpottaa tuotteiden lisäämistä oikeisiin kategorioihin.

Ylläpitäjä voi myös lisätä kategorialle tuotekuvan, jonka lisäämisestä kerrotaan tarkemmin seuraavissa kappaleissa. Jos ylläpitäjä ei kuitenkaan lisää kuvaa, lisätään kategorialle automaattisesti verkkokaupan oletuskuva.

Kategorian muokkauksessa käyttäjä voi vaihtaa kategorianimeä ja tuoteselostusta, sekä vaihtaa kategoriakuvaa tai poistaa sen (Kuva 13). Jos jokin kategoria halutaan poistaa, onnistuu se kategorioiden pääsivulta. Mikäli poistettava kategoria sisältää tuotteita, poistetaan myös ne tietokannasta, mutta vasta käyttäjän hyväksynnän jälkeen.



Kuva 13. Kategorian muokkaus.

Kategoriakuvan lisääminen

Kategoriakuva voidaan lisätä uuden kategorian lisäämisvaiheessa tai jälkeenpäin kategorian muokkaussivulta. Käyttäjän lisäämä kuva haetaan POST-muuttujaan ja kuvan tiedot tallennetaan taulukkotyyppiseen muuttujaan $_FILES, joka sisältää tiedot muun muassa kuvan alkuperäisestä nimestä ja tiedostokoosta.

Kun käyttäjä haluaa tallentaa kategoriakuvan kategorialle, ajetaan PHP-funktio uploadImage() (Koodiesimerkki 17). Funktiolle määritellään parametreiksi POST-muuuttujalla haettu kuva ja kohdekansio palvelimella.

```php
<?php
// Syötetään funktiolle tarvittavat parametrit
// (mistä kentästä tiedosto haetaan ja mihin kansioon se tallennetaan)
$catImage = uploadImage('fleImage', SRV_ROOT . 'images/category/');

function uploadImage($inputName, $uploadDir)
{
    // Tallennetaan FILES-taulukkomuuttujan tiedot
    $image     = $_FILES[$inputName];
    $imagePath = '';

    // Tarkastetaan, onko tiedosto annettu
    if (trim($image['tmp_name']) != '')
    {
        // Haetaan kuvatiedoston tiedostopääte (esimerkiksi .jpg)
        $ext = substr(strrchr($image['name'], "."), 1);

        // Luodaan kuvatiedostolle satunnainen tiedostonimi,
        // jonka luomiseen käytetään MD5-koodausta.
        $imagePath = md5(rand() * time()) . ".$ext";

        // Tarkastetaan kuvan koko pikseleissä
        $size = getimagesize($image['tmp_name']);
        // Jos kuvakoko ylittää maksimimäärityksen,
        // ajetaan seuraava funktio, joka pienentää kuvaa automaattisesti
        // ja tallentaa sen palvelimelle käyttäjän määrittelemään kansioon
        if ($size[0] > MAX_CATEGORY_IMAGE_WIDTH) {
            $imagePath = createThumbnail($image['tmp_name'],
                         $uploadDir . $imagePath, MAX_CATEGORY_IMAGE_WIDTH);
        } else {
            // Jos kuvaa ei tarvitse pienentää, siirretään se palvelimelle kansioon
            if (!move_uploaded_file($image['tmp_name'],
                $uploadDir . $imagePath)) {
                $imagePath = '';
            }
        }
    }
    // Palautetaan tiedostopolku
    return $imagePath;
}
?>
```

Koodiesimerkki 17. Kuvatiedoston lisääminen kategorialle.

UploadImage()-funktion tarkoituksena on luoda kuvatiedostolle satunnainen tiedostonimi ja tallentaa tiedosto palvelimelle. Tiedostonimen satunnaiseen luomiseen käytetään MD5-algoritmia (Message Digest), joka tapahtuu PHP:ssa md5()-funktiolla (Koodiesimerkki 18). Funktiolle annetaan parametriksi rand()-funktiolla saadun satunnaisen kokonaisluvun ja time()-funktiolla palvelimelta saadun aikaleiman kerroin. Rand()- ja time()-funktioiden toimintaa en tarkastele tarkemmin. Md5()-funktiolla ja edellä mainitulla parametrillä tiedostonimestä tulee satunnainen heksadesimaaleiksi käännetty 32-merkkinen merkkijono.

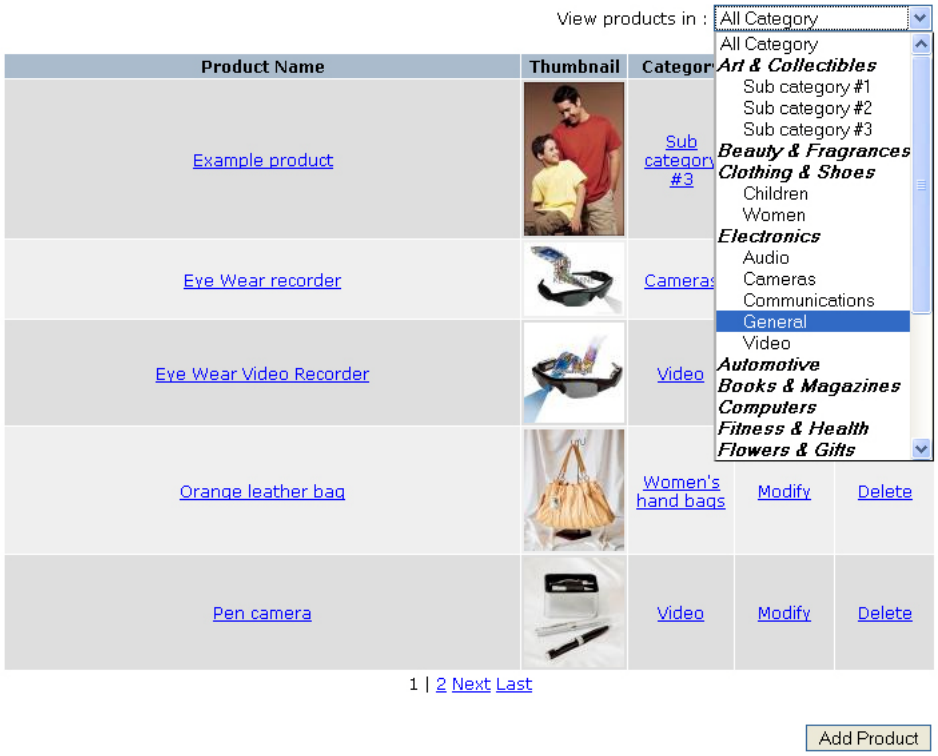
```php
<?php
$str = "Esimerkki";
echo md5($str);
// Tulostaa satunnaisen merkkijonon,
// esimerkiksi:
// 8b1a9953c4611296a827abf8c47804d7
?>
```

Koodiesimerkki 18. MD5-algoritmin käyttö

Tiedostonimen luomisen jälkeen uploadImage()-funktio tarkastaa PHP:n getimagesize()-funktiolla tiedoston pikselikoon ja vertaa sitä verkkokaupan asetustiedostossa olevaan määritykseen kategoriakuvatiedoston maksimi pikselikoosta. Jos koko on pienempi kuin asetuksissa määritelty, tallennetaan kuva palvelimelle. Jos koko ylittää määrityksen, jatketaan kuvatiedoston muokkaamista seuraavassa funktiossa, joka automaattisesti pienentää kuvakoon vastaamaan asetustiedostossa olevaa määritystä maksimi kategoriakuvan pikselikoosta. Tämän jälkeen kuva tallennetaan palvelimelle määriteltyyn kansioon.

5.4.2 Tuotesivu

Tuotesivulla listataan ylläpitäjän näkyviin kaikki verkkokauppaan lisätyt tuotteet. Listauksessa näkyvät tuotenimi, tuotekuva, tuotteen sisältävän kategorian nimi sekä linkit tuotteen muokkaamiseen, poistamiseen ja uuden tuotteen lisäämiseen (Kuva 14). Tuotteet sijaitsevat tietokannassa tbl_product–taulussa ja niiden tarkemmat koko-, väri- ja varastotiedot tbl_product_info–taulussa.



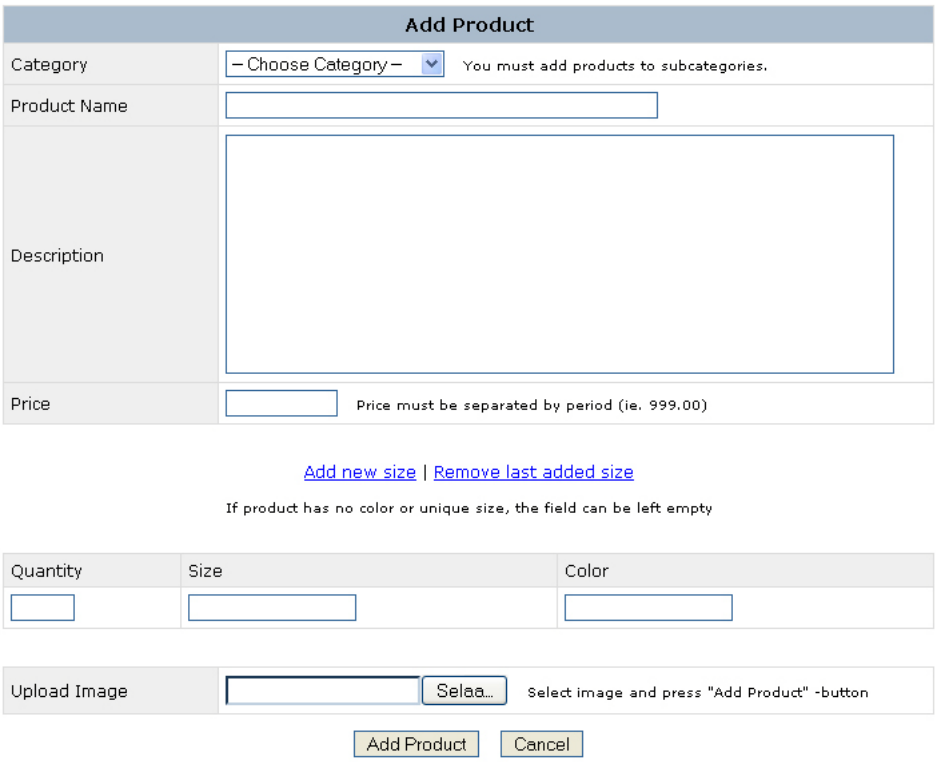
Kuva 14. Tuotelistaus. Alasvetovalikosta ylläpitäjä voi valita, minkä tuotekategorian tuotteet tuodaan esille.

Uuden tuotteen lisääminen

Uuden tuotteen lisäyksessä ylläpitäjän tulee valita tuotteelle tuotekategoria valikosta, johon listautuu automaattisesti kaikki verkkokauppaan lisätyt kategoriat. Valikosta ei voida valita pääkategoriaa, vaan tuote on lisättävä johonkin alakategoriaan.

Pakollisia täytettäviä tietoja tuotteen lisäämisen onnistumiseksi ovat tuotekategorian valinnan lisäksi tuotenimi, tuotteen esittelyteksti, tuotehinta, tuotteen varastosaldo sekä vaihtoehtoisesti joko tuoteväri tai tuotekoko (Kuva 15). Myös tuotekuvan lisääminen on vapaaehtoista ja jos sitä ei lisätä, lisätään tuotteelle sama verkkokaupan oletuskuva kuin kategorioille. Jos jokin vaadittavista kentistä jäte-

tään tyhjäksi, ilmoitetaan siitä ylläpitäjälle JavaScriptilla tuotetulla ponnahdusikkunalla, ja estetään tuotteen tallentaminen tietokantaan.



Kuva 15. Uuden tuotteen lisääminen verkkokauppaan.

Tuotekuvan lisääminen on toteutettu samalla tavalla kuin kategoriakuvan lisääminen, mutta poikkeuksena on, että tuotekuvasta luodaan esikatselukuva sekä tarkennettu kuva, jotka tallennetaan palvelimelle.

Tarkempien tuotetietojen lisääminen

Kun ylläpitäjä lisää uutta tuotetta tai muokkaa jo olemassa olevaa, tulee hänen lisätä tuotteelle tarkemmat tuotetiedot, eli koko- ja väritiedot sekä kappalemäärän varastossa. Tarkempia tietoja voidaan lisätä niin monta kuin halutaan, eli yhdelle tuotteelle voidaan esimerkiksi lisätä 10 eri koko- ja värivaihtoehtoa.

Tarkempien tuotetietojen lisääminen on toteutettu JavaScript jQuery-kirjaston append()-funktiolla (Koodiesimerkki 19). Tätä funktiota on käytetty, koska ei voida olla varmoja siitä, kuinka monta eri koko- ja värivaihtoehtoa kukin tuote saa, kun ylläpitäjä lisää sen tietokantaan.

```
var count = 1;
$(function(){
    var i = 1;
    $('span#add_field').click(function(){
        count += 1;
        $('#container').append(
        '<table width="100%" border="0" align="center" id="deletelink_' + count + '">'
        +'<tr>'
        +'<td class="label">Quantity</td>'
        +'<td class="label">Size</td>'
        +'<td class="label">Color</td>'
        +'</tr>'
        +'<tr>'
        +'<td class="content">'
        +'<input type="hidden" name="counter_new" value="' + count + '" />'
        +'<input id="txtQty" type="text" name="txtQty_' + count + '" size="4" maxlength="10" /> </td>'
        +'<td class="content">'
        +'<input id="txtSize" type="text" name="txtSize_' + count + '" size="17" /> </td>'
        +'<td class="content">'
        +'<input id="txtSizeColor" type="text" name="txtSizeColor_' + count + '" size="17" /></td>'
        +'<input name="count" type="hidden" value="add" />'
        +'</tr>'
        +'</table>' );
        i++;
    });
});
```

Koodiesimerkki 19. jQuery-funktio, joka tulostaa kolme HTML-kenttää käyttäjän painaessa linkkiä.

Append()-funktion avulla käyttäjä voi linkkiä painaessaan lisätä aina uudet HTML-kentät uutta koko- ja värivaihtoehtoa varten, ja näin ollen sivulle ei ladata koskaan tyhjiä kenttiä, joita ei välttämättä täytettäisi. Jos käyttäjä lisää liikaa kenttiä, voi hän halutessaan poistaa viimeisimmän lisäyksen erillisestä linkistä. Linkkejä painaessa sivua ei ladata uudestaan, vaan kenttien lisääminen ja poistaminen tapahtuu dynaamisesti.

5.4.3 Tilauksien hallinta

Verkkokaupan käyttäjien tekemät tilaukset tallentuvat aina onnistuneen tilauksen jälkeen tietokantaan. Tilauksen asiakastiedot tallentuvat tbl_order–tauluun ja tilatut tuotteet tbl_order_item–tauluun. Tilaukset listataan verkkokaupan hallintasovelluksessa erilliselle tilausten käsittelysivulle (Kuva 16). Tälle sivulle, ja tilauksen tarkempiin tietoihin, verkkokaupan ylläpitäjä saa sähköpostissa linkin aina uuden tilauksen tullessa.

View All

Order #	Customer Name	Amount	Order Time	Status
1117	Testi Testi	€150.60	2010-03-25 19:14:34	Paid
1114	Viku Saini	€150.60	2010-03-14 16:44:23	New

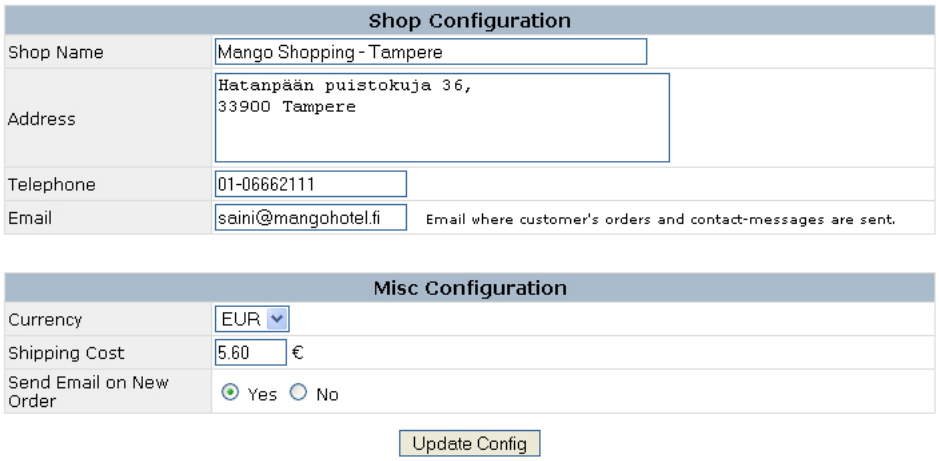
Kuva 16. Tilauksien listaus.

Tilaukset on lajiteltu sivustolla niiden statuksien mukaan. Statuksia ovat “New” (uusi tilaus), “Paid” (maksettu tilaus), “Shipped” (toimitettu tilaus), “Completed” (lopullisesti käsitelty tilaus) ja “Cancelled” (peruutettu tilaus). Oletuksena sivu näyttää aina uudet tilaukset. Jokaisella tilauksella on oma yksilöllinen tilausnumero, jonka avulla voidaan tarkastella tilauksen tarkempia tietoja.

Tarkemmissa tiedoissa ylläpitäjä näkee milloin tilaus on tehty sekä käyttäjän syöttämät tilaus- ja maksutiedot ja hänen tilaamansa tuotteet. Ylläpitäjä voi vaihtaa tilauksen statusta erillisestä valikosta sen mukaan, miten tilauksen käsittely on edennyt.

5.4.4 Verkkokaupan asetussivu

Verkkokaupan asetusten muuttaminen tapahtuu asetussivulta, josta voidaan muokata verkkokaupan nimeä, osoitetta, puhelinnumeroa, ylläpitäjän sähköpostiosoitetta, kaupan käyttämää valuuttamuotoa, postimaksun suuruutta sekä sähköpostin lähetystä uusissa tilauksissa (Kuva 17).



Kuva 17. Verkkokaupan muokattavat asetukset.

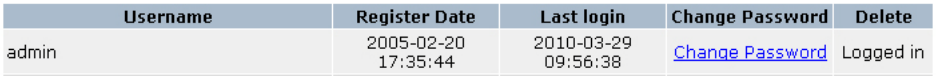
Nimen, osoitteen, puhelinnumeron ja sähköpostiosoitteen muokkaukset päivittyvät automaattisesti myös verkkokauppasovelluksen yhteystiedot -sivulle. Valuuttamuodon muutos vaikuttaa tuotehintojen valuuttamerkin vaihtoon ja postimaksun suuruus päivittyy automaattisesti ostoskorien lopullisen laskun laskukaavaan.

Jos ylläpitäjä haluaa tarkastella uusia tilauksia vain hallintasovelluksen kautta, eikä halua niistä erillistä sähköpostia itselleen, voi hän ottaa sähköpostin lähetyksen

uusista tilauksista pois päältä. Sähköposti, johon uusista tilauksista lähetetään ilmoitus, on sama kuin asetussivulla muokattavissa oleva.

5.4.5 Ylläpitäjäkäyttäjien hallinta

Koska verkkokaupalla voi olla hyvin useita eri ylläpitäjiä, on heitä mahdollista muokata, lisätä ja poistaa ylläpitäjäkäyttäjien hallintasivulta. Sivulla näkyvät tiedot tällä hetkellä luoduista käyttäjistä, heidän rekisteröintipäivistään ja viimeisimmistä kirjautumispäivistä (Kuva 18). Sivulla on myös linkit erikseen jokaisen käyttäjän salasanan vaihtoon sekä käyttäjän poistamiseen.



Username	Register Date	Last login	Change Password	Delete
admin	2005-02-20 17:35:44	2010-03-29 09:56:38	Change Password	Logged in

Kuva 18. Verkkokaupan ylläpitäjä voi tarkastella käyttäjätunnuksensa tietoja erilliseltä sivulta.

Kuka tahansa verkkokaupan ylläpitäjä voi lisätä uuden ylläpitäjän verkkokauppaan. Myös kuka tahansa ylläpitäjä voi poistaa kenet tahansa ylläpitäjistä, mutta ei itseään. Ylläpitäjä voi vaihtaa oman salasanansa ollessaan kirjautuneena sovellukseen.

Jos halutaan lisätä uusi käyttäjä, tarvitsee hänelle luoda käyttäjänimi ja salasana. Koska tietokannassa on määritelty, että käyttäjänimi on uniikki, ei kahta samanlaista ylläpitäjän nimeä voi esiintyä. Jos käyttäjänimeksi kuitenkin syötetään jo olemassa oleva nimi, ilmoitetaan siitä virheilmoituksella ja pyydetään vaihtamaan nimeä.

Ylläpitäjäkäyttäjän lisäys- ja muokkausvaiheessa annettava salasana tulee olla vähintään kuusi merkkiä pitkä ja se tulee sisältää numeroita ja kirjaimia. Tämä tarkastus tehdään aina, kun käyttäjä yrittää lisätä uutta tai muokata jo olemassa olevaa ylläpitäjää.

6 Loppusanat

6.1 Kehitettävää

Mango Hotellin verkkokauppaan jäi paljon kehitettävää ja työn edetessä nousikin esiin useita erilaisia lisäominaisuuksia, joita kaupassa saatetaan tarvita, mutta aika ei riittänyt niiden toteuttamiseen. Verkkokaupan edelleen kehitys on kuitenkin varmaa, joten lisäominaisuuksia tullaan lisäämään sovellukseen vielä tulevaisuudessa.

Lisäominaisuuksia tuli ilmi työn edetessä ja keräsin niistä mielestäni tärkeimmät seuraaviin kappaleisiin. Ominaisuudet eivät ole tärkeysjärjestyksessä.

Asiakasrekisteri

Verkkokaupan asiakas voisi halutessaan rekisteröityä sovellukseen, jolloin hänellä olisi mahdollisuus ylläpitää omia henkilö- ja tilaustietoja sekä käyttäjätunnusta ja salasanaa. Hän voisi omalta henkilökohtaiselta sivultaan seurata tekemiään tilauksia. Rekisteröityneillä asiakkailla olisi myös mahdollisuus toivoa tuotteita, jotka ovat loppuneet varastosta. Näin ollen verkkokaupan ylläpitäjä voisi hankkia varastoon lisää tuotteita, joiden kysyntä on kasvanut.

Asiakasrekisterin hallinta

Jos verkkokaupalle luodaan asiakasrekisteri, tulisi sitä pystyä hallitsemaan ylläpitäjän hallintasovelluksesta. Hallintasovelluksessa ylläpitäjä näkisi kaikki rekisteröityneet asiakkaat, heidän henkilö- ja tilaustietonsa sekä asiakkaiden tekemät tilaukset. Ylläpitäjä voisi hallita rekisteröityneiden asiakkaiden tilauksia muuttamalla niiden statusta, jolloin sovellus lähettäisi automaattisesti asiakkaalle sähköpostia hänen tilauksensa tilanteen muuttumisesta.

Ylläpitäjäkäyttäjien tasot

Ylläpitäjäkäyttäjille voisi luoda käyttäjätasot, jolloin esimerkiksi yksi ylläpitäjä olisi ylin kaikista ylläpitäjäkäyttäjistä ja vain hänellä olisi mahdollisuus poistaa ylläpitäjäkäyttäjiä sovelluksesta. Ylin ylläpitäjä olisi myös ainoa käyttäjä, joka voisi muokata verkkokaupan asetuksia ja hallinnoida rekisteröityneitä asiakkaita.

Tuotekategorioiden tiedot näkyviin asiakkaille

Hallintasovelluksessa tuotekategorioille lisättävät kategoriaselostukset näkyisivät asiakkaille verkkokauppasovelluksessa esimerkiksi silloin, kun he liikuttavat hiiren tietyn tuotekategorialinkin päälle. Tämän ansiosta asiakas näkisi heti, minkälaisia tuotteita kyseinen tuotekategoria sisältää.

Tuotteiden esilletuonti

Verkkokauppasovelluksessa asiakas saisi kerralla näkyviin kaikki verkkokaupassa olevat tuotteet, jolloin hän voisi selata niitä ilman liikkumista tuotekategoriasta toiseen. Hänellä olisi myös mahdollista karsia listauksesta pois sellaiset tuotteet, joita ei varastossa ole.

Ylläpitäjällä olisi mahdollista lisätä tuotteille enemmän kuin yksi tuotekuva. Tämä mahdollistaisi tuotekuvien ottamisen eri kuvakulmista. Tuotekuvat listautuisivat pieninä kuvina tuotteen tarkempiin tietoihin, jolloin käyttäjä voisi valita, mitä kuvaa hän haluaa tarkastella.

6.2 Yhteenveto

Työssäni suunnittelin ja toteutin hotellialalla toimivalle Mango Hotel –yritykselle verkkokaupan, jossa myydään vaatteita, tavaroita ja asusteita. Yrityksellä oli ennen tätä työtä valmiista pohjasta toteutettu verkkokauppa, mutta se ei toiminut läheskään niin kuin yritys halusi. Tästä syystä toteutin tämän työn verkkokaupan käyttämättä tai muokkaamatta mitään valmista verkkokauppapohjaa, jotta työn lopputulos vastaisi mahdollisimman hyvin toimeksiantajani toiveita.

Valitsin työni toteutustavaksi PHP-ohjelmoinnin sekä relaatiotietokannan hallitsemiseen MySQL-hallintajärjestelmän. Ulkoasun toteutin W3-standardeilla XHTML-määrityksillä ja CSS-tyyliohjeilla. Jotta sivuille saatiin dynaamisuutta, käytin JavaScriptia ja siihen kuuluvaa jQuery-kirjastoa ulkoasun paranteluun. PHP, MySQL, XHTML ja CSS olivat minulle ennestään tuttuja kieliä suoritetuista opinnoistani. JavaScriptia, ja erityisesti jQuerya, opin työtä tehdessäni hyvin paljon.

Verkkokaupan toteutus oli mielestäni hyvin haastava ja pitkä projekti, koska minulla ei ollut aikaisempaa kokemusta näin laajan projektin toteuttamisesta ja jouduin tekemään sen yksin. Työ sopi kuitenkin erityisen hyvin opinnäytetyöaiheeksi. Haastavinta verkkokaupan toteutuksessa oli miettiä, miten sen tulisi toimia, ja

miten toiminta voitaisiin rakentaa PHP-ohjelmoinnilla. Työssä kuluikin aikaa eniten verkkokaupan testauksessa sekä toimivan ja järkevän ohjelmointikoodin suunnittelussa ja luomisessa.

Omasta mielestäni työni lopputulos vastasi hyvin toimeksiantajani toiveita. Yksi onnistuneimmista lopputuloksista oli ylläpitäjän hallintasovellus, jonka kautta esimerkiksi tuotteiden hallinta on yksinkertaista ja nopeaa. Sen ansiosta ylläpitäjän työ on vaivatonta, eikä hallintasovelluksen käyttö vaadi erityistä tietämystä ohjelmoinnista tai muusta web-ympäristöstä.

Onnistuin mielestäni luomaan myös helppokäyttöisen, sekä erityisesti toimivan rakenteen verkkokaupalle käyttämättä mitään valmista pohjaa. Tämä rakenne, sekä myös verkkokaupan ulkoasu, ovat eduksi niin verkkokaupan asiakkaille kuin sen jatkokehitykselleki tulevaisuudessa. Verkkokaupan käyttöliittymä toimivalla ulkoasulla huolehtii asiakkaiden viihtyvyydestä ja heidän ostosten tekemisen helppoudesta. Verkkokaupan rakenne tekee myös sen edelleen kehityksen tulevaisuudessa mahdolliseksi.

Koska tämän työn verkkokauppa toteutettiin Mango Hotellin käyttöön sen vaatimusten mukaisesi, uskon sen tuovan taloudellista hyötyä yritykselle. Työtunteja säästetään jo pelkästään yksinkertaisen hallintajärjestelmän ansiosta, mikä aiheutti ennen käytössä olleessa verkkokauppapohjassa ylimääräisiä työtunteja päivitettävyysongelmien vuoksi. Jos verkkokaupassa esiintyy ongelmia, voi Mango Hotelli olla yhteydessä työn tekijään, ja ratkaista ongelmat yhteistyöllä. Tällä tavalla yritys säästää työtunteja, koska he eivät joudu itse ratkomaan ongelmia.

Vaikka verkkokaupan suunnittelu ja toteutus oli haastava ja pitkä työ, niin se kuitenkin palkitsi, sillä uuden sovelluksen kehittäminen alusta loppuun oli aivan uusi asia itselleni. Opin paljon PHP-ohjelmoinnista ja erityisesti sen valmiiden funktioiden käyttämisestä, sekä uutena asiana JavaScript jQuery-kirjaston käytön. Opinnäytetyössäni huomasin, kuinka tärkeässä asemassa huolellinen suunnittelu ja testaaminen ovat ohjelmoitaessa hyvin laajoja sovelluksia.

Lähdeluettelo

Painetut lähteet

Gilmore, Jason W. 2005. PHP & MySQL – Tehokas hallinta, suomennus. Suomentaja: Kuvaja, Arto. Helsinki: Readme.fi.

Heinisuo, Rami & Rauta, Ilkka 2007. PHP ja MySQL Tietokantapohjaiset verkko palvelut. Valikko-sarja. Helsinki: Talentum Media Oy.

Linjama, Tero 2001. XHTML. Jyväskylä: Docendo Finland Oy.

Peltomäki, Juha 2000. JavaScript. Jyväskylä: Teknolit Oy.

Sähköiset lähteet

Cascading Style Sheets 2010. [online] [viitattu 24.02.2010]. Saatavissa: http://en.wikipedia.org/wiki/Cascading_Style_Sheets

Ekonoja, Antti, Lahtonen, Tommi & Mäntylä, Jukka 2004. Relaatiotietokannat. [online] [viitattu 02.03.2010]. Saatavissa: http://appro.mit.jyu.fi/doc/tiedonhallinta/tietokannat/

JavaScript 2010. [online] [viitattu 26.02.2010]. Saatavissa: http://en.wikipedia.org/wiki/JavaScript

JavaScript library 2010. [online] [viitattu 24.02.2010]. Saatavissa: http://en.wikipedia.org/wiki/JavaScript_library

Lindley, Cody 2009. JQuery Cookbook. O'Reilly Media, Inc. ISBN 978-059-615-977-1. [online] [viitattu 24.02.2010]. Saatavissa: http://books.google.fi/books?id=7kfyzf2BnP0C

PHP 2009. [online] [viitattu 26.02.2010]. Saatavissa: http://fi.wikipedia.org/wiki/PHP

PHP.NET 2010. $_SERVER. [online] [viitattu 12.03.2010]. Saatavissa: http://php.net/manual/en/reserved.variables.server.php

Smilehouse Oy 2010. Kuluttajien verkko-ostaminen on kasvanut taantumassakin kertoo tuore tutkimus. [online] [viitattu 23.02.2010]. Saatavissa: http://www.smilehouse.fi/uutiset/2010-01-14-kuluttajien-verkko-ostaminen-on-kasvanut-taantumassakin-kertoo-tuore-tutkimus

Snook, Jonathan, Gustafson, Aaron, Langridge, Stuart & Webb, Dan 2007. Accelerated DOM Scripting with Ajax, APIs, and Libraries. Apress. ISBN 978-159-059-764-4. [online] [viitattu 25.02.2010]. Saatavissa: http://books.google.com/books?id=COaAs3dZ0rsC

W3 Schools 2010a. Browser Statistics. [online] [viitattu 29.03.2010]. Saatavissa: http://www.w3schools.com/browsers/browsers_stats.asp

W3 Schools 2010b. HTML script tag. [online] [viitattu 26.02.2010]. Saatavissa: http://www.w3schools.com/TAGS/tag_script.asp

W3C XHTML 2002. XHTML 1.0: The Extensible HyperText Markup Language (Second Edition). [online] [viitattu 26.02.2010]. Saatavissa: http://www.w3.org/TR/xhtml1/

XHTML 2009. [online] [viitattu 26.02.2010]. Saatavissa: http://fi.wikipedia.org/wiki/XHTML