UNIVERSITY OF GOTHENBURG

Designing content distribution with trading options - "Yank Yank": a prototype system

Master of Science Thesis in the program of Computer Science

Peter Henriksson

University of Gothenburg
Chalmers University of Technology
Department of Computing Science and Engineering
GÖTEBORG, Sweden, December 2009

Designing content distribution with trading options - "Yank Yank": a prototype system

Peter Henriksson

Examiner: Marina Papatriantafilou

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden

Abstract

There exist several market-places on the Internet where authors offer free and premium content to end-users for download. The published content differs but might be articles, music, photos, books etc, all depending on the end-users needs. Essentially such marketplaces are influenced by both content management systems (CMS) and e-commerce. General characteristics of such systems offer authors a central way to upload and publish their digital assets to the public, without having the actual technical knowledge of doing so. There are however not many systems that can offer both authors and end-users the benefit of a fair trade from selling and buying content due to the small margin in micropayments. On commission by the IP-telephony provider Cellip this master thesis is about investigation and implementing the basics and logic of such a system. The thesis starts with a introduction of general CMS and e-commerce marketplaces. Proceeding it explains an structure of the system, its functionality and how the iterative procedure of the development process made progress. The thesis resulted in a mixed CMS/e-commerce system where authors can promote, upload and publish their content, where end-users can create accounts, search for, rate and buy content. The system also includes an administration interface for managing.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis is about how to build a system where authors can create, publish, present digital content to the public. The system has similarities in content management systems because the system needs of an authoring tool and the work-flow model that content management systems provide. All created content in the system needs to be distributed to the public in an efficient way. By putting created content directly into the system database like in simple database systems it would in growth with increasing users soon become a bottleneck. Thus created content should be distributed by using a Content Distribution Network (CDN) technology. The system itself is server based and therefore centralised which differs from P2P networks. The work includes a literature investigation on content management systems and content distribution. The result from this study underlies the choice of structure and components of this system.

1.1 Background

There are many market-places available on the Internet. However, very few will cater for re-attracting the occasional visitor and pleasing her with both free and low-priced premium content; free to let her try anything out, and low-priced premium content to supply good value for the money. There are many sites that will provide sellers of high-priced goods and content an efficient market-place, but extremely few for offering authors of low-priced premium content to actually get paid without the royalty-owner seeing that all possible profit has been munched up. In particular, there are no sites that offer both visitor and royalty-owner an equal beneficial proposition. A major reason for this is that micro-payments are difficult to make profitable which is why most means of smaller payments (premium-sms, premium-calls e.g.) will make most of the margin for the royalty-owner disappear. Up until now, all other micro-payment mechanisms have failed due to the difficulty of changing end-user behaviour

1.2 Purpose

The purpose of this thesis is to investigate and learn how to implement a system where

1. Royalty-owners of electronic goods such as music, e-books, photo etc can
   - promote their content

   - sell their content and
   - get a fair return even for micro-payments

2. End-users can reap the benefits of a vast amount of free and premium content
3. Cellip would be getting a small margin for the micro-payments, free or low-cost marketing and new customers.

So in general the main goal is to build a system that can create web pages dynamically. The creation of these pages will depend on the specific user of the system. A research for existing similar systems need to be done and if there exist any, investigate and present how the functionality from those systems can be derived and applied to this system. Another goal is to investigate how the content of the system should be distributed to the end-users in a convenient way.

1.3 System requirement specification

As a start the system should provide following functions

- Creating accounts for Royalty-owners and End-users.
- Login interface for both groups of users.
- Login interface for the administrator.
- A tool for searching content.
- Preview or try free content.
- View royalty-owners homepage.
- Ways to rate content.
- Purchase process for buying content.
- The royalty-owner interface shall have functions to:
  - upload content.
  - edit content.
  - list uploaded and accepted content.
  - promote their self and their content.
  - edit homepage.
  - add news to homepage.
  - view account info.
- The end-user interface shall include functions to:
  - make a payment to their account.
  - view purchase history.
- The administration interface shall include functions to:

- search for content and users.
- delete content and users.
- view submitted content
- grant or reject submitted content.

1.4 Restrictions

Due to the size of the project it was decided to leave the billing and payment processes as *dummy* processes. Thus the thesis describes what type of billing and payment mechanisms that can be used in this system but not implemented. It was also desirable that end-users should be able to become reviewers, these functions are not implemented and are left to the administrators interface. There are no automated shortening processes for uploaded content since this was expected to take to much time in demand. The system also only provide uploaded content which is neither free or premium.

Chapter 2

Technical Background

In this chapter a general description of content management systems is given with focus on the work-flow model. A description of payments methods and how to distribute content is covered. An explanation of the technical tools used for the implementation are also described as well.

2.1 Content Management Systems

Now days the Internet is overflowed with information and content from companies and many other sources. To be attractive for end-users it put high demands on that the information is up to date and accurate.

Many company sites has grown out of their hands and the web masters have become bottlenecks due to growing process of keeping the site updated. Thus, the needs for a system to dynamically creating web pages where the content creators can publish their content became bigger then ever. This is called the *Business problem*. A solution to this problem is called a Content Management System (CMS). A CMS manages the entire life cycle of web pages from creation to archive Robertsson[1]. A CMS provides following business benefits:

- Easy creating of new pages
- Faster updates
- Greater consistency
- Decentralised authoring
- Less duplication
- Reduced costs

Robertsson[1] describes how the functionality of a CMS can be divided into four main categories.

- Content creation
- Content management
- Publishing
- Presentation

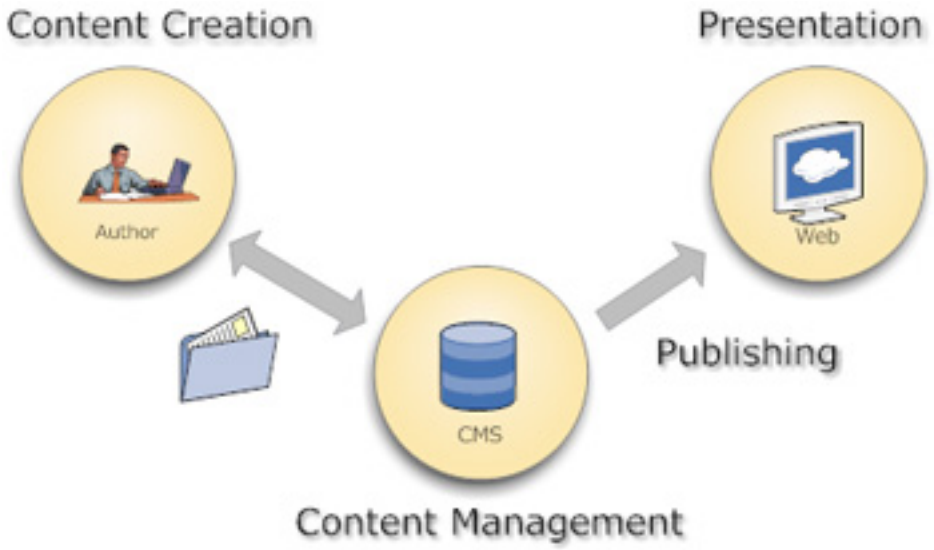


Figure 2.1: CMS functionality

2.1.1 Content creation

Content creation is a CMS authoring tool the provide mechanisms for the creation of new pages or updates without having the technical knowledge. The authoring tool is the CMS key to success since it can be divided itself into the business of a company. Thus, it can be divided by the managers for each department of a company which gives a decentralised authoring.

2.1.2 Content management

This describes a central repository in the CMS where all new data and pages are put. The system keep track of all pages and logging who has made changes to a particular document. Also ensuring that content can only be manipulated by authorised people at corresponding section. The CMS also provides an authoring work flow, thus once that new content is created it is automatically sent to a manager for approval before publishing.

2.1.3 Publishing

Now the created content are ready for publishing. The content might be published at several sources provided by the CMS. Each page might have its own design, so the cms provides ways for web designers to design each page. This mean that consistency of the look of a page is maintained. So authors does not need to worry about layout and design thus, they can concentrate on writing.

2.1.4 Presentation

The CMS provides tools for designers and web developers to bring a consistent look and feel for each site the CMS manages. Same content can be published over several sites but with each site having its own look. So the CMS totally automates the publishing process which gives the authors more time to concentrate on creating content.

2.2 E-commerce systems

Purchases between a companies and customers on the Internet are generally denoted as e-commerce. E-commerce can be divided into two branches namely, net shops and net auctions.

The characteristics of net shops is that they consist of a stock and a website. On the website customers can search for products in the stock and make an order. E-commerce websites are often supplied with a shopping-cart to where customers can add products.

Net auctions reminds of regular auctions but with the difference that the traditional auctioneer functionality is maintained by a website. Net auction vendors can be both companies and private persons.

E-commerce systems also often provides several types of payment mechanisms for the customers to make their purchases. Some e-commerce systems depending on the product also give the customer the opportunity to download the purchased product.[10]

2.3 Payment mechanisms

There exists different types of payment mechanisms for e-commerce, direct payments by credit card, invoice and cash on delivery. Most of the e-commerce websites offer customers to make purchases by credit cards. This put high demands on security. One way to provide such a service is to use a Payment Service Provider(PSP). A PSP is a company that is connected to several billing agents and provide the e-commerce websites with an interface to obtain access to those. A PSP works and is similar as a card terminal in a physical store. By using a PSP the e-commerce sites do not need to take care of the responsible of handling credit card numbers and sensitive personal information.[11]

2.4 Content distribution networks

2.4.1 Background

Internet-users today put high demands in the availability and reliability of websites, it can be of crucial importance whether or not the end-users decides to stay or choose another website. Especially commercial websites offering digital content needs to have fast response time, low latency and provide end-users downloading at high streaming rates. So what approach should be initialised to achieve those demands? Letting a single server take care of all requests would lead to a very congested link and then became a bottleneck. Also if a user are not geographically close to the content the data-packets would have to travel thru several ISPs which will likely make cost in delay and loss of data-packets. Popular content that is often requested will be sent many times thru the same links.

Having this in mind the main approach would be to cache content geographically close to the end-users. This can be achieved by using content distribution networks.

2.4.2 Architecture

Content distribution networks (CDNs) can be described as a set of computers connected thru the Internet. To do this in an efficient way the net load are balanced over several nodes. Those nodes are strategically placed close geographically to the users which make downloads available with fever hops. This gives the users lower latency and increased delivery speed. The balancing process is choose by an algorithm on the CDN what node to use for the current client.

Some CDNs uses P2p technology, this differs from traditional CDN technology because peers are also uploading data. This is very powerful when popular content is about to be distributed, the more requests of same content the more available the content become.

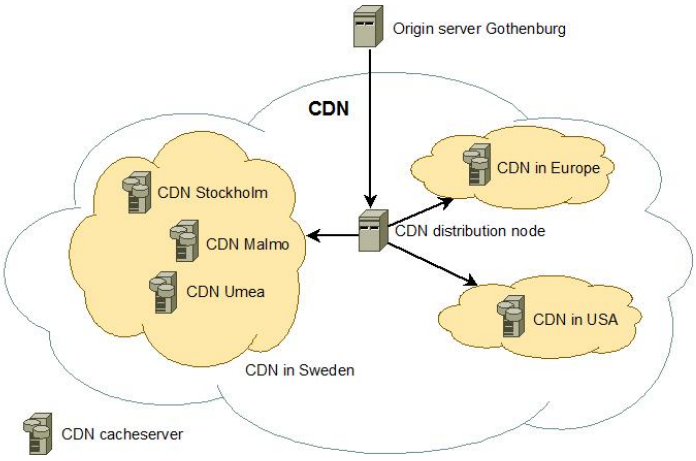


Figure 2.2: CDN architecture

2.4.3 Functionality

The purpose of CDNs is to distribute content to cache servers that are geographically close to the clients. By placing the content closer to the clients it is possible to increase the bandwidth to be even larger than the backbone network capacity.

There exists several techniques for CDNs to choose how to redirect end-users requests to the closest content server. One technique is called request routing. Routing tables can be either static or dynamic, thus in static manner the CDN company has chosen the best routing path and in dynamic manner the routing-tables are changing due to the statistic of the clients performance. Based on the routing-tables the content distributor can make a decision on where to redirect the client-request to the closest cache-server.

Client location	CDN cache server	Distance
USA	Gothenburg	140
Europe	Gothenburg	40
Sweden	Gothenburg	20
USA	Umea	180
Europe	Umea	80
Sweden	Umea	60
USA	Malmo	100
Europe	Malmo	10
Sweden	Malmo	5
USA	New York	20
Europe	New York	80
Sweden	New York	180

Table 2.1: Routing table

The structure of a CDNs includes

- Cache servers, these are placed in different locations and provide clients with copies of the requested content.
- A request routing procedure, that is responsible of redirect clients to the closest cache-server at lowest cost.
- The content distributor, distributes content from the origin-server to the cache servers.
- The origin server, contains the original of all copied data. The origin-server pushes new data to the content distributor, cache servers can also ask the origin-server for updates of copied content.

The main customer for a CDN company are content providers. The content providers pays CDN companies to distribute their content so it will become available to the end-users or customers. Thus, by using CDNs for content distribution you will gain great diversity which gives a reliable service to the users. There exists several commercial CDNs with support of both streaming and download.

2.4.4 Implementation

How is CDN functionality implemented into websites? Asume we have a website called www.music.com that give end-users the opportunity to purchase and download mp3 music files. A typical html-link at the website to a music file is www.music.com/music/mysong.mp3. The website is a customer of a CDN called www.cdns.com, thus all html-links in the website are now replaced with a reference to www.cdns.com/www.music.com. So the link to www.music.com/music/mysong.mp3 is now www.cdns.com/www.music.com/music/mysong.mp3. When a user clicks on the html-link the origin-server at www.music.com redirects the request to the CDN www.cdns.com/www.music.com, the CDN then redirects the request to a server that is most suitable for the request[12].

The picture below shows a typical procedure for a end-user request.

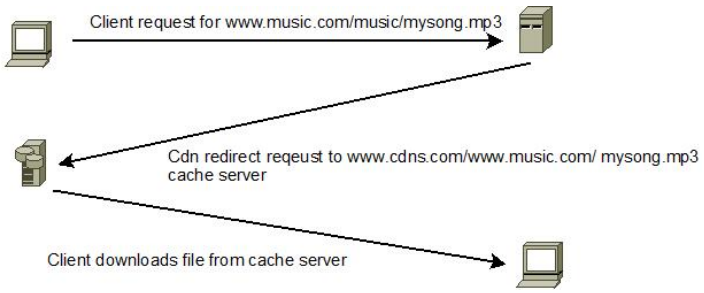


Figure 2.3: DNS routing

2.5 Background on main tools

2.5.1 HTML

Hypertext Markup Language (HTML) is a markup language for describing the structure and logic of a web document. It was originally created by Tim Burners-Lee, but was at this time not yet a specification. The first published specification for a HTML was published in 1993 by the Internet Engineering Taskforce (IETF). Since 1996 the HTML specifications have been maintained by the World Wide Web Consortium (W3C)[2]. The latest HTML specification is HTML 4.01 recommendation and was published by the W3C in 1999.

HTML provides elements for presenting text, headings, paragraphs, lists, tables, pictures etc into a web document. Almost every elements comes with two important properties, attributes and content. These properties have restrictions for corresponding element that must be followed to be considered as a valid HTML element. A HTML element is written on the form as a label, thus each element has a start label denoted as *<label>* and an end label denoted as *</label>* where the attributes is put into the start label and the content is put between the start and the end label. For example

<label attribute=value>Content</label>

The basic structure of a web document are based on so called root elements, these are containers for all other elements. Each web document starts and ends with the start label *<HTML>* and end label *</HTML>* these makes the main container. Then follows the header which has the start label *<HEAD>* and end label *</HEAD>*, here you put the meta keys, title, style sheets, script etc. The last root element is the body which have the start label *<BODY>* and end label *</BODY>* here are all elements put such a headings, tables, pictures etc. Except from the root elements each document should start with the element *<!DOCTYPE>*, this is to tell the browser what version of HTML that is used[3]. A small example is shown in picture below:

```
<!DOCTYPE HTML PUBLIC
    "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
<html>
    <head>
        <title>Document title</title>
    </head>
    <body>
        Text text text...
    </body>
</html>
```

Figure 2.4: A very simple web-page

2.5.2 CSS

Cascading Style Sheets (CSS) was first introduced by Håkon Wium Lie and Bert Bos 1994 in the browser Argo. CSS gives both authors and readers the possibility to create

their own style of how a document written in HTML should be presented. The main approach is to provide the separation of document content written in a markup language such as HTML from the document presentation written in CSS. CSS comes in various levels and profiles. Each level inherits the previous one but with new features and are denoted as CSS1, CSS2, CSS3.

Websites can contain hundreds or even thousands of HTML pages, imagine the work of changing just a simple feature such that change the colour on a header in all those pages. CSS can control the presentation of an arbitrary number of HTML-pages, thus a simple change in the CSS affects all HTML-pages including that style sheet. From an author perspective there are three ways how to provide CSS information into an HTML page[4].

1. External styles heets. A separate CSS-file which is referenced from the header in the html-document.

```
<head>
  <link href="templ.css" rel="stylesheet" type="text/css">
</head>
```

Figure 2.5: External css

2. Embedded style. Blocks of CSS information is provided inside the HTML document itself.

```
<head>
  <style type="text/css">
  <!--
    body {color: blue;}
    p {color: yellow;}
  -->
  </style>
</head>
```

Figure 2.6: Embedded css

3. Inline styles. By using the *style* attribute CSS information can be provideded to a single element in the HTML document.

2.5.3 PHP

PHP Hypertext Preprocessor (PHP) is an open source server-side script language for creating dynamic webpages. It was originally written by Rasmus Lerdorf in 1994 and was initially created to display his résume. In 1997 the Zeev Suraski and Andi Gutmans during their studies on the Israel Institute of Technology continued with Lerdorfs work, rewroting a new base called PHP 3 introducing basic object-oriented programming functionality. Later in 1999 Suraski and Gutmans completely rewrite the core of PHP which produced a new script engine called the Zend Engine. This new script engine lay ground

for further releases of PHP up until version PHP 5 which was powered by Zend Engine 2. The latest stable release for now is PHP 5.2.2 released in May 2 2007.

A PHP script starts with the label *<?PHP* and ends with the label ? >. PHP can either be mixed with HTML embedded into the HTML code or just written as plain PHP syntax in a separate php file. Following example shows how to embed a PHP script into HTML code. The script includes a PHP function *echo* which will when executed print the value of the variable *$title* as the title of the HTML page.

```
<html>
<head>
<title><?php echo title;?></title>
</head>
<body>
<p>My first PHP page</p>
</body>
</html>
```

When a browser requests a php page from a web server the PHP code is never revealed, instead the server takes the requested file, hand it to the PHP-engine which will produce HTML code and hand it back to the web server which in turn sends a HTML-file to the browser [5].

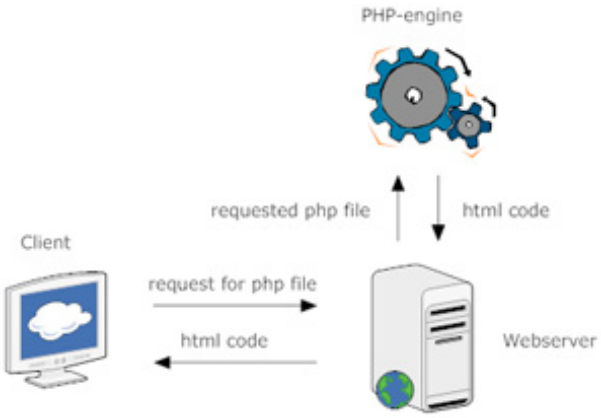


Figure 2.7: php engine

2.5.4 PEAR

PEAR is short for PHP Extension and Application Repository and was founded by Stig S. Bakken in 1999. The purpose of PEAR is to promote the re-use of code, hence it is providing a structured library of code for common functions which is free for anyone to use. PEAR also recommends a standard coding style for programs written in PHP. The library is built up on packages, each package includes functions for a specific task, for example the top tree library section database, includes packages with functions for

manipulating databases. Each package is a separate project with its own developers, version control and documentation. Packages does not need to depend on each others, thus dependencies from other packages have to be explicit expressed.[6]

```php
<?php
require_once 'MDB2.php';

$dsn = 'pgsql://someuser:apasswd@localhost/thedb';
$options = array(
    'debug' => 2,
    'result_buffering' => false,
);

$mdb2 =& MDB2::factory($dsn, $options);
if (PEAR::isError($mdb2)) {
    die($mdb2->getMessage());
}

$mdb2->disconnect();
?>
```

Figure 2.8: using PEAR package

2.5.5 Smarty

Smarty is a web template system written in PHP. The objective is to promote *compartmentalization*, that is to provide the separation of *business logic* as PHP from *presentation logic* as HTML. Using templates is a very attractive development-model since it allows web designers and web programmers to be separated from each other. Thus, programmers do not need to consider about changes made by the designers and can focus on the application logic, in a similar fashion the designers does not need to be involved with any application coding and can concentrate on the presentation coding.

Smarty do provide some logic to the designers but only concerning presentation logic. Hence, designers can alter assigned content to the templates given directives as variables, functions, logic or control flow statements. All directives is enclosed by square brackets, variables are denoted as supplementing the $ character as prefix. A smarty template is compiled into php code only once, that is avoiding run time parsing which gives a high performance[7].

The following example shows hows smarty can be used to separate the application logic from the presentation logic and also how to manipulate assigned content. In the php file index.php the variables *'title'* and *'message'* are assigned the static values *Index page* and *My home page*. As shown no concern is made about the presentation of the content in the php file.

2.5.6 MySQL

MySQL is a multi threaded, multi-user SQL database management system (DBMS) and was first released internally on May 23, 1995. MySQL is owned by the Swedish company MySQL AB, they also holds the copyright to most of the code base. MySQL is free to use and can be downloaded at their homepage www.mysql.com. MySQl is a relational

```php
<?php

include('Smarty.class.php');

/* create smarty object */
$smarty = new Smarty;

$array = array('hello', 'who' => 'you');

/* assigning content to the template */
$smarty->assign('title', 'Index page');
$smarty->assign('messages', $array);

/* display */

$smarty->display('test.tpl');

?>
```

Figure 2.9: index.php

```
<html>
    <head>
    <title>{$title}</title>
    </head>
    <body>
    {$messages[0]|capitalize} to {$messages.who}<br>
    </body>
</html>
```

Figure 2.10: test.tpl

database and have support for several storage-engines such as MyISAM which is the default storage engine and InnoDB which have support for transactions and foreign keys[8].

2.5.7 SQL Administration tool

PhpMyAdmin is a free web administration tool for administrate MySql databases over the Internet. It is written in PHP and is developed by *The phpMyAdmin Project* which took over the development from Tobias Ratschiller in 2001 due to the lack of time.

Chapter 3

System analysis and method

Underlying the literature research of Content Management System, Content Distribution and e-commerce following components and functionality has been identified as needed according to the requirement specification.

The functionality taken from content management system is mainly the work-flow model and the authoring tool. More specific a user should be able to create an account on the system and from this account manage the whole process from creation of content to publishing. Published content should be distributed to the public for evaluation and eventually purchase, this is maintained by using functionality from e-commerce system and content delivery networks.

3.1 Method

Building such a system from the beginning has to consider to be a quite large project. To get a good grasp of what to do you need to abstract the project into lower levels and identify the components at corresponding level. Therefore a top-down approach has been used structuring this project. To receive fast result during the implementation phase, the development phase has been characterised by an iterative process. Since the system is a web-application the three-tier model concept was selected as the system architecture, thus the system is divided into three tiers, presentation, application and data.

By using PHP in combination with the Smarty template engine the logic and presentation can be separated from each other just as the three-tier model requires. Due to this PHP was selected as the server side scripting language and Smarty as template engine.

To keep the Html-code clean from styling tags CSS was decided to manage the design and styling of the web pages. MySql was chosen to be the data tier since it has a well documented API towards PHP and also supports foreign keys and transactions .

All implementation and testing has been made on a local computer. That required a installation of an Apache server to run to PHP locally. To make the implementation convenient the development tool PHP Designer 2007 was selected, it has syntax highlighning for PHP, Html, CSS and it also includes a compiler for PHP code. To

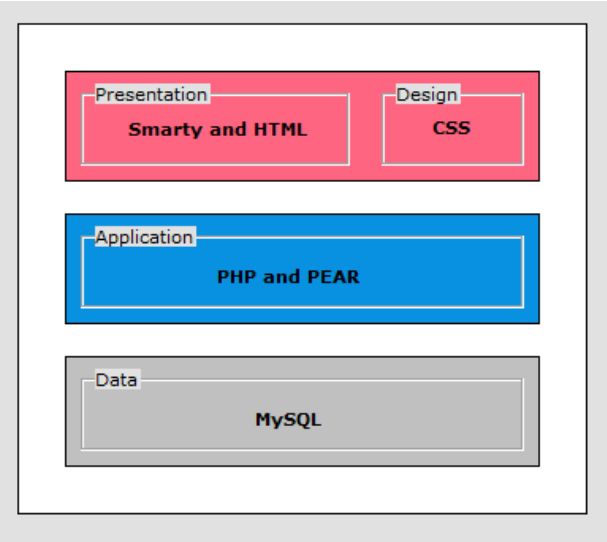


Figure 3.1: System architecture

administrate the MySql database the web based tool PHPAdmin was used.

Alternative techniques to the server side scripting language is ASP, ASP.net and JSP. The choice of using PHP was based on its strength by using Smarty that gives a great separation of logic and design, in addition it is well documented thus easy to learn how to use.

3.2 The database

All system data is stored into a database. The database structure consists of eight tables. The tables *admin*, *author*, *member* and *orders* are independent from each other, thus data can be added to a table without affecting any other table.

The tables *address*, *author_profile*, *news*, and *content* has one or more foreign key constraints meaning that the tables has primary keys related to another table.

Data can not be inserted into this tables if there are no such data in the refereed table. This is also very useful when tables are updated or deleted. For example if a user is deleted from the system all table data related to that user is also deleted. Each data table is briefly described in the list below and the relationship between the tables can be viewed in the picture.

- *admin*, includes login data of the systems administrators.
- *author*, includes login data and email of the systems authors.
- *member*, includes login data, email and user profile data such as account creation date, lastlogin date, balance and if the account is active.
- *adress*, includes the address data referring an author or a member of the system.

– *author_profile*, refers to an author account. Each author will have a profile page on the system and all that data is hold into this table.

– *news*, refers to authors table and holds information about the news published to the authors profile pages.

– *content*, refers to authors table. The content tables includes information of its owner, when it was published, the price, if its free or premium, how many downloads, rating etc.

– *orders*, holds information of a purchase of content. The date of the purchase, the content id, the owner of the content and the purchaser.

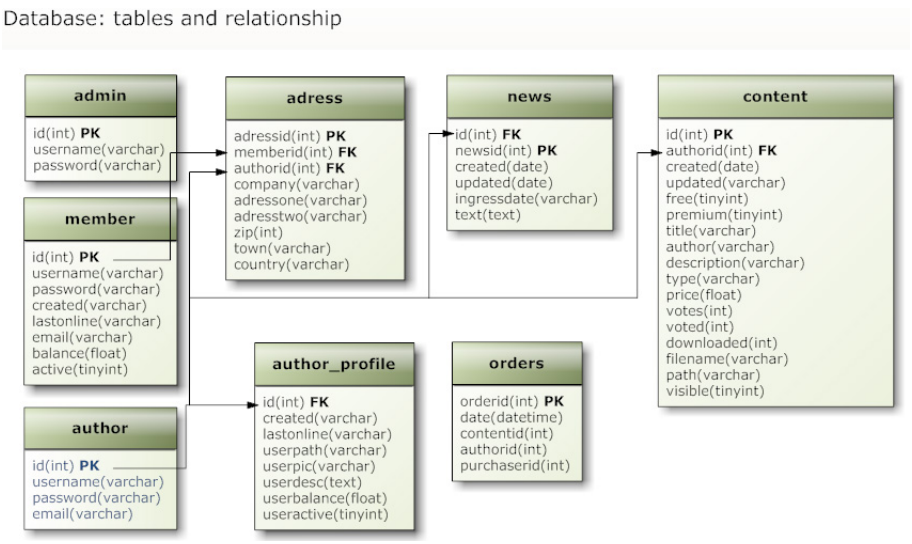


Figure 3.2: database schema

3.3 System structure

The application logic where broken down into four modules where each module corresponds to a component from the system specification. The presentation section includes one web-interface tool for the users to interact with the system accounts and one public web page where the content of the system will be published for the end-users. The application section handles all data processing in the system such as form data submitted by an user. All data is stored into the database in the data section.

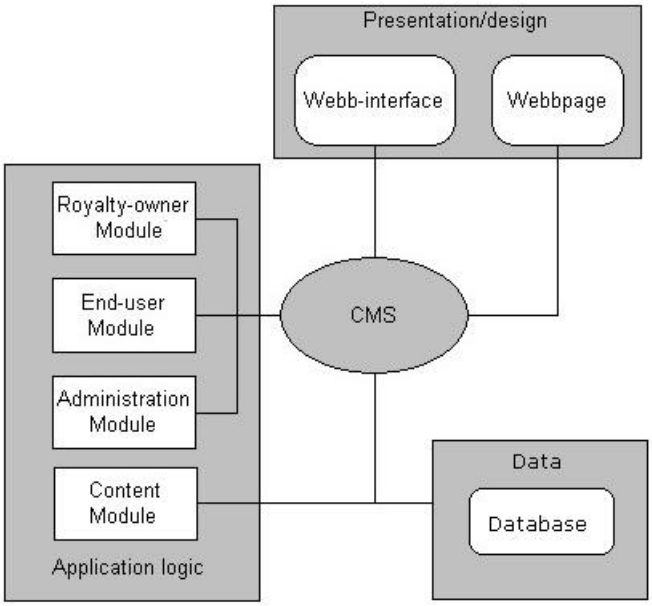


Figure 3.3: System structure

3.4 Application logic

There are three participants in the system, *royalty-owners*, *end-users* and the *administrator*. This section brings up the logic and processes for each participants and also explains the interaction between these. The forth module, *the content-module* is considered to be an entity rather than a participant.

3.4.1 Page templates

The get a nice look and feel of the system web pages they need to have a structured design. This is solved by using three smarty-templates header.tpl, menu.tpl and content.tpl. Each template is generating different html-code depending on what page that is requested. Down below the templates are generating the html-code for the index.php (System mainpage)

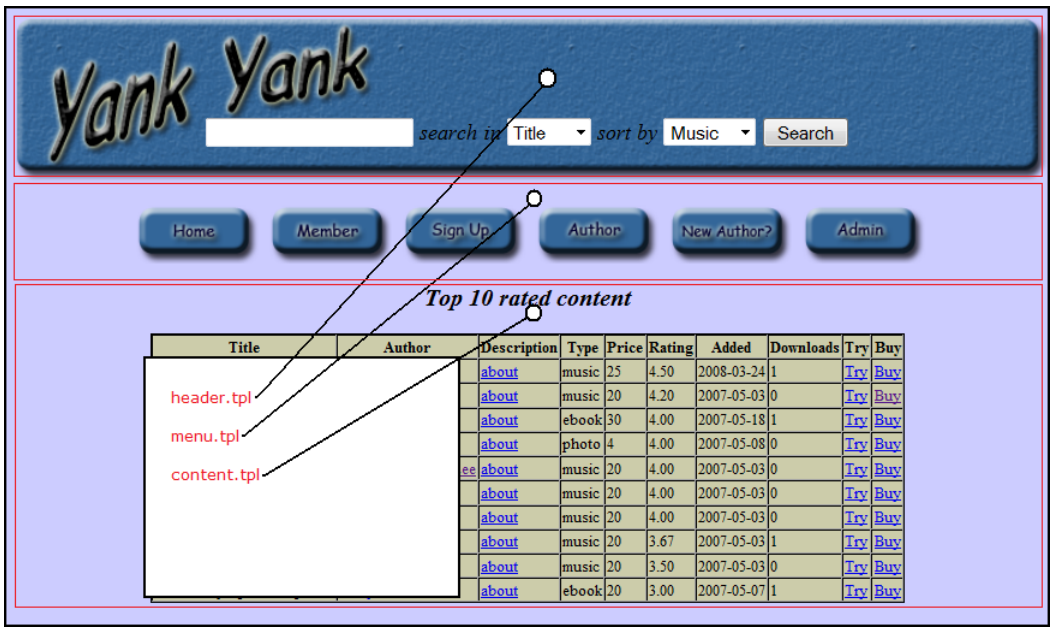


Figure 3.4: Generating index.php

This is very powerful since the same templates can be used for all pages of the system and independent of the functionality of these pages still remain the look and feel.

3.4.2 Login process

Users should be restricted from accessing specific areas in the system therefore a login-system was implemented. The login system allows end-users and royalty-owners to login to their accounts. To gain access to the system a user must submit a login form with the user name and password.

By using the PEAR-plugin HTML_QuickForm a login form can easily be created. First a new HTML_QuickForm object is created by the code statement.

```
$loginform = new HTML_QuickForm('login', 'post');
```

The constructor has two arguments *l*ogin which is the name of the form and *p*ost which sets that the data should be send as a post request. By using the following code statements elements are added to the form.
The form can have an arbitrary nr of element and different types, in this case three element is added, two text fields and one submit button. The text box element takes four

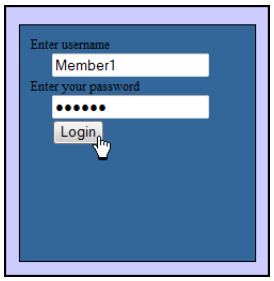
```
$loginform->addElement('text', 'username', 'Enter username', array('maxlength' => 20));
$loginform->addElement('password', 'password', 'Enter your password', array('maxlength' => 20));
$loginform->addElement('submit','sb','Login');
```

arguments, the type of text field, the logical name of the text field, the text displayed to the user and an array with the rules for the text field element. Here the rule is set to prevent the user from typing more than 20 characters. The button element also takes four arguments which are mostly the same but now we are dealing with a button

Validation rules and filters can also be added explicit. These code lines prevents the text fields text and password from being empty, a filter which removes white spaces is also applied.

```
$loginform->addRule('username', 'You need to type your username', 'required');
$loginform->applyFilter('username', 'trim');
$loginform->addRule('password', 'You need to type your password', 'required');
$loginform->applyFilter('password', 'trim');
```

The generated output from the previous code gives us the following html webform displayed in a web browser.



When a user hits the submit button the form validates according to the rules applied to the form. The validated data is processed by a function which compares the submitted data with the user data in the database. If the username and password is correct the user is directed to the users account page, else the user is redirected back to the login page.

The system needs to keep track of the users login state between the http requests. There are two ways to keep state of users, either by cookies or sessions. Since cookies are disabled in many browsers, sessions was used to keep state of users. To keep state of a user a session variable with the users username is added to the session.

Each time a request is made to the system, the session variable can be checked if it is set. If the session variable is set, the user is stilled logged in, else the user has logged out or the session has expired.

```
$_SESSION['username'] = $userName;
```

Figure 3.5: Session variable

3.4.3 Adding new users

The system provides ways to add both authors and members. There are two separate signup pages for each part of user. To sign up a user fill in a form with appropriate data like name, address and submit the form. If the form is valid the data is passed to function which sends it to the database where it will be stored. The user is now able to login to the system.

Figure 3.6: Webform signing up royalty-owner

3.5 Royalty-owner module

The royalty-owner module is the creative part in the system. This module should provide the royalty-owners with tools from which they can administrate and publish their content to make it available to the public.

The modules interface consists of several web pages, each page corresponds to a function in the system. The user can select the pages from a menu to interact with the system.

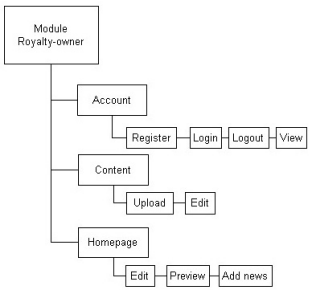


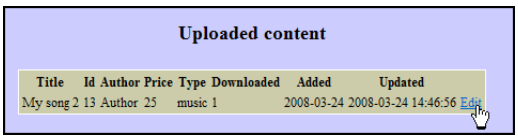
Figure 3.7: Flowchart royalty-owner interface

3.5.1 Creating Content

The purpose of the royalty-owners is to provide the system with content. It is rather important that this can be made easily. Adding content is handled by the php-page upload.php. The page consists of a number of web forms where the royalty-owner can name the title, owner, type, description and price of the content.

It also includes a file-upload control from where the user can select a file on it's harddrive to upload. When the user press the upload button the data of the content is stored in the database and the file is uploaded to the server.

The content is not immediately published to the public because it need to be reviewed and granted by the system administrator. The royalty-owner can list and view all uploaded content.



The list also gives the royalty-owner an opportunity to alter already uploaded content by pressing the edit link in the listed content column.

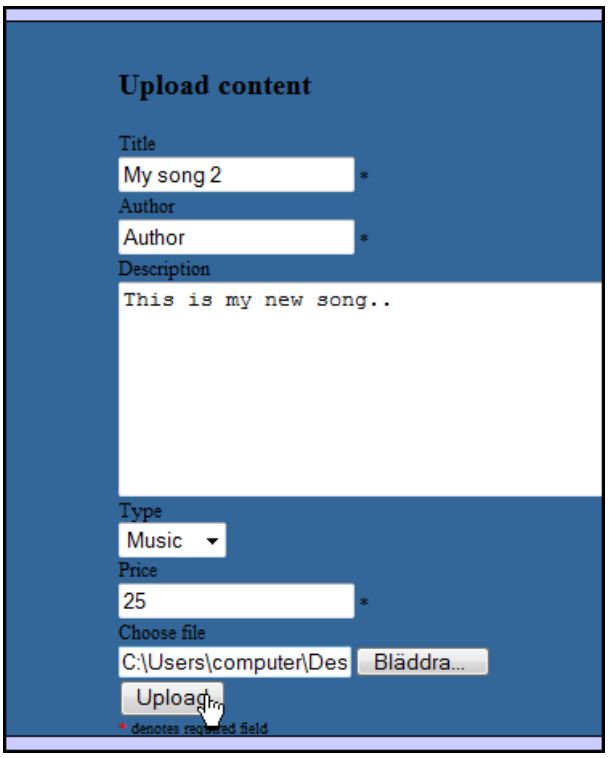


Figure 3.8: Form for uploading content

3.5.2 Managing profile page

Royalty-owners also need a way to promote themselves and their uploaded content. Therefore all royalty-owner accounts include a profile page. On this page it is possible to give a presentation of themselves by adding text, pictures and news to be viewed by visitors on the site. All uploaded content is automatically linked to the profile page for presentation.

The profile page consists of the php-page owner.php and the smarty-template owner.tpl. The layout of the page is managed by owner.tpl which includes the html-code of the page. The page-layout is buildt up on html div boxes. The first div box is the main container of the page, inside this box there are four other div boxes are included. These childboxes will be containers of the text, picture, news and content that are edited by the royalty-owner.

```
<div id="authorhomepage">
        <div class="authorPictureBox">
            {if $image}
                    <img src={$image} width="250" height="250">
            {else}
                    <img src="http://localhost/pictures/newpictures/no_pic.png" width="250" height="250">
            {/if}
        </div>
        <div class="authorDescriptionBox">
            {if $info}
                {$info}
            {else}
                <h3><em>No user profile</em></h3>
            {/if}
        </div>
        <div class="authorNewsBox">
            {if $news}
                <h3><em>News</em></h3>
                {foreach from=$news item=title}
                    <em>Date:</em> {$title[0]}<br />
                    {$title[1]}<br /><hr />
                {/foreach}
            {else}
                <h2>No news added</h2>
            {/if}
        </div>
        <div class="authorContentBox">
            {if $content}
            <h3><em>Content</em></h3>
            {foreach from=$content item=title}
            <em>Title:</em>{$title[6]}<br />
            <em>Type:</em>{$title[9]}<br />
            <em>Price:</em>{$title[10]}<br /><hr />
            {/foreach}
            {else}
                <h3><em>No content</em></h3>
            {/if}
        </div>
</div>
```

Figure 3.9: owner.tpl

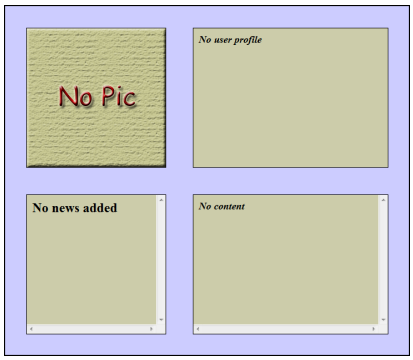


Figure 3.10: Output generated profilepage

3.6 End-users Module

The end-user module is the public part of the system, this is what visitors will see when they arrive to the system. The main page is index.php which will present the top ten most popular uploaded content.

To make it possible for visitors to search content by themselves a search-engine was implemented. To make the search results more specific and make it convenient to visitors the search can be made on title or author and the result can be sorted by type of content. The visitor types a word into the search field and then select from the two

Figure 3.11: Mainpage

dropdownlists how the result should be sorted. The text string from the text field and the values chosen from the dropdownlists is put into a sql-query, which is sent into the database. Some functions like rating and purchasing is only available for members of the

Figure 3.12: Mainpage search

system. To become a member a visitor needs to signup for the system, this is similar as signing up as a royalty-owner. By becoming a member of the system it gives the visitor an opportunity to rate and buy content, and overview over purchases.

3.7 Administration module

The administration module controls all users of the system. The purpose of the administrator is to make sure that no improper content is uploaded to the system. All content that is upload by the royalty-owners shows up in the administrators sandbox for review. If the administrator accepts the content the system will publish it for the public, else if the administrator rejects the content it will be removed from the system.

The administrator can also search for both content and users in the system and either edit those or remove them from the system.

Figure 3.13: Sandbox

3.8 Content Module

The content consists of three different types, music, photos and pdf-files. Visitors can test content by clicking on it. Clicking on a music content triggers a javascript to open the operatingsystems mediaplayer to play the content, clicking on a photo content also triggers a javascript which displays the photo to the visitor, pdf-files requires that a visitor has a pdf-reader plugin installed into the web browser. Each content also has a link to its royalty owner, by clicking this link the visitor is directed to the royalty-owners profile page. A content also has a description which can be reach by hovering with the mouse pointer over the content, this triggers a javascript to display a description of the content.

3.8.1 Purchase content

To purchase content a payment service provider (PSP) should be used. When an end-user is about to make a purchase they are automatically redirected to the PSP. The PSP provides all the security that is needed and handle all the transactions between the involved billing-agents.

3.8.2 Distribution of content

The system should not be responsible for the distribution of uploaded content. If the system grows big it would become a bottleneck when end-users are trying to access uploaded content which will make the system run slow. The system should instead use a content delivery network (CDN) to distribute uploaded content. Those provide servers with high capacity for a convenient downloading purpose from an end-user perspective. Thus the upload filelinks are located at the CDNs.

Chapter 4

Result and Discussion

4.1 Result

The work has resulted in a content management system the handles two groups of users and one administrator with basic functions. The first group, royalty-owners is the creator of content. These can sign-up to accounts which provides them with a tool from where they can publish and administrate their content. They do not need to have any knowledge of either html or webdesign to achieve their needs. The royalty-owners administration interface consists of a menu with following functions, implemented from the requirement specification

- upload content.
- edit content.
- list uploaded and accepted content.
- promote their self and their content.
- edit homepage.
- add news to homepage.
- view account info.

The other group is the visitors of the system. The functionality of the system is restricted for the visitors depending if the choose to become members or not. By becoming a member they get access to rate and buy content.

The system also provides a search engine from where the visitors can search for content. All content is able to be tested by the visitors.

4.2 Discussion

The work with this project has been very interesting and has given me a great experience of web development. It is obviously that the separation of logic and design is of great benefit, especially in bigger projects there different persons with different roles during the development are involved.

The work progress was a little slow in the beginning due to the indistinctness of the requirement specification. A lot of time was also spent to get familiar with the chosen tools for the implementation. The project felt quite big in the beginning and it was hard to focus on where to start. Therefore the choice of the top-down method by dividing the problem into smaller parts was very successful.

4.3 Restrictions

Since Cellips accounts for IP-telephony already had mechanisms for handling payments the purpose was to link those accounts to this system accounts. After discussion this was rejected due to the difficulty of merging those accounts. It was also wanted that end-users should become reviewers but that was left for future plans.

4.4 Limitations

The system does not provide ways for end-users to make purchases. This will not be treated in this project since its not within the limits of this project. Providing purchase mechanisms demands highly security on both the system and on the server where the system is running. There are also restrictions and laws of how to handle credit-card numbers and sensitive personal information.

4.5 Future work

Since the design and logic is separated future work on the design of the system is easily changed. Purchasing content should be implemented by using a Payment Service Provider.

In principle there is no restriction how much the system can be extended but it should be done with care. Adding to many functions can make the system less user friendly. The system should provide a mechanism to make test versions of uploaded content for example automatically create 30 seconds of an uploaded mp3 song or making a low resolution picture. All uploaded content should also be placed on servers(CDNs) with supreme bandwidth to relieve stress on the system server and make download processes quick.

4.6 References

[1] Robertsson, James *So, what is a content management system?*
http://www.steptwo.com.au/papers/kmc_what/
2003-06-03

[2] WC3 - HTML 4.01 Specification
http://www.w3.org/TR/html401/
1999-12-24

[3] HTML - Wikipedia
http://en.wikipedia.org/wiki/Html
2007-05-05

[4] Cascading Style Sheets - Wikipedia
http://en.wikipedia.org/wiki/Cascading_Style_Sheets
2007-05-05

[5] PHP
http://php.net/
2007-05-02

[6] PEAR - PHP Extension and Application Repository
http://pear.php.net/manual/en/introduction.php/
2007-05-05

[7] Smarty - Template engine
http://www.smarty.net/whyuse.php
2007-05-05

[8] MySql
http://en.wikipedia.org/wiki/MySQL
2007-08-01

[9] The phpMyAdmin Project
http://www.phpmyadmin.net/home_page/
2007-08-01

[10] E-commerce system - Wikipedia
http://sv.wikipedia.org/wiki/E-handel
2007-05-05

[11] Payment mechanisms - Swedbank
http://www.swedbank.se
2008-12-16

[12] Content Delivery Network - Wikipedia
http://en.wikipedia.org/wiki/Content_Delivery_Network
2008-11-05

Appendix A

Source code

```
<?php

/*
 *  sourcecode for index.php
 */

session_start();

require_once('Smarty.class.php');
require_once "functions.php";
    require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
    require_once "HTML/QuickForm.php";

$smarty = new Smarty;
$searchform = new HTML_QuickForm('search', 'get');

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

/* Add elements to form */

   $searchform->addElement('text', 'searchtext', '', array('maxlength' => 20));
$searchform->addElement('submit','sb','Search');
$options = array('title'=>'Title','author'=>'Author');
$typeoptions = array('music'=>'Music','photo'=>'Photo', 'ebook'=>'E-book');
   $attributes = array('size'=>'1');
   $searchform->addElement('select', 'searchselect', 'search in', $options, $attributes);
$searchform->addElement('select', 'searchselecttype', 'sort by', $typeoptions, $attributes);

   /* Add rules to fields */

$searchform->addRule('searchtext', '', 'required');
```

```
$searchform->applyFilter('searchtext', 'trim');


/* Start a new shopping cart */

if(!isset($_SESSION['cart']))
{
$_SESSION['items'] = 0;
$_SESSION['total'] = 0;
}

/* check the searchform */

if ($searchform->validate())
{

$value = $_GET['searchselect'];
$type = $_GET['searchselecttype'];
$searchString = $_GET['searchtext'];
$query = "select * from content where $value like \"%$searchString%\" and type='$type' and vi

/*if we get a match display hits */

$content = searchContent($query);

if($content)
{

/* display the page and searchresult */

$rendererSearch = new HTML_QuickForm_Renderer_ArraySmarty($smarty);

$searchform->accept($rendererSearch);

$smarty->assign('searchform', $rendererSearch->toArray());
$title = "Yank Yank";
$smarty->assign('title', $title);
$home = "home";
$smarty->assign('home', $home);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
    $smarty->assign('content', $content);
    $smarty->display('content2.tpl');
    $smarty->display('footer2.tpl');
}
else
{
/* display the page and searchresult */
```

```
$rendererSearch = new HTML_QuickForm_Renderer_ArraySmarty($smarty);

$searchform->accept($rendererSearch);

$smarty->assign('searchform', $rendererSearch->toArray());
$title = "Yank Yank";
$smarty->assign('title', $title);
$home = "home";
$smarty->assign('home', $home);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
$message = "Search resulted in no hits";
$smarty->assign('message', $message);
$chartInfo = getTopTen();
$smarty->assign('toptenfree', $chartInfo);
    $smarty->display('content2.tpl');
    $smarty->display('footer2.tpl');
}

}
else
{
/* Display index page */

$rendererSearch = new HTML_QuickForm_Renderer_ArraySmarty($smarty);

$searchform->accept($rendererSearch);

/* display the page content */

$smarty->assign('searchform',$rendererSearch->toArray());
$title = "Yank Yank";
$smarty->assign('title', $title);
$home = "home";
$smarty->assign('home', $home);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
$chartInfo = getTopTen();
$smarty->assign('toptenfree', $chartInfo);
    $smarty->display('content2.tpl');
$smarty->display('footer2.tpl');
   }
?>



<?php

/*
 *  sourcecode for accept.php
```

```
 */

    session_start();

require('Smarty.class.php');
require_once "functions.php";

    $smarty = new Smarty;
$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

//if(check_valid_user()) {
if(admin_logged_in()) {

  $user = $_SESSION['username'];


  if(isset($_REQUEST['vis']) && isset($_REQUEST['cid']) && isset($_REQUEST['oid'])) {

if($_REQUEST['vis']==1) {

$ownerid = $_REQUEST['oid'];
$contentid = $_REQUEST['cid'];

if(setVisible($ownerid, $contentid)) {

  $smarty->display('header2.tpl');
$smarty->display('adminmenu.tpl');
    $message = "Content accepted";
$smarty->assign('message',$message);
$content = getSandbox();
$smarty->assign('content',$content);
    $smarty->display('contentAdmin.tpl');
    $smarty->display('footer2.tpl');
exit;
}
}
}
$smarty->display('header2.tpl');
$smarty->display('adminmenu.tpl');
$message = "Failed to accept content, pls try again later";
$smarty->assign('message',$message);
$smarty->display('contentAdmin.tpl');
$smarty->display('footer2.tpl');


  }
```

```
else
echo 'Not authorized';
?>
```

```
<?php

/*
 *  sourcecode for addNews.php
 */

session_start();

require('Smarty.class.php');
require_once "functions.php";
    require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
    require_once "HTML/QuickForm.php";

    $smarty = new Smarty;
$newsform = new HTML_QuickForm('addnews', 'post');

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

/* Add elements to field */

   $options = array(
          'language'  => 'swe',
          'format'    => 'yMd',
          'minYear'   => date("Y"),
          'maxYear'   => 2010
    );

   $newsform->addElement('date', 'publishdate', 'Choose date', $options);
   $newsform->addElement('textarea', 'newstext', 'News', array('maxlength' => 200, 'rows' =>
   $newsform->addElement('submit', 'sb', 'Add news');

   /* add rules to fields */

   $newsform->addRule('textarea', 'You need to type something', 'required');

   //if(check_valid_user()) {
   if(author_logged_in()) {
/* check that fields are valid */

   if ($newsform->validate()) {

$id = $_SESSION['id'];
$date = $_POST['publishdate'];
```

```
$date = $date['y']."-".$date['M']."-".$date['d'];
$text = $_POST['newstext'];

if(addNews($id, $date, $text)) {

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $newsform->accept($renderer);

/* display the page content */

$title = "Yank Yank - author add news";
$smarty->assign('addnews', $title);
$smarty->display('header2.tpl');
    $smarty->display('authormenu.tpl');
    $message = "News added";
$smarty->assign('message', $message);
$smarty->display('user.tpl');
$smarty->display('footer2.tpl');
exit;
}
else {
$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $newsform->accept($renderer);

/* display the page content */

$title = "Yank Yank - author add news";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('authormenu.tpl');
    $error = "Failed to add news, please try again later";
$smarty->assign('error', $error);
$smarty->display('user.tpl');
$smarty->display('footer2.tpl');
exit;
}

   }
   else {

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $newsform->accept($renderer);

/* display the page content */

$title = "Yank Yank - author add news";
$smarty->assign('add-news', $title);
$smarty->display('header2.tpl');
$smarty->display('authormenu.tpl');
```

```
$smarty->assign('newsform',$renderer->toArray());
    $smarty->display('user.tpl');
$smarty->display('footer2.tpl');

}
}
else {
echo "Not authorized";
exit;
    }
?>

<?php

/*
 *  sourcecode for addtocart.php
 */

session_start();

if(isset($_GET['add']))
{
/* first item */

if($_SESSION['items'] == 0)
{
$_SESSION['items']++;
$_SESSION['cart'] [$_SESSION['items']] = array('ownerid' => $_GET['oi'], 'contentid' => $_GET
}

/* there are allready items in cart */

else
{
$_SESSION['items']++;
$_SESSION['cart'] [$_SESSION['items']] = array('ownerid' => $_GET['oi'], 'contentid' => $_GET
}
$_SESSION['total'] += $_GET['price'];
}
header("Location: http://localhost/index.php");
?>

<?php

/*
 *  sourcecode for admin.php
 */

/* Admin home page */
```

```
session_start();

require('Smarty.class.php');
require_once "functions.php";

$smarty = new Smarty;

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

//if(check_valid_user()) {
if(admin_logged_in()) {

  $user = $_SESSION['username'];
  $id = $_SESSION['id'];

$smarty->display('header2.tpl');
    $smarty->display('adminmenu.tpl');
    $message = "Logged in as ".$user;
$smarty->assign('message',$message);
    $smarty->display('contentAdmin.tpl');
    $smarty->display('footer2.tpl');
}
else
echo header("Location: http://localhost/adminlogin.php");
?>

<?php

/*
 *  sourcecode for adminlogin.php
 */

require_once "HTML/QuickForm.php";
   require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
   require_once "functions.php";
require('Smarty.class.php');

$smarty = new Smarty;
$adminloginform = new HTML_QuickForm('login', 'post');

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

/* Add elements to form */
```

```
   $adminloginform->addElement('text', 'usradmin', 'Enter admin username', array('maxlength' =
   $adminloginform->addElement('password', 'pswdadmin', 'Enter password', array('maxlength' =
   $adminloginform->addElement('submit','sb','Login');

   /* Add rules to form */

   $adminloginform->addRule('usradmin', 'You need to type your username', 'required');
   $adminloginform->applyFilter('usradmin', 'trim');
   $adminloginform->addRule('pswdadmin', 'You need to type your password', 'required');
   $adminloginform->applyFilter('pswdadmin', 'trim');

   /* validate the loginform */

if ($adminloginform->validate()) {

/* now check with database and if ok redirect to userpage */

$usrName = $_POST['usradmin'];
$pswd = $_POST['pswdadmin'];

session_start();

if(loginAdmin($usrName, $pswd)) {

$_SESSION['username'] = $usrName;

/* redirect to userpage */

header("Location: http://localhost/admin.php");
}
else {

/* login failed */

$rendererLogin = new HTML_QuickForm_Renderer_ArraySmarty($smarty);

     $adminloginform->accept($rendererLogin);

$title = "Yank Yank - admin login";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
    $smarty->assign('adminloginform', $rendererLogin->toArray());
    $error = "Login failed";
$smarty->assign('error', $error);
$smarty->display('content2.tpl');
    $smarty->display('footer2.tpl');
exit;
}
```

```
}
else {

/* Show the login form */

$rendererLogin = new HTML_QuickForm_Renderer_ArraySmarty($smarty);

$adminloginform->accept($rendererLogin);
  $title = "Yank Yank - admin login";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
    $smarty->assign('adminloginform', $rendererLogin->toArray());
$smarty->display('content2.tpl');
    $smarty->display('footer2.tpl');
}
?>
```

```
<?php

/*
 *  sourcecode for delete.php
 */

    session_start();

require('Smarty.class.php');
require_once "functions.php";
require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
    require_once "HTML/QuickForm.php";

    $smarty = new Smarty;
$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

$deleteform = new HTML_QuickForm('delete', 'post');

    /* set default values in fields */

$defaults = array('title' => $_REQUEST['title'] , 'contentid' => $_REQUEST['contentid']);
$deleteform->setDefaults($defaults);

    $deleteform->addElement('hidden', 'title');
   $deleteform->addElement('hidden','contentid');
   $deleteform->addElement('submit','delete','YES');

//if(check_valid_user()) {
if(admin_logged_in()) {
```

```
/* present form fields */

  $user = $_SESSION['username'];


  if ($deleteform->validate()) {

    /* set variables */

     $values['id'] = $_SESSION['id'];
     $values['title'] = $_POST['title'];
     $values['contentid'] = $_POST['contentid'];

     if(deleteContent($values))
     {
     /* display the page content */

$smarty->display('header.tpl');
$smarty->display('usermenu.tpl');
$message = "Content deleted";
$smarty->assign('message', $message);
$smarty->display('user.tpl');
$smarty->display('footer.tpl');
exit;
}
else {

/* display the page content */

$smarty->display('header.tpl');
    $smarty->display('usermenu.tpl');
    $err = "Delete failed";
$smarty->assign('err', $err);
    $smarty->display('user.tpl');
$smarty->display('footer.tpl');
}
}
else {

/* display the page content */
$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
    $deleteform->accept($renderer);
    $message = "Are you sure you want to delete content?";
$smarty->display('header.tpl');
    $smarty->display('usermenu.tpl');
    $smarty->assign('deleteform',$renderer->toArray());
    $smarty->assign('message',$message);
    $smarty->display('contentuser.tpl');
```

```
$smarty->display('footer.tpl');

}

}
else
echo 'Not authorized';
exit;
?>

<?php

/*
 *  sourcecode for download.php
 */

session_start();

echo '<a href="http://localhost/user/music.mp3">Click on link to download content</a>';


?>

<?php

/*
 *  sourcecode for edit.php
 */

    session_start();

require('Smarty.class.php');
require_once "functions.php";
require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
    require_once "HTML/QuickForm.php";

    $smarty = new Smarty;
$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

$editform = new HTML_QuickForm('edit', 'post');

    /* set default values in fields */

    $defaults = array('title' => $_REQUEST['title'] ,
       'contentid' => $_REQUEST['contentid'],
  'author' => $_REQUEST['author'],
  'description' => $_REQUEST['description'],
```

```
  'type' => $_REQUEST['type'],
  'price' => $_REQUEST['price']);

    $editform->setDefaults($defaults);

    $editform->addElement('text', 'title', 'Title', array('maxlength' => 40));
   $editform->addElement('text', 'author', 'Author', array('maxlength' => 40));
   $editform->addElement('textarea', 'description', 'Description', array('maxlength' => 200))
   $editform->addElement('hidden','contentid');
   $options = array('music'=>'Music','ebook'=>'Ebook','photo'=>'Photo');
   $attributes = array('size'=>'1');
   $editform->addElement('select', 'type', 'Type',$options,$attributes);
   $editform->addElement('text', 'price', 'Price', array('maxlength' => 10));
   $editform->addElement('submit','sb','Submit changes');

   /* add rules to fields */

   $editform->addRule('title', 'You need to type title', 'required');
   $editform->applyFilter('title', 'trim');
   $editform->addRule('author', 'You need to type author', 'required');
   $editform->applyFilter('author', 'trim');
   $editform->addRule('price', 'You need to add a price', 'required');
   $editform->applyFilter('price', 'trim');
   $editform->addRule('price', 'Can only contain digits', 'numeric');

//if(check_valid_user()) {
if(author_logged_in()) {
/* present form fields */

  $user = $_SESSION['username'];


  if ($editform->validate()) {

    /* now put info into database */

     $values['id'] = $_SESSION['id'];
     $values['title'] = $_POST['title'];
     $values['contentid'] = $_POST['contentid'];
     $values['author'] = $_POST['author'];
     $values['description'] = $_POST['description'];
     $values['type'] = $_POST['type'];
     $values['price'] = $_POST['price'];

     if(updateContent($values))
     {
     /* display the page content */

$smarty->display('header2.tpl');
```

```
$smarty->display('authormenu.tpl');
$message = "Changes completed";
$smarty->assign('message', $message);
$smarty->display('user.tpl');
$smarty->display('footer2.tpl');
exit;
}
else {

/* display the page content */

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $editform->accept($renderer);

$smarty->display('header2.tpl');
$smarty->display('authormenu.tpl');
    $smarty->assign('editform',$renderer->toArray());
    $err = "Changes failed";
$smarty->assign('err', $err);
    $smarty->display('user.tpl');
$smarty->display('footer2.tpl');

}
}
else {

/* display the page content */

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
    $editform->accept($renderer);

$smarty->display('header2.tpl');
    $smarty->display('authormenu.tpl');
    $smarty->assign('editform',$renderer->toArray());
    $smarty->display('user.tpl');
$smarty->display('footer2.tpl');

}

}
else
echo 'Not authorized';
?>

<?php

/*
 *  sourcecode for functions.php
 */
```

```
require_once "DB.php";

/* */

function loginOwner($username, $password){

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */
$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

/* query for user */

$password = crypt($password,"project");

$res =& $db->query('select * from user where username="'.$username.'" and password="'.$passwo

if (PEAR::isError($res)) {
return false;
     }

     /* get user id for actuall user */

     $res->fetchInto($row);

     $_SESSION['id'] = $row[0];
     $_SESSION['author'] = $row[0];

     if($res->numRows()==1) {
     return true;
     }
     return false;
}

function loginAdmin($username, $password) {
```

```
$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */
$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

/* query for user */
$res =& $db->query('select id from admin where name="'.$username.'" and password="'.$password

if (PEAR::isError($res)) {
return false;
     }

     /* get user id for actuall user */

     $res->fetchInto($row);

     $_SESSION['id'] = $row[0];
     $_SESSION['admin'] = $row[0];

     if($res->numRows()==1)
     return true;
     return false;
}

function loginMember($username, $password) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);
```

```
$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */
$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

/* query for user */

$password = crypt($password,"project");

$res =& $db->query('select id from members where membername="'.$username.'" and password="'.$

if (PEAR::isError($res)) {
return false;
     }

     /* get user id for actuall user */

     $res->fetchInto($row);

     $_SESSION['id'] = $row[0];
     $_SESSION['customer'] = $row[0];

     if($res->numRows()==1)
     return true;
     return false;
}

function logout($usertype)
{
$old_user = $_SESSION['username'];
$id = $_SESSION['id'];

if (isset($_COOKIE[session_name()])) {
    setcookie(session_name(), '', time()-42000, '/');
}

unset($_SESSION);

$result_dest = session_destroy();

if(!empty($old_user))
{
if($result_dest)
```

```
{
/* logout succeded */

/* set last login */

         $dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}
         if($usertype=="user")
         {
     $date = date("Y-m-d H:i:s");
     $query = "update userprofile set lastonline='$date' where userid='$id'";
     $res =& $db->query($query);

if (PEAR::isError($res)) {
echo $res->getMessage();
echo '<br />';
exit;
return false;
     }
return true;
}
else if($usertype=="member")
         {
     $date = date("Y-m-d H:i:s");
     $query = "update members set lastonline='$date' where id='$id'";
     $res =& $db->query($query);

if (PEAR::isError($res)) {
echo 'here';
echo $res->getMessage();
echo '<br />';
exit;
```

```
return false;
     }
return true;
}
return true;
}
else
{
/* logout failed */
return false;
}
}
else
{
/* user were not logged in but reached this page somehow */
return true;
}
}

function check_username($username) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$res =& $db->query("select * from user where username='$username'");

if (PEAR::isError($res)) {
return false;
     }
     /* check if value is uniqe */

     if($res->numRows()>0)
```

```
     return false;
return true;
}

function isUnique($table, $row, $value) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "select * from $table where $row='$value'";

$res =& $db->query($query);


if (PEAR::isError($res)) {
return false;
     }
     /* check if value is uniqe */

     if($res->numRows()>0)
     return false;
return true;
}

function register($values) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
```

```
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$username = $values['username'];
$email = $values['email'];
$password = crypt($values['password'],"project");

/* put data into user table */
//echo "insert into user values(null, '$username', '$email', '$password')";
//exit;
$res =& $db->query("insert into user values(null, '$username', '$email', '$password')");

if (PEAR::isError($res)) {
     return false;
}

     /* now create user dir */

     $res =& $db->query("select id from user where username='$username' and password='$passwo

if (PEAR::isError($res))
return false;

     /* get user id for actuall user */

     $res->fetchInto($row);

     if($res->numRows()==0)
     return false;

     $path = 'c:\\\root\\\public\\\users\\\\'.$row[0];

     if(!mkdir($path, 0600))
     return false;

/* put userpath and rest of data into table userprofile */

     $date = date("Y-m-d H:i:s");
```

```
     $query = "insert into userprofile values ('$row[0]', '$date', '$date', '$path', null, nu

$res =& $db->query($query);
if (PEAR::isError($res)) {
echo $res->getMessage();
echo '<br />';
exit;
return false;
     }

/* insert values into table adress */

$company = $values['company'];
$adressone = $values['adressone'];
$adresstwo = $values['adresstwo'];
$zip = $values['zip'];
$town = $values['town'];
$country = $values['country'];

$query = "insert into adress values ('$row[0]', null, null, '$company', '$adressone', '$adres

$res =& $db->query($query);
if (PEAR::isError($res)) {
echo $res->getMessage();
echo '<br />';
exit;
return false;
     }

return true;
}

function registerMember($values) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */
```

```
$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$username = $values['username'];
$email = $values['email'];
$password = crypt($values['password'],"project");

/* put data into user table */

$date = date("Y-m-d H:i:s");

$res =& $db->query("insert into members values(null, '$date', null, '$username', '$password',

if (PEAR::isError($res)) {
echo 'here'.'<br />';
echo $res->getMessage();
exit;
     return false;
}

     /* set previliges on the user */


     $res =& $db->query("select id from members where membername='$username' and password='$p

if (PEAR::isError($res))
return false;

     /* get user id for actuall user */

     $res->fetchInto($row);

     if($res->numRows()==0)
     return false;

/* insert values into table adress */

$adressone = $values['adressone'];
$zip = $values['zip'];
$town = $values['town'];
$country = $values['country'];

$query = "insert into adress values (null, '$row[0]', null, null, '$adressone', null, '$zip',

$res =& $db->query($query);
if (PEAR::isError($res)) {
```

```
return false;
     }

return true;
}

function uploadfile($path) {
global $file;
if ($file->isUploadedFile()) {
         $file->moveUploadedFile($path);
          return true;
     }
return false;
}


function putToDatabase($values) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

$table = $values['table'];
$owner = $values['id'];
$description = $values['description'];
$filename = $values['filename'];
//$path = $values['path'];

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "select userpath from userprofile where userid='$owner'";
$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
```

```
    }

$res->fetchInto($row);
$path = $row[0];

/* if uploading content */

if($table=='content') {

$title = $values['title'];
$author = $values['author'];
$type = $values['type'];
$price = $values['price'];

$query = "select * from $table where title = '$title' and ownerid = '$owner'";

$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
     }
     /* check that file doesnt allready exists */

     if($res->numRows()>0)
return false;
else {
     /* put into database */

     $date = date("Y-m-d H:i:s");

$query = "insert into content values('$owner', null, (select current_timestamp), '$date', 0, (
$res =& $db->query($query);

if (PEAR::isError($res)) {
return false;
     }
     return true;
}
}

/* if saving userinfo for homepage */

else if($table=='userprofile') {

$query = "update userprofile set userpic='$filename', userdesc='$description' where userid='$
$res =& $db->query($query);
if (PEAR::isError($res)) {
echo $query;
echo $res->getMessage();
```

```
exit;
return false;
     }
     return true;

}
else
return false;
}

function getContent($id) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "select * from content where ownerid = '$id' and visible=1";
$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
     }
     $rows = $res->numRows();

     if($rows==0)
     return false;

     $i = 0;
     while($res->fetchInto($content[$i]) and ($i < $rows-1)) {
$i++;
}
     if(PEAR::isError($res)) {
return false;
```

```
     }
     else
return $content;
}

function getSingleContent($ownerid, $contentid) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "select * from content where ownerid = '$ownerid' and id='$contentid'and visible=1";
$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
     }
     $rows = $res->numRows();

     if($rows==0)
     return false;

        $res->fetchInto($content);

     if(PEAR::isError($res)) {
return false;
     }
     else
return $content;
}

function getCartContent($oid, $cid) {
```

```
$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "select * from content where ownerid = '$oid' and id='$cid' and visible=1";
$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
     }
     $rows = $res->numRows();

     if($rows==0)
     return false;

     $i = 0;
     while($res->fetchInto($content[$i]) and ($i < $rows-1)) {
$i++;
}
     if(PEAR::isError($res)) {
return false;
     }
     else
return $content;
}

function addNews($id, $date, $news) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
```

```
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}
$query = "insert into news values($id,null,(select current_timestamp),(select current_timesta

$res =& $db->query($query);

if(PEAR::isError($res)) {
echo $res->getMessage();
exit;
return false;
     }
     else
return true;
}

function getNews($id) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}
```

```
$query = "select ingressdate, text from news where id = '$id' order by ingressdate";
$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
     }
     $rows = $res->numRows();

     if($rows==0)
     return false;

     $i = 0;
     while($res->fetchInto($info[$i]) and ($i < $rows-1)) {
$i++;
}
     if(PEAR::isError($res)) {
return false;
     }
     else
return $info;
}

function searchContent($query) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

/* query database */

$res =& $db->query($query);

if(PEAR::isError($res)) {
```

```
return false;
     }
     $rows = $res->numRows();

     if($rows==0)
     return false;

     $i = 0;
     while($res->fetchInto($content[$i]) && ($i < $rows-1)) {
     $row = $content[$i];
     if ($row[11]==null); /* check if content not yet have been rated */
     $row[11]=0;
     if ($row[12]==null);
     $row[12]=0;
    $i++;
}
     if(PEAR::isError($res)) {
return false;
     }
     else
return $content;

}

function updateContent($values) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

$owner = $values['id'];
$title = $values['title'];
$contentid = $values['contentid'];
$author = $values['author'];
$description = $values['description'];
$type = $values['type'];
$price = $values['price'];

/* connect to database */
```

```
$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}
$date = date("Y-m-d H:i:s");
$query = "update content set updated='$date', title='$title', author='$author', description='

$res =& $db->query($query);
if (PEAR::isError($res)) {
echo $res->getMessage();
echo '<br />';
echo $query;
exit;
return false;
     }
     return true;


}

function deleteContent($values) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

$owner = $values['id'];
$title = $values['title'];
$contentid = $values['contentid'];

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "delete from content where title='$title' and ownerid='$owner' and id='$contentid' l

$res =& $db->query($query);
```

```
     if (PEAR::isError($res)) {
echo $res->getMessage();
exit;
return false;
     }

     $rows =& $db->affectedRows();

if($rows==0)
     return false;

     return true;


}

function rateContent($values) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

$owner = $values['ownerid'];
$contentid = $values['contentid'];
$votes = $values['voted'];

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "update content set votes=votes + '$votes', voted= voted + 1 where ownerid='$owner' a

$res =& $db->query($query);
if (PEAR::isError($res)) {
return false;
     }
```

```
     $rows =& $db->affectedRows();

     if($rows==0)
     return false;

return true;


}



function getUserInfo($id) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

//$query = "select userpath, userpic, userdesc, userbalance from userprofile where userid = '
$query = "select * from userprofile where userid = '$id'";

$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
     }
     $rows = $res->numRows();

     /* check if user exists */

     if($rows == 0)
     return false;
```

```
     $i = 0;
     while($res->fetchInto($info[$i]) and ($i < $rows-1)) {
$i++;
}
     if(PEAR::isError($res)) {
return false;
     }
     else {
return $info;
}
}

function getMemberInfo($id) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "select * from members where id = '$id'";

$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
     }
     $rows = $res->numRows();
     $i = 0;
     while($res->fetchInto($info[$i]) and ($i < $rows-1)) {
$i++;
}
     if(PEAR::isError($res)) {
return false;
     }
```

```
     else {
return $info;
}
}

function getUserAdress($id) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

//$query = "select userpath, userpic, userdesc, userbalance from userprofile where userid = '
$query = "select * from adress where userid = '$id'";

$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
     }
     $rows = $res->numRows();
     $i = 0;
     while($res->fetchInto($adress[$i]) and ($i < $rows-1)) {
$i++;
}
     if(PEAR::isError($res)) {
return false;
     }
     else {
return $adress;
}
}

function getMemberAdress($id) {
```

```
$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

//$query = "select userpath, userpic, userdesc, userbalance from userprofile where userid = '
$query = "select * from adress where memberid = '$id'";

$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
     }
     $rows = $res->numRows();
     $i = 0;
     while($res->fetchInto($adress[$i]) and ($i < $rows-1)) {
$i++;
}
     if(PEAR::isError($res)) {
return false;
     }
     else {
return $adress;
}
}

function getTopTen()
{
$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
```

```
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "select * from content where visible=1 order by (votes/voted) desc limit 0, 10";
$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
     }
     $rows = $res->numRows();

     if($rows==0)
     return false;

     $i = 0;
     while($res->fetchInto($chart[$i]) and ($i < $rows-1)) {
$i++;
}
     if(PEAR::isError($res)) {
return false;
     }
     else
return $chart;
}

function customer_logged_in()
{
if(isset($_SESSION['customer']))
if(isset($_SESSION['id']))
if($_SESSION['customer'] == $_SESSION['id'])
return $_SESSION['id'];
return false;
}

function author_logged_in()
{
if(isset($_SESSION['author']))
```

```
if(isset($_SESSION['id']))
if($_SESSION['author'] == $_SESSION['id'])
return $_SESSION['id'];
return false;
}

function admin_logged_in()
{
if(isset($_SESSION['admin']))
if(isset($_SESSION['id']))
if($_SESSION['admin'] == $_SESSION['id'])
return $_SESSION['id'];
return false;
}

function check_valid_user()
{
//check if a user is logged in

if(isset($_SESSION['id']))
return $_SESSION['id'];
else
return false;
}

function getSandbox()
{
$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "select * from content where visible=0";
$res =& $db->query($query);
```

```
if(PEAR::isError($res)) {
return false;
     }
     $rows = $res->numRows();

     if($rows==0)
     return false;

     $i = 0;
     while($res->fetchInto($content[$i]) and ($i < $rows-1)) {
$i++;
}
     if(PEAR::isError($res)) {
return false;
     }
     else
return $content;
}

function setVisible($ownerid, $contentid) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "update content set visible=1 where ownerid='$ownerid' and id='$contentid'";

$res =& $db->query($query);
if (PEAR::isError($res)) {
return false;
     }
     return true;
```

```
}

function setNotVisible($ownerid, $contentid) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "update content set visible=0 where ownerid='$ownerid' and id='$contentid'";

$res =& $db->query($query);
if (PEAR::isError($res)) {
echo $res->getMessage();
exit;
return false;
     }
     return true;
}

function searchUser($query) {

$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);
```

```
/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
     }
     $rows = $res->numRows();

     if($rows==0)
     return false;

     $i = 0;
     while($res->fetchInto($userinfo[$i]) and ($i < $rows-1)) {
$i++;
}
     if(PEAR::isError($res)) {
return false;
     }
     else
return $userinfo;
}

function count_items($cart)
{
$items = 0;
if(is_array($cart))
$items = array_sum($cart);
return $items;
}

function getContentPrice($ownerid, $contentid)
{
$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
```

```
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "select price from content where ownerid = '$ownerid' and id = '$contentid' and visi

$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
}

$rows = $res->numRows();

     if($rows==0)
     return false;

$res->fetchInto($row);
$price = $row[0];
return $price;

}


function getCustomerBalance($customerid)
{
$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
```

```
return false;
}

$query = "select balance from members where id = '$customerid' and active=1";

$res =& $db->query($query);

if(PEAR::isError($res))
return false;

$rows = $res->numRows();

if($rows==0)
     return false;

$res->fetchInto($row);
$balance = $row[0];
return $balance;
}

function registerorder($contentid, $ownerid, $customerid)
{
$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}
$date = date("Y-m-d H:i:s");
$query = "insert into orders values(null, '$date', '$contentid', '$ownerid', '$customerid')";

$res =& $db->query($query);

if(PEAR::isError($res))
{
echo $res->getMessage();
```

```
exit;
return false;
}
return true;
}

function updatecustomer($customerid, $amount)
{
$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}
$date = date("Y-m-d H:i:s");
$query = "update members set balance=balance-'$amount' where id='$customerid'";

$res =& $db->query($query);

if(PEAR::isError($res))
return false;

return true;
}

function updateauthor($userid, $amount)
{
$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
```

```
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "update userprofile set userbalance=userbalance+'$amount' where userid='$userid'";

$res =& $db->query($query);

if(PEAR::isError($res))
return false;

return true;
}


function updatecontentdownload($contentid, $ownerid)
{
$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "update content set downloaded=downloaded+1 where id='$contentid' and ownerid='$owne

$res =& $db->query($query);

if(PEAR::isError($res))
```

```
return false;

return true;
}

function getOrders($memberid)
{
$dsn = array(
     'phptype'  => 'mysql',
     'username' => 'root',
     'password' => 'datanord',
     'hostspec' => 'localhost',
     'database' => 'hemsidan',
);

$options = array(
'debug'       => 2,
'portability' => DB_PORTABILITY_ALL,
);

/* connect to database */

$db =& DB::connect($dsn, $options);
if (PEAR::isError($db)) {
return false;
}

$query = "select orderid, date, content.title, content.type, content.author, content.price  fi
$res =& $db->query($query);

if(PEAR::isError($res)) {
return false;
     }
     $rows = $res->numRows();

     if($rows==0)
     return false;

     $i = 0;
     while($res->fetchInto($orders[$i]) and ($i < $rows-1)) {
$i++;
}
     if(PEAR::isError($res)) {
return false;
     }
     else
return $orders;
}
?>
```

```
<?php

/*
 *  sourcecode for history.php
 */

    session_start();

require('Smarty.class.php');
require_once "functions.php";

$smarty = new Smarty;

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

//if(check_valid_user()) {
if(customer_logged_in()) {

  $user = $_SESSION['username'];
  $id = $_SESSION['id'];

/* get customers order history */

$orders = getOrders($id);

if($orders) {

$smarty->display('header2.tpl');
    $smarty->display('membermenu.tpl');
    $smarty->assign('orders',$orders);
$smarty->display('member.tpl');
$smarty->display('footer2.tpl');
}
else {

    $smarty->display('header2.tpl');
    $smarty->display('membermenu.tpl');
    $message = "Your order history is empty";
    $smarty->assign('message',$message);
$smarty->display('member.tpl');
$smarty->display('footer2.tpl');
}
}
else
header("Location: http://localhost/memberlogin2.php");
?>
```

```
<?php

/*
 *  sourcecode for login.php
 */

require_once "HTML/QuickForm.php";
   require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
   require_once "functions.php";
require('Smarty.class.php');

$smarty = new Smarty;
$loginform = new HTML_QuickForm('login', 'post');

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

/* Add elements to form */

   $loginform->addElement('text', 'username', 'Enter username', array('maxlength' => 20));
   $loginform->addElement('password', 'password', 'Enter your password', array('maxlength' =>
   $loginform->addElement('submit','sb','Login');

   /* Add rules to form */

   $loginform->addRule('username', 'You need to type your username', 'required');
   $loginform->applyFilter('username', 'trim');
   $loginform->addRule('password', 'You need to type your password', 'required');
   $loginform->applyFilter('password', 'trim');

   /* validate the loginform */

if ($loginform->validate()) {

/* now check with database and if ok redirect to userpage */

$usrName = $_POST['username'];
$pswd = $_POST['password'];

session_start();

if(loginOwner($usrName, $pswd)) {

$_SESSION['username'] = $usrName;

/* redirect to userpage */
```

```
header("Location: http://localhost/user.php");
}
else {

/* login failed */

$rendererLogin = new HTML_QuickForm_Renderer_ArraySmarty($smarty);

     $loginform->accept($rendererLogin);

$title = "Yank Yank - Author login";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
    $smarty->assign('loginform', $rendererLogin->toArray());
    $error = "Login failed";
$smarty->assign('error', $error);
$smarty->display('content2.tpl');
    $smarty->display('footer2.tpl');
exit;
}
}
else {

/* Show the login form */

$rendererLogin = new HTML_QuickForm_Renderer_ArraySmarty($smarty);

$loginform->accept($rendererLogin);
  $title = "Yank Yank - Author login";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
    $smarty->assign('loginform', $rendererLogin->toArray());
$smarty->display('content2.tpl');
    $smarty->display('footer2.tpl');
}
?>
```

```
<?php

/*
 *  sourcecode for logout.php
 */

require_once "functions.php";

session_start();

if(isset($_GET['type']))
```

```
{
if(logout($_GET['type']))
{
/* user succeded to logout */
header("Location: http://localhost/index2.php");
}
}
else
{
if(logout(""))
{
/* user succeded to logout */
header("Location: http://localhost/index2.php");
}
}
?>
```

```
<?php

/*
 *  sourcecode for makepay.php
 */

session_start();

require_once('Smarty.class.php');
require_once "functions.php";
    require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
    require_once "HTML/QuickForm.php";

$smarty = new Smarty;
$paymentform = new HTML_QuickForm('pay', 'post');

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

/* Add elements to form */

    $options = array(
          'language'  => 'swe',
          'format'    => 'ym',
          'minYear'   => 2000,
          'maxYear'   => 2014
    );

    $attributes = array('size'=>'1');
$paymentform->addElement('text', 'cardnumber', 'Card Number', array('maxlength' => 16, 'size'
$paymentform->addElement('text', 'ctrlnr', 'Three digits', array('maxlength' => 3, 'size' =>
```

```
$paymentform->addElement('date', 'expdate', 'Exparations date: (Y/M))', $options);
$paymentform->addElement('text', 'name', 'Name', array('maxlength' => 40, 'size' => 20));
$paymentform->addElement('submit','sb','Go');

/* Add rules to fields */

$paymentform->addRule('cardnumber', 'Can only contain digits', 'numeric');
$paymentform->addRule('cardnumber', 'Please type card number', 'required');
$paymentform->applyFilter('cardnumber', 'trim');
$paymentform->addRule('ctrlnr', 'Can only contain digits', 'numeric');
$paymentform->addRule('ctrlnr', 'Please type three digits', 'required');
$paymentform->applyFilter('ctrlnr', 'trim');
$paymentform->addRule('name', 'Please type your name', 'required');
$paymentform->applyFilter('name', 'trim');

/* check the searchform */

if ($paymentform->validate())
{
   echo 'redirect to https server or bank server';
exit;
}
else
{
$rendererPayment = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $paymentform->accept($rendererPayment);
  $title = "Yank Yank";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('membermenu.tpl');
$smarty->assign('paymentform', $rendererPayment->toArray());
$smarty->display('member.tpl');
    $smarty->display('footer2.tpl');
   }
?>
```

```
<?php

/*
 *  sourcecode for manage.php
 */

    session_start();

require('Smarty.class.php');
require_once "functions.php";
require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
    require_once "HTML/QuickForm.php";

    $smarty = new Smarty;
```

```
$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

$manageform = new HTML_QuickForm('manage', 'post');

/* present form fields */

   $manageform->addElement('textarea', 'description', 'Description', array('maxlength' => 200

   $attributes = array('size'=>'1');
   $file =& $manageform->addElement('file', 'file', 'Choose picture');
   $manageform->addElement('submit','sb','Save info');

   /* add rules to fields */

    /* max file size for a photo is 256 kB */

$maxfilesize = 256*1024;

   $manageform->addRule('file','Your file is too big','maxfilesize', $maxfilesize);

//if(check_valid_user()) {
if(author_logged_in()) {

/* present form fields */

  $user = $_SESSION['username'];


  if ($manageform->validate()) {

    /* create path to the users content */

$path = 'c:\\root\\public\\users\\'.$_SESSION['id'].'\\';

    if(uploadfile($path))
    {
     /* now put info into database */

     $values['table'] = 'userprofile';
     $values['id'] = $_SESSION['id'];
     $values['description'] = $_POST['description'];
     $values['filename'] = $_FILES['file']['name'];
$values['path'] = 'c:\\\root\\\public\\\users\\\\'.$_SESSION['id'];

     if(putToDatabase($values))
     {
```

```
     /* display the page content */

$smarty->display('header2.tpl');
    $smarty->display('authormenu.tpl');
    $message = "Info saved";
$smarty->assign('message', $message);
    $smarty->display('user.tpl');
$smarty->display('footer2.tpl');
exit;
}
}
else {

/* display the page content */

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $manageform->accept($renderer);

$smarty->display('header2.tpl');
$edit = "edit";
$smarty->assign('edit', $edit);
    $smarty->display('authormenu.tpl');
    $smarty->assign('manageform',$renderer->toArray());
    $err = "Failed to save userinfo";
$smarty->assign('err', $err);
    $smarty->display('user.tpl');
$smarty->display('footer2.tpl');

}
}
else {

/* display the page content */

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
    $manageform->accept($renderer);

$smarty->display('header2.tpl');
    $edit = "edit";
$smarty->assign('edit', $edit);
    $smarty->display('authormenu.tpl');
    $smarty->assign('manageform',$renderer->toArray());
    $smarty->display('user.tpl');
$smarty->display('footer2.tpl');
}

}
else
echo 'Not authorized';
```

```
?>


<?php

/*
 *  sourcecode for member.php
 */

    session_start();

require('Smarty.class.php');
require_once "functions.php";

$smarty = new Smarty;

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

//if(check_valid_user()) {
if(customer_logged_in()) {

  $user = $_SESSION['username'];
  $id = $_SESSION['id'];

/* get user info */

$info = getMemberInfo($id);
$adress = getMemberAdress($id);

if($info) {

$memberinfo = $info[0];
  $smarty->display('header2.tpl');
    $smarty->display('membermenu.tpl');
    $message = "Logged in as ".$user;
    $smarty->assign('message',$message);
    if($adress) {
$memberadress = $adress[0];
    $smarty->assign('memberadress',$memberadress);
    }
$smarty->assign('memberinfo',$memberinfo);
$smarty->display('member.tpl');
$smarty->display('footer2.tpl');
}
else {
echo 'error';
```

```
exit;
}
}
else
header("Location: http://localhost/memberlogin2.php");
?>

<?php

/*
 *  sourcecode for memberlogin.php
 */

require_once "HTML/QuickForm.php";
   require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
   require_once "functions.php";
require('Smarty.class.php');

$smarty = new Smarty;
$loginform = new HTML_QuickForm('login', 'post');

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

/* Add elements to form */

   $loginform->addElement('text', 'username', 'Enter username', array('maxlength' => 20));
   $loginform->addElement('password', 'password', 'Enter your password', array('maxlength' =>
   $loginform->addElement('submit','sb','Login');

   /* Add rules to form */

   $loginform->addRule('username', 'You need to type your username', 'required');
   $loginform->applyFilter('username', 'trim');
   $loginform->addRule('password', 'You need to type your password', 'required');
   $loginform->applyFilter('password', 'trim');

   /* validate the loginform */

if ($loginform->validate()) {

/* now check with database and if ok redirect to userpage */

$usrName = $_POST['username'];
$pswd = $_POST['password'];

session_start();
```

```
if(loginMember($usrName, $pswd)) {

$_SESSION['username'] = $usrName;

/* redirect to userpage */

header("Location: http://localhost/member.php");
}
else {

/* login failed */

$rendererLogin = new HTML_QuickForm_Renderer_ArraySmarty($smarty);

     $loginform->accept($rendererLogin);

$title = "Yank Yank - member login";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
    $smarty->assign('loginform', $rendererLogin->toArray());
    $error = "Login failed";
$smarty->assign('error', $error);
$smarty->display('content2.tpl');
    $smarty->display('footer2.tpl');
exit;
}
}
else {

/* Show the login form */

$rendererLogin = new HTML_QuickForm_Renderer_ArraySmarty($smarty);

$loginform->accept($rendererLogin);
  $title = "Yank Yank - member login";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
$smarty->assign('loginform', $rendererLogin->toArray());

/* show message if directed here from another page */

if(isset($_REQUEST['login']))
{
$message = "Not logged in?, please login to make a purchase or sign up to get an account";
$smarty->assign('message', $message);
$smarty->display('content2.tpl');
    $smarty->display('footer2.tpl');
```

```
}
else
{
$smarty->display('content2.tpl');
$smarty->display('footer2.tpl');
}

}
?>
```

```
<?php

/*
 *  sourcecode for order.php
 */

session_start();

require_once('Smarty.class.php');
require_once "functions.php";

$smarty = new Smarty;

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';


if($customerid = customer_logged_in())
{

/* display the page content */

if(isset($_REQUEST['oi']) && isset($_REQUEST['ci']))
{
$ownerid = $_REQUEST['oi'];
$contentid = $_REQUEST['ci'];

$title = "Yank Yank - purchase";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('membermenu.tpl');
$purchase = getSingleContent($ownerid, $contentid);
$smarty->assign('purchase', $purchase);
    $smarty->display('member.tpl');
$smarty->display('footer2.tpl');

}
}
```

```
else
{
/* If customer not logged in, redirect to customer login */

header("Location: http://localhost/memberlogin2.php?login");
}

//kolla om inloggad om inte meddela att logga in eller skapa konto

//om inloggad kolla med virtual billing

//om giltigt köp updatera konto, skicka mail med kvitto.
//lägg till order i tabell

//visa pop_up med nedladdnings sida

//annars meddela fyll på pengar
?>

<?php

/*
 *  sourcecode for owner.php
 */

    session_start();

require('Smarty.class.php');
require_once "functions.php";

$smarty = new Smarty;

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';


if(isset($_REQUEST['id'])) {

$id = $_REQUEST['id'];
$info = getUserInfo($id);

if($info) {

$news = getNews($id);
$content = getContent($id);

/* display the page content */
```

```
$userinfo = $info[0];
if($userinfo[4]!=null)
$image = "http://localhost/users/".$id."/".$userinfo[4];
else
$image = "";
$info = $userinfo[5];

  $smarty->display('header2.tpl');
    $smarty->display('menu2.tpl');
    $smarty->assign('news',$news);
    $smarty->assign('content',$content);
$smarty->assign('info',$info);
    $smarty->assign('image',$image);
    $smarty->display('owner.tpl');
$smarty->display('footer2.tpl');
}
else {
echo 'Couldnt get user info pls try again later';
header("Location: http://localhost/index2.php");
}
}

?>
```

```
<?php

/*
 *  sourcecode for purchase.php
 */

session_start();

require_once "functions.php";

$customerid = $_SESSION['id'];
$ownerid = $_POST['ownerid'];
$contentid = $_POST['contentid'];

$userbalance = getCustomerBalance($customerid);
$contentprice = getContentPrice($ownerid,$contentid);

if($userbalance >= $contentprice)
{

if(registerorder($contentid, $ownerid, $customerid))
{
updatecustomer($customerid, $contentprice);
updatecontentdownload($contentid, $ownerid);
updateauthor($ownerid, $contentprice);
header("Location: http://localhost/download.php");
```

```
exit;
}

}
else
header("Location: http://localhost/member.php?prepay");

//kolla om inloggad om inte meddela att logga in eller skapa konto

//om inloggad kolla med virtual billing

//om giltigt köp updatera konto, skicka mail med kvitto.
//lägg till order i tabell

//visa pop_up med nedladdnings sida

//annars meddela fyll på pengar
?>

<?php

/*
 *  sourcecode for rate.php
 */

require_once "functions.php";
session_start();

if(isset($_REQUEST['mode']))
{
if ( $_REQUEST['mode']=="vote")
{

$values['voted'] = $_REQUEST['voted'];
$values['ownerid'] = $_REQUEST['oi'];
$values['contentid'] = $_REQUEST['ci'];
$ownerindex = $_REQUEST['oi'];
$contentindex = $_REQUEST['ci'];

/* First time a user with content is rated */

if(!isset($_SESSION['$ownerindex']))
{
$_SESSION['$ownerindex'] = $_REQUEST['oi'];

if(!isset($_SESSION['$contentindex']))
{
if(rateContent($values))
{
$_SESSION['$contentindex'] = $_REQUEST['ci'];
```

```
header("Location: http://localhost/index2.php");
}
else
{
echo 'rating failed';
exit;
}
}
}

/* A user has been voted before but now check that its not the same content */

else
{
if($_SESSION['$contentindex'] != $contentindex)
{
if(rateContent($values))
{
$_SESSION['$contentindex'] = $_REQUEST['ci'];
header("Location: http://localhost/index2.php");
}
else
{
echo 'rating failed';
exit;
}
}
header("Location: http://localhost/index2.php");
}
}
}
?>

<?php

/*
 *  sourcecode for register.php
 */

require('Smarty.class.php');
require_once "functions.php";
    require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
    require_once "HTML/QuickForm.php";

    $smarty = new Smarty;
$registerform = new HTML_QuickForm('register', 'post');

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
```

```
$smarty->compile_dir = 'c:/smarty/templates_c';

/* Add elements to field */

$registerform->addElement('text', 'username', 'Enter username',array('maxlength' => 20));
   $registerform->addElement('password', 'password', 'Enter your password',array('maxlength' =
   $registerform->addElement('password', 'password2', 'Confirm password',array('maxlength' =>
$registerform->addElement('text', 'email', 'Enter email',array('maxlength' => 50));
  $registerform->addElement('text', 'company', 'Company name',array('maxlength' => 40));
  $registerform->addElement('text', 'adressone', 'Adress one',array('maxlength' => 40));
  $registerform->addElement('text', 'adresstwo', 'Adress two',array('maxlength' => 40));
  $registerform->addElement('text', 'zip', 'Zip-code',array('maxlength' => 8));
  $registerform->addElement('text', 'town', 'Town',array('maxlength' => 40));
  $registerform->addElement('text', 'country', 'Country',array('maxlength' => 40));
   $registerform->addElement('submit','sb','register');

   /* add rules to fields */

   $registerform->applyFilter('username', 'trim');
   $registerform->addRule('username', 'You need to type a username', 'required');
   /*$registerform->registerRule('check', 'callback', 'check_username', 'Validate');
$registerform->addRule('username', 'Username already taken', 'check_username', null, 'client'

   $registerform->addRule('email', 'Your email is required', 'required');
   $registerform->applyFilter('email', 'trim');
   $registerform->addRule('email', 'Not a valid email', 'email');

   $registerform->addRule('password', 'Your need to type a password', 'required');
   $registerform->applyFilter('password', 'trim');
   $registerform->addRule('password2', 'You need to confirm your password', 'required');
   $registerform->applyFilter('password2', 'trim');
   $registerform->addRule(array('password', 'password2'), 'The passwords do not match', 'comp

   $registerform->applyFilter('company', 'trim');
   $registerform->applyFilter('adressone', 'trim');
   $registerform->addRule('adressone', 'You need to type an adress', 'required');
   $registerform->applyFilter('adresstwo', 'trim');
   $registerform->addRule('zip', 'Can only contain digits', 'numeric');
   $registerform->addRule('zip', 'You must type a zip-code ', 'required');
   $registerform->applyFilter('zip', 'trim');
   $registerform->applyFilter('town', 'trim');
   $registerform->addRule('town', 'You need to type a town', 'required');
   $registerform->addRule('country', 'You need to type a country', 'required');
$registerform->applyFilter('country', 'trim');


   /* check that fields are valid */

   if ($registerform->validate()) {
```

```
/* check that username is unique */

if(isUnique("user", "username", $_POST['username'])) {


/* now add a new user to the database */

$values['username'] = $_POST['username'];
$values['password'] = $_POST['password'];
$values['email'] = $_POST['email'];
$values['company'] = $_POST['company'];
$values['adressone'] = $_POST['adressone'];
$values['adresstwo'] = $_POST['adresstwo'];
$values['zip'] = $_POST['zip'];
$values['town'] = $_POST['town'];
$values['country'] = $_POST['country'];

if(register($values)) {

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $registerform->accept($renderer);

/* display the page content */

$title = "Yank Yank - register Author";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
    $smarty->display('menu2.tpl');
    $message = "Registration success";
$smarty->assign('message', $message);
//$smarty->assign('registerform',$renderer->toArray());
$smarty->display('content2.tpl');
$smarty->display('footer2.tpl');
}
else {

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $registerform->accept($renderer);

/* display the page content */

$title = "Yank Yank - register Author";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
    $error = "Registration failed";
$smarty->assign('error', $error);
$smarty->assign('registerform',$renderer->toArray());
```

```
$smarty->display('content2.tpl');
$smarty->display('footer2.tpl');
}
}
else {

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $registerform->accept($renderer);

/* display the page content */

$title = "Yank Yank - register Author";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
    $error = "Username already taken, please choose another one";
$smarty->assign('error', $error);
$smarty->assign('registerform',$renderer->toArray());
$smarty->display('content2.tpl');
$smarty->display('footer2.tpl');
}
   }
   else {

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $registerform->accept($renderer);

/* display the page content */

$title = "Yank Yank - Register Author";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
$smarty->assign('registerform',$renderer->toArray());
    $smarty->display('content2.tpl');
$smarty->display('footer2.tpl');
    }
?>
```

```
<?php

/*
 *  sourcecode for registermember.php
 */

require('Smarty.class.php');
require_once "functions.php";
    require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
    require_once "HTML/QuickForm.php";
```

```
    $smarty = new Smarty;
$registerform = new HTML_QuickForm('register', 'post');

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

/* Add elements to field */

$registerform->addElement('text', 'username', 'Enter username', array('maxlength' => 20));
   $registerform->addElement('password', 'password', 'Enter your password', array('maxlength'
   $registerform->addElement('password', 'password2', 'Confirm password', array('maxlength' =
$registerform->addElement('text', 'email', 'Enter email', array('maxlength' => 50));
  $registerform->addElement('text', 'adressone', 'Adress', array('maxlength' => 40));
  $registerform->addElement('text', 'zip', 'Zip-code', array('maxlength' => 8, 'size' => 8));
  $registerform->addElement('text', 'town', 'Town', array('maxlength' => 40));
  $registerform->addElement('text', 'country', 'Country', array('maxlength' => 40));
   $registerform->addElement('submit', 'sb', 'register');

   /* add rules to fields */

   $registerform->applyFilter('username', 'trim');
   $registerform->addRule('username', 'You need to type a username', 'required');

   $registerform->addRule('email', 'Your email is required', 'required');
   $registerform->applyFilter('email', 'trim');
   $registerform->addRule('email', 'Not a valid email', 'email');

   $registerform->addRule('password', 'Your need to type a password', 'required');
   $registerform->applyFilter('password', 'trim');
   $registerform->addRule('password2', 'You need to confirm your password', 'required');
   $registerform->applyFilter('password2', 'trim');
   $registerform->addRule(array('password', 'password2'), 'The passwords do not match', 'comp

   $registerform->applyFilter('adressone', 'trim');
   $registerform->addRule('adressone', 'You need to type an adress', 'required');
   $registerform->addRule('zip', 'Can only contain digits', 'numeric');
   $registerform->addRule('zip', 'You must type a zip-code ', 'required');
   $registerform->applyFilter('zip', 'trim');
   $registerform->applyFilter('town', 'trim');
   $registerform->addRule('town', 'You need to type a town', 'required');
   $registerform->applyFilter('country', 'trim');
   $registerform->addRule('country', 'You need to type a country', 'required');

   /* check that fields are valid */

   if ($registerform->validate()) {
```

```
/* check that username is unique */

if(isUnique("members", "membername", $_POST['username'])) {

/* now add a new user to the database */

$values['username'] = $_POST['username'];
$values['password'] = $_POST['password'];
$values['email'] = $_POST['email'];
$values['adressone'] = $_POST['adressone'];
$values['zip'] = $_POST['zip'];
$values['town'] = $_POST['town'];
$values['country'] = $_POST['country'];

if(registerMember($values)) {

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $registerform->accept($renderer);

/* display the page content */

$title = "Yank Yank - Sign up";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
    $smarty->display('menu2.tpl');
    $error = "Registration success";
$smarty->assign('error', $error);
//$smarty->assign('registerform',$renderer->toArray());
$smarty->display('content.tpl');
$smarty->display('footer.tpl');
}
else {

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $registerform->accept($renderer);

/* display the page content */

$title = "Yank Yank - Sign up";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
    $error = "Registration failed";
$smarty->assign('error', $error);
$smarty->assign('registerform',$renderer->toArray());
$smarty->display('content2.tpl');
$smarty->display('footer2.tpl');
}
}
```

```
else {

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $registerform->accept($renderer);

/* display the page content */

$title = "Yank Yank - Sign up";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
    $error = "Username already taken, please choose another one";
$smarty->assign('error', $error);
$smarty->assign('registerform',$renderer->toArray());
$smarty->display('content2.tpl');
$smarty->display('footer2.tpl');
}
   }
   else {

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $registerform->accept($renderer);

/* display the page content */

$title = "Yank Yank - Sign up";
$smarty->assign('title', $title);
$smarty->display('header2.tpl');
$smarty->display('menu2.tpl');
$smarty->assign('registerform',$renderer->toArray());
    $smarty->display('content2.tpl');
$smarty->display('footer2.tpl');
    }
?>
```

```
<?php

/*
 *  sourcecode for sandbox.php
 */

/* Admin home page */

session_start();

require('Smarty.class.php');
require_once "functions.php";

$smarty = new Smarty;
```

```
$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

//if(check_valid_user()) {
if(admin_logged_in()) {

  $user = $_SESSION['username'];
  $id = $_SESSION['id'];

$smarty->display('header2.tpl');
$smarty->display('adminmenu.tpl');
    $content = getSandbox();
    if($content)
$smarty->assign('content',$content);
else {
$message = "Sandbox is empty";
$smarty->assign('message',$message);
}

    $smarty->display('contentAdmin.tpl');
    $smarty->display('footer2.tpl');
}
else
echo 'Not authorized';
?>
```

```
<?php

/*
 *  sourcecode for search.php
 */

session_start();

require('Smarty.class.php');
require_once "functions.php";
    require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
    require_once "HTML/QuickForm.php";

$smarty = new Smarty;
$usersearchform = new HTML_QuickForm('search', 'get');

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

/* Add elements to form */
```

```
   $usersearchform->addElement('text', 'searchtext', '', array('maxlength' => 20));
   $usersearchform->addElement('radio','usertype','Search for', 'Member', 'members');
   $usersearchform->addElement('radio','usertype', null, 'Author', 'owner');
$usersearchform->addElement('radio','usertype', null, 'Content', 'content');
$usersearchform->addElement('submit','sb','Search');

/* Set default radiobutton */
$usersearchform->setDefaults(array('usertype' => 'members'));
   /* Add rules to fields */

$usersearchform->addRule('searchtext', '', 'required');
$usersearchform->applyFilter('searchtext', 'trim');



/* check the searchform */

   if ($usersearchform->validate()) {

$searchString = $_GET['searchtext'];
$table = $_GET['usertype'];

if($table=="members")
$query = "select * from members where membername like \"%$searchString%\" or id = '$searchStr
else if($table=="user")
$query = "select * from user where username like \"%$searchString%\" or id = '$searchString'"
else if($table=="content")
$query = "select * from content where title like \"%$searchString%\" or id = '$searchString' o
else
$query = "select * from user where username like \"%$searchString%\" or id = '$searchString'"

/* if we get a match display hits */

$usersearch = searchUser($query);

if($usersearch) {

/* display the page and searchresult */

$smarty->display('header2.tpl');
$smarty->display('adminmenu.tpl');
if($table=="members")
$smarty->assign('members', $usersearch);
    else if($table=="user")
    $smarty->assign('user', $usersearch);
    else if($table=="content")
    $smarty->assign('contents', $usersearch);
    else
```

```
    $smarty->assign('user', $usersearch);

    $smarty->display('contentAdmin.tpl');
    $smarty->display('footer2.tpl');
}
else {

/* else tell that were no hits on that search */

$rendererSearch = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $usersearchform->accept($rendererSearch);
$smarty->display('header2.tpl');
$smarty->display('adminmenu.tpl');
    $nomatch = "Your search resulted in no match";
    $smarty->assign('nomatch', $nomatch);
    $smarty->assign('usersearchform', $rendererSearch->toArray());
    $smarty->display('contentAdmin.tpl');
    $smarty->display('footer2.tpl');
}
}
else {


$rendererSearch = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $usersearchform->accept($rendererSearch);

$smarty->display('header2.tpl');
$smarty->display('adminmenu.tpl');
    $smarty->assign('usersearchform', $rendererSearch->toArray());
    $smarty->display('contentAdmin.tpl');
    $smarty->display('footer2.tpl');
   }
?>
```

```
<?php

/*
 *  sourcecode for setinvisible.php
 */

    session_start();

require('Smarty.class.php');
require_once "functions.php";

    $smarty = new Smarty;
$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';
```

```
//if(check_valid_user()) {
if(admin_logged_in()) {

  $user = $_SESSION['username'];


  if(isset($_REQUEST['vis']) && isset($_REQUEST['cid']) && isset($_REQUEST['oid'])) {

if($_REQUEST['vis']==0) {
$ownerid = $_REQUEST['oid'];
$contentid = $_REQUEST['cid'];

if(setNotVisible($ownerid, $contentid)) {

  $smarty->display('header2.tpl');
    $smarty->display('adminmenu.tpl');
    $message = "Content unset";
$smarty->assign('message',$message);
    $smarty->display('contentAdmin.tpl');
    $smarty->display('footer2.tpl');
exit;
}
}
}
$smarty->display('header2.tpl');
$smarty->display('adminmenu.tpl');
$message = "Failed to unset content, pls try again later";
$smarty->assign('message',$message);
$smarty->display('contentAdmin.tpl');
$smarty->display('footer2.tpl');


  }
else
echo 'Not authorized';
?>
```

```
<?php

/*
 *  sourcecode for upload.php
 */

    session_start();

require('Smarty.class.php');
require_once "functions.php";
require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
    require_once "HTML/QuickForm.php";
```

```
    $smarty = new Smarty;
$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

$uploadform = new HTML_QuickForm('upload', 'post');


/* present form fields */

   $uploadform->addElement('text', 'title', 'Title', array('maxlength' => 40));
   $uploadform->addElement('text', 'author', 'Author', array('maxlength' => 40));
   $uploadform->addElement('textarea', 'description', 'Description', array('maxlength' => 400
   $options = array('music'=>'Music','ebook'=>'Ebook','photo'=>'Photo');
   $attributes = array('size'=>'1');
   $uploadform->addElement('select', 'type', 'Type',$options,$attributes);
   $uploadform->addElement('text', 'price', 'Price');
   $file =& $uploadform->addElement('file', 'file', 'Choose file');
   $uploadform->addElement('submit','sb','Upload');

   /* add rules to fields */

   $uploadform->addRule('title', 'You need to type title', 'required');
   $uploadform->applyFilter('title', 'trim');
   $uploadform->addRule('author', 'You need to type author', 'required');
   $uploadform->applyFilter('author', 'trim');
   $uploadform->addRule('price', 'You need to add a price', 'required');
   $uploadform->applyFilter('price', 'trim');
   $uploadform->addRule('price', 'Can only contain digits', 'numeric');
   $uploadform->addRule('file', 'You must select a file', 'uploadedfile');

/* max file size for content is 8Mb */

$maxfilesize = 8*1024*1024;

   $uploadform->addRule('file','Your file is too big','maxfilesize', $maxfilesize);
   $uploadform->addRule('file', 'The extension of the Song file should be *.mp3', 'filename',



//if(check_valid_user()) {
if(author_logged_in()) {

/* present form fields */

  $user = $_SESSION['username'];
```

```
  if ($uploadform->validate()) {

    $id = $_SESSION['id'];
    $path = "c:\\root\\public\\users\\".$id."\\";


    if(uploadfile($path))
    {
     /* now put info into database */

     $values['table'] = 'content';
     $values['id'] = $_SESSION['id'];
     $values['title'] = $_POST['title'];
     $values['author'] = $_POST['author'];
     $values['description'] = $_POST['description'];
     $values['type'] = $_POST['type'];
     $values['price'] = $_POST['price'];
     $values['filename'] = $_FILES['file']['name'];
$values['path'] = 'c:\\\root\\\public\\\users\\'.$id."\\";

     if(putToDatabase($values))
     {
     /* display the page content */

$smarty->display('header.tpl');
$upload = "upload";
$smarty->assign('upload', $upload);
    $smarty->display('usermenu.tpl');
    $message = "Upload completed";
$smarty->assign('message', $message);
    $smarty->display('user.tpl');
$smarty->display('footer.tpl');
}
}
else {

/* display the page content */

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $uploadform->accept($renderer);

$smarty->display('header2.tpl');
$smarty->display('authormenu.tpl');
    $smarty->assign('uploadform',$renderer->toArray());
    $err = "Upload failed";
$smarty->assign('err', $err);
    $smarty->display('user.tpl');
$smarty->display('footer2.tpl');
```

```
}
}
else {

/* display the page content */

$renderer = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
    $uploadform->accept($renderer);

$smarty->display('header2.tpl');
    $smarty->display('authormenu.tpl');
    $smarty->assign('uploadform',$renderer->toArray());
    $smarty->display('user.tpl');
$smarty->display('footer2.tpl');

}

}
else
echo 'Not authorized';
?>



<?php

/*
 *  sourcecode for user.php
 */

    session_start();

require('Smarty.class.php');
require_once "functions.php";

$smarty = new Smarty;

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

//if(check_valid_user()) {
if(author_logged_in()) {

  $user = $_SESSION['username'];
  $id = $_SESSION['id'];

/* get user info */
```

```
$info = getUserInfo($id);
$adress = getUserAdress($id);

if($info) {

$userinfo = $info[0];
$userprofile[0] = $userinfo[1];
$userprofile[1] = $userinfo[2];
$userprofile[2] = $userinfo[3];
$userpic = $userinfo[4];
$userdesc = $userinfo[5];
$userprofile[3] = $userinfo[6];

if($userpic!=null)
$image = "http://localhost/users/".$id."/".$userpic;
else
$image = "";

  $smarty->display('header2.tpl');

    if(isset($_REQUEST['prev'])) {
    $smarty->display('authormenu.tpl');
    $message = "Logged in as ".$user;
    $news = getNews($id);
    $content = getContent($id);
    $smarty->assign('news',$news);
    $smarty->assign('content',$content);
     $smarty->assign('message',$message);
    $smarty->assign('info',$userdesc);
    $smarty->assign('image',$image);
    $smarty->display('userpreview.tpl');
    }
    else {
    $smarty->display('authormenu.tpl');
    $message = "Logged in as ".$user;
     $smarty->assign('message',$message);
    $smarty->assign('info',$info);
    $smarty->assign('image',$image);

if($adress) {
$useradress = $adress[0];
    $smarty->assign('useradress',$useradress);
    }
$smarty->assign('userprofile',$userprofile);
$smarty->display('user.tpl');

}
$smarty->display('footer2.tpl');
```

```
}
else {
echo 'error';
exit;
}
}
else
header("Location: http://localhost/login.php");
?>
```

```
<?php

/*
 *  sourcecode for userDisplay.php
 */

session_start();

require('Smarty.class.php');
require_once "functions.php";

$smarty = new Smarty;

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

//if(check_valid_user()) {
if(author_logged_in()) {

  $user = $_SESSION['username'];
  $id = $_SESSION['id'];

$content = getContent($id);

if($content) {

/* display the page and user content */

$smarty->display('header2.tpl');
    $smarty->display('authormenu.tpl');
    $smarty->assign('content', $content);
    $smarty->display('user.tpl');
$smarty->display('footer2.tpl');
}
else {

/* display that user has no content */
```

```
$smarty->display('header2.tpl');
    $smarty->display('authormenu.tpl');
    $nocontent = "You have no content in database";
    $smarty->assign('nocontent', $nocontent);
    $smarty->display('user.tpl');
$smarty->display('footer2.tpl');

}
}
else
echo 'Not authorized';
?>
```

```
<?php

/*
 *  sourcecode for usersearch.php
 */

session_start();

require('Smarty.class.php');
require_once "functions.php";
    require_once "HTML/QuickForm/Renderer/ArraySmarty.php";
    require_once "HTML/QuickForm.php";

$smarty = new Smarty;
$usersearchform = new HTML_QuickForm('search', 'get');

$smarty->template_dir = 'c:/root/public/smarty/templates';
$smarty->config_dir = 'c:/root/public/smarty/config';
$smarty->cache_dir = 'c:/smarty/cache';
$smarty->compile_dir = 'c:/smarty/templates_c';

/* Add elements to form */

   $usersearchform->addElement('text', 'searchtext', '', array('maxlength' => 20));
   $usersearchform->addElement('radio','usertype','Search for', 'member', 'members');
   $usersearchform->addElement('radio','usertype', null, 'owner', 'owner');
$usersearchform->addElement('radio','usertype', null, 'content', 'content');
$usersearchform->addElement('submit','sb','Search');

   /* Add rules to fields */

$usersearchform->addRule('searchtext', '', 'required');
$usersearchform->applyFilter('searchtext', 'trim');



/* check the searchform */
```

```
   if ($usersearchform->validate()) {

$searchString = $_GET['searchtext'];
$table = $_GET['usertype'];

if($table=="members")
$query = "select * from members where membername like \"%$searchString%\" or id = '$searchStr
else if($table=="user")
$query = "select * from user where username like \"%$searchString%\" or id = '$searchString'"
else if($table=="content")
$query = "select * from content where title like \"%$searchString%\" or id = '$searchString' o
else
$query = "select * from user where username like \"%$searchString%\" or id = '$searchString'"

/*if we get a match display hits */

$usersearch = searchUser($query);

if($usersearch) {

/* display the page and searchresult */

$smarty->display('header.tpl');
$searchuser = "searchuser";
$smarty->assign('searchuser', $searchuser);
$smarty->display('adminmenu.tpl');
if($table=="members")
$smarty->assign('members', $usersearch);
    else if($table=="user")
    $smarty->assign('user', $usersearch);
    else if($table=="content")
    $smarty->assign('contents', $usersearch);
    else
    $smarty->assign('user', $usersearch);

    $smarty->display('contentAdmin.tpl');
    $smarty->display('footer.tpl');
}
else {

/* else tell that were no hits on that search */

$rendererSearch = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $usersearchform->accept($rendererSearch);
$smarty->display('header.tpl');
$searchuser = "searchuser";
$smarty->assign('searchuser', $searchuser);
    $smarty->display('adminmenu.tpl');
```

```
    $nomatch = "Your search resulted in no match";
    $smarty->assign('nomatch', $nomatch);
    $smarty->assign('usersearchform', $rendererSearch->toArray());
    $smarty->display('contentAdmin.tpl');
    $smarty->display('footer.tpl');
}
}
else {


$rendererSearch = new HTML_QuickForm_Renderer_ArraySmarty($smarty);
     $usersearchform->accept($rendererSearch);

$smarty->display('header.tpl');
$searchuser = "searchuser";
$smarty->assign('searchuser', $searchuser);
    $smarty->display('adminmenu.tpl');
    $smarty->assign('usersearchform', $rendererSearch->toArray());
    $smarty->display('contentAdmin.tpl');
    $smarty->display('footer.tpl');
   }
?>
```