Bachelor's thesis

Degree programme in Information Technology

Digital Media

2012

Heikki Kesa

USER MANAGEMENT IMPLEMENTATION USING CAKEPHP FRAMEWORK

TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR'S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Degree programme in Information Technology | Digital Media

2012 | Total number of pages: 36

Instructors: Keijo Leinonen, B. Eng & Mika Luimula, Ph. D., Principal Lecturer

Heikki Kesa

USER MANAGEMENT IMPLEMENTATION USING CAKEPHP FRAMEWORK

Web development frameworks have gained lot of fans in recent years. Frameworks are designed to make web developers life easier by providing dynamic and tested built in functions for the most used aspects of web development. The framework used in this thesis is CakePHP.

This thesis starts out by introducing basic web development technologies and tools. Readers without previous knowledge will get basic understanding of web development before diving into frameworks and CakePHP.

The main part of the thesis is the CakePHP framework. First the concept of PHP –frameworks will be covered from the developer’s standpoint. Then the CakePHP’s MVC architecture will be examined. The project covered in this thesis is about creating a user management system, the CakePHP includes Access Control Lists as the primary user rights handling mechanism. Access Control Lists are explained in detail and examples are used to make readers understand how they work. Other CakePHP features are covered in lesser extent.

An example project has been made for this thesis that uses CakePHP Access Control Lists to create a user management system. The last part of this thesis is dedicated to this project. The projects database and structure will be demonstrated and explained, complete with code examples.

KEYWORDS:

(CakePHP, MVC, ACL, User Management, framework)

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tietotekniikan koulutusohjelma | Mediatekniikan suuntautumisvaihtoehto

2012 | Sivumäärä: 36

Ohjaajat: Ins. (AMK) Keijo Leinonen, Yliopettaja FT Mika Luimula

Heikki Kesa

KÄYTTÄJÄHALLINNAN TOTEUTTAMINEN CAKEPHP–SOVELLUSKEHYSTÄ KÄYTTÄEN

Web-ohjelmointikehykset ovat keränneet viimeisten vuosien aikana paljon kannattajia. Ohjelmointikehysten päätarkoitus on helpottaa web–kehittäjien elämää, antamalla heidän käyttöönsä hyväksi todettuja menetelmiä yleisimpien asioiden hoitoon. Tässä opinnäytetyössä käsitellään CakePHP –ohjelmointikehystä.

Opinnäytetyön alkuosassa käsitellään yleisimpiä web- tekniikoita ja työkaluja, jotta lukijat saavat ymmärryksen myöhemmin käsiteltävästä ohjelmointikehyksestä.

Opinnäytetyön keskeisin osa on CakePHP:n tutkiminen. Aluksi esitellään web-ohjelmointikehys –käsite yleisellä tasolla kehittäjän näkökulmasta. Tämän jälkeen käydään läpi CakePHP:n käyttämä MVC-arkkitehtuuri. Tämän opinnäytetyön projektina on ollut käyttäjähallinnan luominen, CakePHP:n käyttämä Access Control Lists –käyttöoikeus järjestelmä kuvataan esimerkkejä hyväksikäyttäen. Muut CakePHP:n osat käsitellään vähemmissä määrin.

Tätä opinnäytetyötä varten on tehty esimerkkiohjelma jossa on käytetty Access Control Lists –järjestelmää käyttäjähallinnan toteutuksessa. Opinnäytetyön viimeinen osa käsittelee kyseistä ohjelmaa. Ohjelman tietokanta- ja ohjelmarakenne kuvataan yleisellä tasolla ja syvemmin koodiesimerkkejä käyttäen.

ASIASANAT:

(CakePHP, MVC, ACL, käyttäjähallinta, ohjelmistokehys)

CONTENT

PROGRAMS

PICTURES

TABLES

LIST OF ABBREVIATIONS (OR) SYMBOLS

HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
PHP	PHP: Hypertext Preprocessor
SQL	Structured Query Language
AJAX	Asynchronous JavaScript and XML
MVC architecture	Model View Controller architecture
ACL	Access Control Lists
AROs	Access Request Objects
ACOs	Access Control Objects

1 INTRODUCTION

A client of this thesis Data Media Gazelle had an online price/specification comparison service called Hintajahti in development. It originally used a custom framework but was redesigned to use CakePHP –framework. My job was to create a user management system for it. This was first project where Data Media Gazelle used CakePHP, so some of the emphasis was also on researching the possibilities and limitations of the framework mainly related but not confined to user management.

Readers are expected to be at least familiar with basic web technologies, but basics will be covered and previous knowledge is not necessarily mandatory. The document starts by covering basics like HTML and PHP, the most important element of the thesis being the CakePHP –framework. CakePHP will be introduced to the readers that might not be familiar with web development framework concept. Other main parts are the MVC –architecture and Access Control Lists.

2 OPERATING ENVIRONMENT

2.1 Web Technologies

2.1.1 HTML and CSS

HTML (the Hypertext Markup Language) and CSS (Cascading Style Sheets) are the most basic technologies for building web pages.
HTML is used for defining positions and attributes of different elements that could be present on a web page. Basic web pages can be coded by using only HTML. HTML is not a programming language, but a markup language. It consist wide array of different tags, for example “table” tag for defining tables for tabular data or “img” for images. [1]

Program 1. Example of a HTML file that prints out a picture and some text.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

      <title>Welcome</title>

      <link href="style.css" rel="stylesheet" type="text/css" />

   </head>

   <body>
      <div id="wrap">
         <div id="header">
            <a href="index.html" id="headerlink" target="_top">
               <img src="images/header.png" />
            </a>
         </div>
      </div>
   </body>
</html>
```

CSS is used for defining styles of HTML elements. CSS allows web developers to change the size, color, visibility and many other properties. CSS can be also used to attach background images to elements. Web pages with same HTML file can look entirely different if using different CSS. CSS is not restricted to only HTML, it can be also used for defining styles for XHTML and XML markup.

Program 2. Example of a CSS file.

```
body {
  margin: 0;
  padding: 0;
  height: 100%;
}
#wrap {
  margin: 0 auto;
  border: 2px solid #000000;
  width: 1200px;
  min-height: 100%;
  height: 100%;
   position: relative;
}
#header a img {
  border: 0;
}
#headerlink {
  width: 1000px;
  height: 100px;
  min-height: 100%;
  background-repeat: no-repeat;
}
```

Like the name implies, the styles cascade. Meaning that same element can have multiple styles. Web designer can define general style sheet for the website, but can use another style sheet that will be used some specific page. The page specific styles will then override the styles defined in the general style sheet. [1, 2]

2.1.2 PHP

PHP (PHP: Hypertext Preprocessor) is a server-side scripting language, originally developed for producing dynamic web pages. Its syntax is mostly influenced by C, Java and Perl. Most commonly PHP is embedded into HTML. PHP has earned its number one spot in web development scripting languages by having built-in functions for database manipulation, most notably MySQL. In order to use PHP in web development, the web server has to have PHP installed. PHP documents end with ".php" extension by default, if web server encounters one it automatically passes it to PHP processor. Web servers can be configured to also pass other files to PHP processor if needed. PHP processor will process all the code inside PHP-tags. The tag used for opening a PHP code segment is “<?php” and it will be closed with “?>”. Everything outside those tags will be processed as normal HTML.

Program 3. PHP code embedded into a HTML file.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
      <title>Welcome</title>
      <link href="style.css" rel="stylesheet" type="text/css" />
   </head>
   <body>
      <?php
         $firstname = "John";
         $lastname = "Smith";
         echo ("Hi! Welcome to my website $firstname $lastname.");
      ?>
   </body>
</html>
```

Although PHP is embedded into HTML, it is very common to put PHP code in separate files and then include it into HTML. PHP supports object-oriented programming, therefore medium to large scale websites mainly have files that consist of only PHP code. [3, 4]

2.1.3 JavaScript

JavaScript is a client-side scripting language. In web development, it can be used to alter web page contents without reloading the page. JavaScript can be used for variety of things, for example showing hidden elements when user clicks a button or detecting the browser user is using and loading a page specifically designed for that browser. JavaScript code is also embedded into a HTML document, and will be executed if it’s inside “<script type=”text/javascript”></script>” tags.

Program 4. JavaScript code embedded into a HTML file.

```
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <title>Welcome</title>
  <link href="style.css" rel="stylesheet" type="text/css" />
</head>
<body>
  <script type="text/javascript">
    document.write("Hi! Today is " + Date());
  </script>
</body>
```

JavaScript has many third-party libraries that are designed to make JavaScript writing more easy and flexible, the most used one is jQuery. jQuery gives web developers means to write complex JavaScript functions like Ajax calls with ease. Ajax will be explained in chapter 2.1.5 of this document. [4, 5]

2.1.4 MySQL

MySQL is the most used relational database management system in the world. Key factors to its success are the fact that it is free to use, and it is exceptionally fast and powerful. It is very lightweight, and runs on over 20 operating systems. Due to these features, MySQL is used even on the largest volume websites including Facebook, Google, Wikipedia and many others. MySQL database consists of tables, each table contains rows. Each row contains columns in which the data is stored.

Table 1. Example of a MySQL table, containing user information.

id	username	joined	password
3	john_33	2012-02-15	11c931a873eb
4	northlite	2012-05-09	88c89be093d4
5	maroon	2012-06-05	ea1fd4d63f2b5

SQL in MySQL stands for Structured Query Language, it is loosely based on English and is also used in other databases like Oracle and Microsoft SQL Server. Query “SELECT username FROM users WHERE id = 5” will return “maroon”. [4, 6]

2.1.5 Ajax

Ajax (Asynchronous JavaScript And XML) is not literally a web technology itself, but a collection of different methods to transfer data between a web page and a server without reloading the page. It reduces the stress of a web server and makes website user experience substantially better. It is very widely used in modern web development, for example in Google Maps when loading new sections of map when changing location or for loading new status updates on the main page of Facebook. Main component of Ajax is a JavaScript object XMLHttpRequest that is made for asynchronous communication. Other vital part is data transferring, originally handled by XML but now more commonly by JSON. DOM (Document Object Model) is used for data interaction, HTML and CSS for displaying it and JavaScript to make it all work. Ajax has been a big factor in transforming web applications indistinguishable from actual standalone applications. [4, 7]

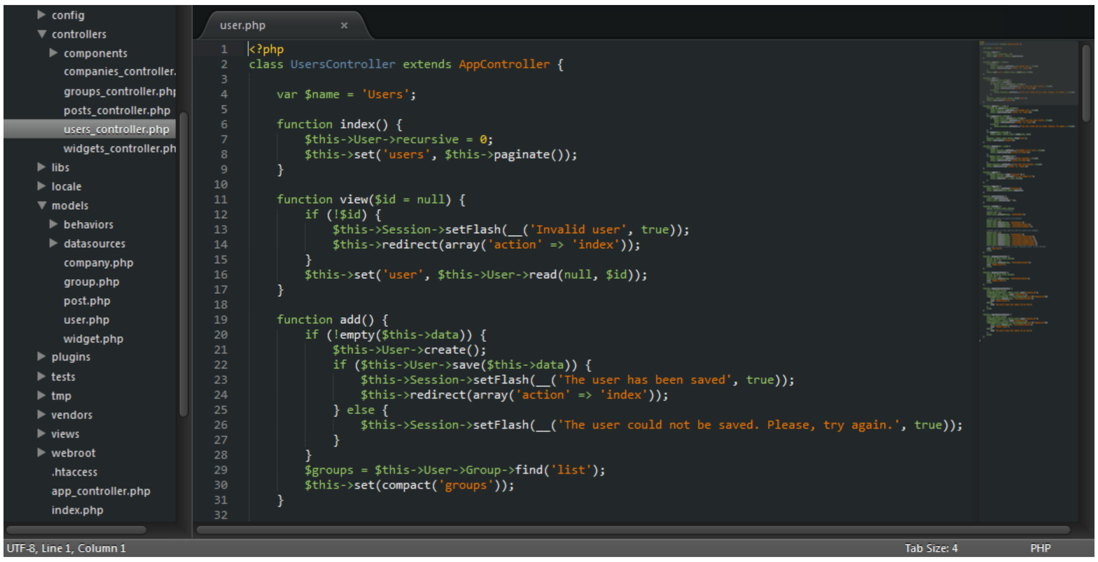
2.2 Tools

2.2.1 XAMPP

All dynamic websites need a web server. Servers are basically computers which are functioning around the clock, and primarily storing and processing website data. When developing web applications, it is possible to use a dedicated web server or run a virtual web server on the computer the development is done. [8] In this project, development is done on a local virtual server environment called XAMPP(Cross-platform, Apache HTTP Server, MySQL, PHP, Perl). It is a package that contains all the essential web development technologies. The main component is the server software Apache HTTP Server, it is the most used web server software available with over 60% market share. Other main components are MySQL, PHP and Perl. The package also includes other components, the one that is used in this project is phpMyAdmin. It is a web application that is used for manipulating MySQL data. XAMPP is much easier to set up than installing the Apache server and other programs separately, but it is developed to be used only as a local developing server and doesn’t have very strong security. The final product of this project will be running on a dedicated Apache based server. [4, 9]

2.2.2 Sublime Text 2

Programming in web development can be entirely done with the most basic text editors like Notepad or TextEdit. But the development can be significantly improved by using a text editor designed for programming. These editors have functions to color the syntax, highlight errors, automatically complete the words and make code look better by indenting and spacing it. [4] The text editor used in this project is Sublime Text 2. It is very lightweight, has clean user interface, is very customizable and offers a lot of add-on features thanks to a large user base.



```php
<?php
class UsersController extends AppController {

    var $name = 'Users';

    function index() {
        $this->User->recursive = 0;
        $this->set('users', $this->paginate());
    }

    function view($id = null) {
        if (!$id) {
            $this->Session->setFlash(__('Invalid user', true));
            $this->redirect(array('action' => 'index'));
        }
        $this->set('user', $this->User->read(null, $id));
    }

    function add() {
        if (!empty($this->data)) {
            $this->User->create();
            if ($this->User->save($this->data)) {
                $this->Session->setFlash(__('The user has been saved', true));
                $this->redirect(array('action' => 'index'));
            } else {
                $this->Session->setFlash(__('The user could not be saved. Please, try again.', true));
            }
        }
        $groups = $this->User->Group->find('list');
        $this->set(compact('groups'));
    }

```

Picture 1. Example of a Sublime Text 2 interface.

It is cross platform and really easy to transfer from a computer to another. All the user specific configurations like color themes and file history are in a separate folder, so the users can continue working on a different computer just by copying that folder. [10]

2.2.3 Git

Git is a free open source Version Control System (VCS) developed by Linus Torvalds in 2005. VCS:s are used to keep track of the changes in program development. Git has the ability among many to revert back the changes in code, merge together code that has been simultaneously edited by multiple users, create separate development branches. If developer notices some issues with the program under development, with

Git he can track the source of the problem by reverting back to older versions and after pinpointing the problem go back to the newest version. Developers can also create branches, data edited in a separate branch doesn’t affect the data in the main branch. When the functions in the new branch is completed, it is possible to merge that branch to the main branch. That way developers can experiment on different things without worrying about messing up the program their developing. [11]

3 CAKEPHP

3.1 General info

CakePHP is a free open-source PHP framework. Its main objective is to give web developers ways to rapidly create robust web applications. It has an active development team to ensure the security and keep the framework bug free. CakePHP has lots of features to make web development more straightforward, for example: MVC architecture, ACL, integrated tools for database interaction, built-in validation, HTML and JavaScript helpers, localization. The most important functions will be covered later in this document. [12, 13]

3.2 PHP Framework

Web developers usually have specific ways of building web applications, the ways are usually learned through experience in different projects. When developing applications, developers tend to reuse parts of the code they have used earlier or build applications entirely on top of the old projects and tweak the code to work with new one. If new application is considerably different from the old one, code and the structure of the application may become complex and unreadable. PHP frameworks are designed to help in these situations. The main idea of frameworks is to give developers regularly used functionalities and core to build their applications on. The frameworks are a well-tested collection of code, libraries and conventions. They are flexible to support wide array of application designs, simple and complex. Frameworks aren’t usually made by one person, but a team of experienced developers. That means the code and conventions used in the framework are usually picked from a selection of different ones, usually the most dynamic or flexible. Frameworks are also constantly tested for security issues. It is easy for new developers to get productive in projects if they already know the conventions of the framework, it takes lot more time to use custom framework and teach each newcomer separately. It’s safe to say that developers both beginning and experienced are better off using robust and tested framework, than to build or modify their own. [13]

3.3 MVC Architecture

The MVC (Model View Controller) is the design pattern CakePHP uses. It is widely used in all kind of software development, but is most used in web development. As the name implies, MVC architecture separates code into three parts: models, views and controllers. In this document these parts the way used in CakePHP. [13]

3.3.1 Models

Models represent the data parts of the application, more precisely the database tables. In CakePHP, each database table has its own model. PHP code for retrieving, editing and deleting the data is located in models. Relationships between models are defined in model, and models also handle the data validation. The applications core business logic is defined in models. [12, 13]

3.3.2 Controllers

Controllers control the logic of the application. All the methods and functions are located in Controllers. Controllers act in the middle of the user interface and database. Controllers usually manage the user authentication, decide what kind of data is given to models and what is shown to users. [13]

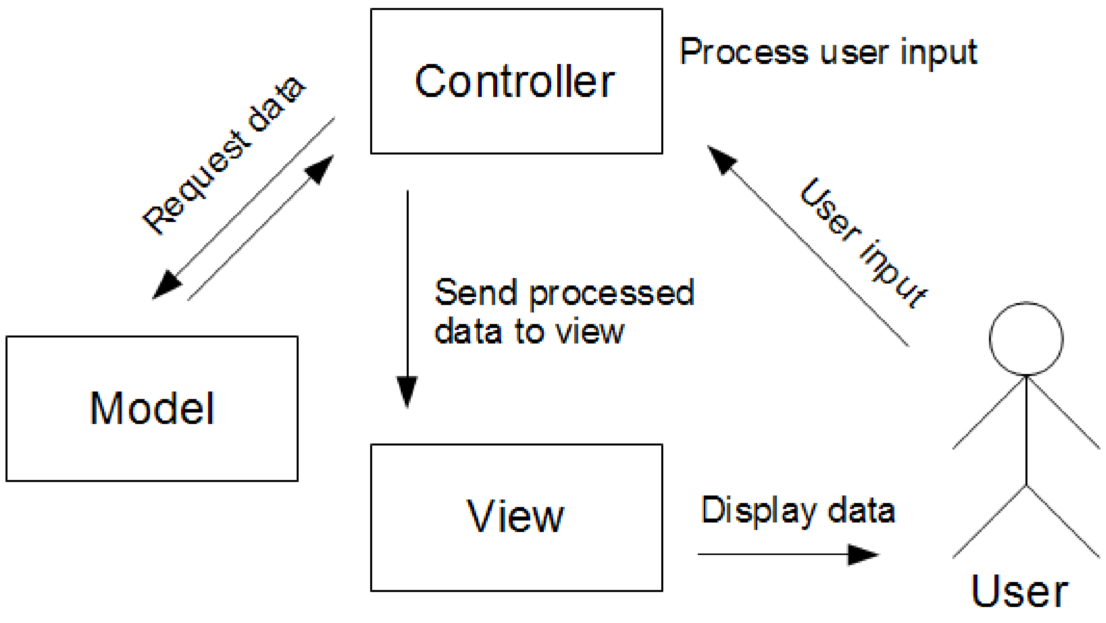
3.3.3 Views

Views are the presentation parts of the application. View is usually a HTML file that shows the static data and/or the data given by a Controller. Views are not restricted to only be HTML pages, they can also be PDF files, videos, and many others. [12 ,13]

3.3.4 MVC cycle

CakePHP requests generally follow this basic pattern: user requests a page or some other part of the application, the request is then dispatched to the correct Controller.

Controller will communicate with model to receive, edit or in some other way manage the data. Controller will then select the view object and pass the data into it, view will be then outputted to the user. [12]



Picture 2. Typical CakePHP MVC request cycle.

Here is a demonstration of one possible MVC cycle. User clicks a button on a webpage that is used for showing the user's personal information, the Controller checks if the user is logged in and if he has the permissions to access that data. Controller then requests the data from the model and passes the data to the view, view gets the data and shows parts of the data to the user. [12, 13]

3.3.5 Benefits

MVC architecture helps to arrange the code into logical segments. It will help developers to easily navigate the code and find specific functions more quickly. It also makes code more modular, it is easier to add new functionalities and reuse the code. MVC architecture also makes it possible for back-end and front-end developers to work more efficiently. Developers can make new functions to the Controllers without worrying how will the outcome look, and the designers can work on view files without concentrating on the code. [12, 13]

3.4 Access Control Lists

Websites that feature database manipulation often need to have a system to manage user permissions. The most basic way is to make the database management only available for authenticated site administrators. It is often done by having a login page where admins enter their username and password, if authentication is successful they will get access to features needed for altering the database. This is usually good enough if the website has only few administrators. User management will come into play when the website becomes more complex and needs more administrators or needs to have other types of registered users. The website might have multiple different types of users, regular users that should only be able to edit their own data, user managers that can edit the data of all the regular users, and site administrators that have full access to everything. One way is to separately implement user level checking on every function of the website. It would work, but it will need lot of work every time new functions are added or new user types introduced. ACL is designed to simplify the user rights management. ACL independently manages what has access to where. [14]

3.4.1 Access Control Lists in CakePHP

In CakePHP, ACL is divided into two object groups: ACOs (Access Control Objects) and AROs (Access Request Objects). ACOs are something that access is wanted to, often Controllers or certain functions in the Controllers. In MVC framework, all the connections to the database goes through Controllers, therefore Controllers are used as ACOs. AROs are something that wants to access something, usually users or user groups. Connections between AROs and ACOs are defined in the database. The database features three tables: “aros”, “acos” and “aros_acos”. As the names imply, “aros” and “acos” tables contain all the AROs and ACOs featured in the website. The values are unique identification numbers (id:s). The primary key of the user or user group will be used as ARO value. Also Controllers and their functions will be given unique ACO values. These values will be linked together in the “aros_acos” table. [14]

3.4.1.1 CakePHP AROs table

As explained earlier, Access Request Objects are something that want to access something. In CakePHP, AROs are generally users or groups of users. In *aros* data-base table the AROs are nested in a tree structure.

Table 2. A basic CakePHP aros database table that features three groups, each group has one user.

id	parent_id	model	foreign_key	alias	lft	rght
1	NULL	Group	1	Admins	1	4
2	NULL	Group	2	Managers	5	8
3	NULL	Group	3	Users	9	14
4	1	User	1	Admin1	2	3
5	2	User	2	Manager1	6	7
6	3	User	3	User1	10	11

Users have a “parent_id” value for identifying which group they belong to. Users and Groups also have their own tables that include more specific information about them, “foreign_key” is used for referencing IDs of those tables. The tree structure is defined by “lft” and “rght” values. [14, 15]

3.4.1.2 CakePHP ACOs table

Access Control Objects are something that are wanted to access by something. In CakePHP those are generally Controllers or specific functions in a Controller.

Table 3. A CakePHP acos table that includes ACOs related to “Posts” Controller.

id	parent_id	alias	lft	rght
1	NULL	controllers	1	16
2	1	Posts	2	15
3	2	add	3	4
4	2	edit	5	6

5	2	index	7	8
6	2	view	9	10
7	2	delete	11	12

The ACOs table generally lists all the Controllers and functions available, functions' "parent_id" references the "id" of its Controller. "lft" and "rght" are again used for tree structuring. [14, 15]

3.4.1.3 CakePHP join table for AROs and ACOs

The *aros_acos* table is used for defining the actual permissions. There are different approaches for doing it, the most common one is to first deny all the permissions for everyone and then gradually grant those when needed.

Table 4. A CakePHP aros_acos table for granting and denying permissions. In this table the Admins group is granted a full access to all the Controllers. Managers and Users are denied the access to all Controllers, but Managers get full access to the Posts Controller and Users get access to Posts Controller functions "view" and "index".

id	aro_id	aco_id	_create	_read	_update	_delete
1	1	1	1	1	1	1
2	2	1	-1	-1	-1	-1
3	2	2	1	1	1	1
4	3	1	-1	-1	-1	-1
5	3	5	1	1	1	1
6	3	6	1	1	1	1

The way to grant or deny permissions, is to create a row in the table that has the id of an ARO and the id of an ACO, followed by a CRUD (Create, Read, Update, Delete) listing that is filled by "1" if access is allowed or "-1" if access is denied. [14, 15]

3.4.2 Using Access Control Lists in CakePHP

It would be too cumbersome to do the permission editing in a database. CakePHP has built-in functions for granting and denying permissions. First the ACL structure needs to be set up as defined in CakePHP Cookbook. After that, the permission handling could be done with just a few lines of code.

Program 5. The CakePHP ACL functions to achieve the same permissions as defined in chapter 3.4.1.3.

```
$group =& $this->User->Group;

$group->id = 1;
$this->Acl->allow($group, 'controllers');

$group->id = 2;
$this->Acl->deny($group, 'controllers');
$this->Acl->allow($group, 'controllers/Posts');

$group->id = 3;
$this->Acl->deny($group, 'controllers');
$this->Acl->allow($group, 'controllers/Posts/index');
$this->Acl->allow($group, 'controllers/Posts/view');
```

Those functions can be used in many ways. First it is possible to add all those lines in a separate function that will create all the needed database entries when executed. The permissions will then be set. After that the ACL-functions could be tied to certain functions in Controllers, for example there could be a permissions handling page on the website where admins could change permissions for individual groups. [15]

3.5 Other Features

The CakePHP has wide array of features in addition to ACL that will speed up web application development. This document will cover the basics of those without going too much into detail.

3.5.1 Authentication

The AuthComponent is used for authenticating user that try to access the website. The component basically checks if the user exists, and if the users' username and password match. More advanced websites use AuthComponent combined with the ACL Component to handle more complex levels of user access. [16]

3.5.2 Email

Adding simple email sending functionalities to web applications is done using CakePHPs emailComponent. It can be configured to send mails through PHP:s own email functions or smtp. It also supports file attachments and features other email related checking and filtering. [16]

3.5.3 Request Handling

The Request Handler component is used for obtaining information about HTTP-requests made in the application. It can be used to check the request types and changing the site behavior to match those. For example, it can be used for detecting if the requests are AJAX and if they are done using mobile browser. [16]

3.5.4 Security Component

The component is designed for adding extra security to the website. Some of the features are attaching hidden randomly generated values to all the HTTP-requests or use timeouts on form submissions. [16]

3.5.5 Sessions

The CakePHP sessions component features methods to interact with PHP sessions. It is used for reading, adding or otherwise managing the session values. [16]

3.5.6 Translate

As the name implies the TranslateBehavior is CakePHPs’ way to handle the application localization. [17]

3.5.7 Tree

A way to store hierarchical data in a database. CakePHP uses MPTT (Modified Preorder Tree Traversal) logic for tree traversal, this logic is used in a table at chapter 3.4.1.1 of this document. [17]

3.5.8 Helpers

CakePHP features lot of built-in helpers to speed up the application development. CakePHP helpers are: Ajax, Cache, Form, HTML, Js, JavaScript, Number, Paginator, RSS, Session, Text, Time and XML. For example the PaginatorHelper takes off the burden of creating pagination, developers can define the pagination parameters like the amount of results per page and how will those be ordered in a Controller that uses the pagination. Then the pagination function will be called in the View with just a few lines and the pagination is done. Other really useful helper is the FormHelper, it will do all the form validation, form population and layout automatically. It is many ways better to use CakePHP helpers instead of writing own PHP or HTML code, whenever possible. [18]

3.5.9 Bake

Bake is an automatic code generator. The CakePHP Bake console can be used for automatically generate core architecture of the web application. First the database needs to have desired tables, then the Bake can be used to generate all the needed Models, Controllers and Views. The Bake console is run using PHP Command Line Interface. [19]

4 PROJECT

The main goal of this project was to create a user management system using CakePHP Access Control Lists and implement it into Hintajahti, an online price comparison service by Data Media Gazelle. Due to the Data Media Gazelles' request, the projects actual structure and code will not be featured in this document. However, before implementing the ACL and other user management functions into the main project, they were tested out in a smaller scale but similar test application. The test application will be covered here.

4.1 Hintajahti

Hintajahti is an upcoming online price/product comparison service. Its main feature is listing of products that are sold by third party companies. Hintajahti's users can select multiple products for comparison and find out the location and prices of the products. The site will have many different user levels, starting with regular users that can only compare the products, up to admins that have full access to everything.

4.2 The requirement specification

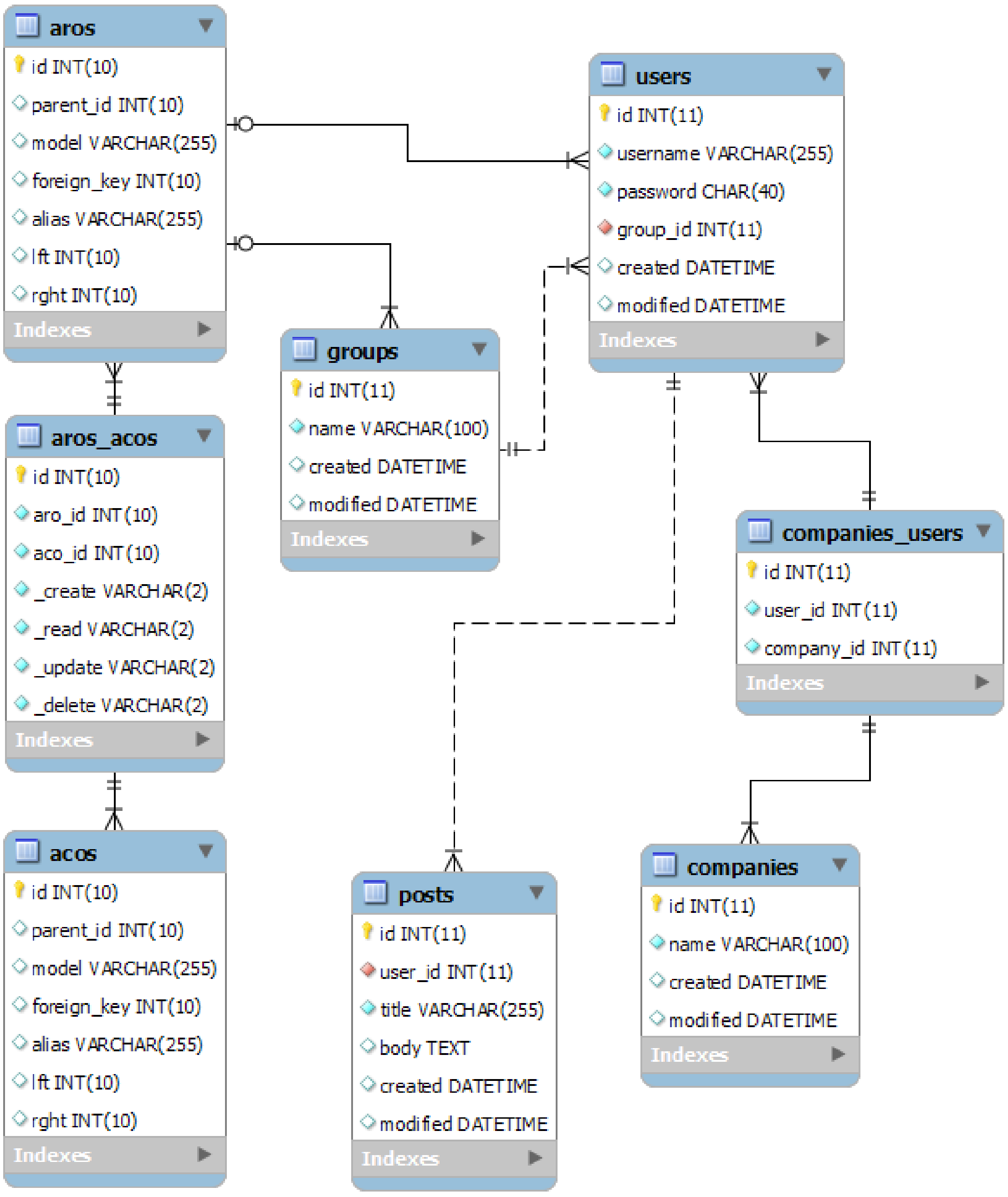
The main objective of the project was to create an ACL system that supports multiple groups and enables users to belong to many groups. Secondary objective was to create user interface for the user management.

4.3 Test application

Due to the security reasons and the possibility of the main project Hintajahti to store personal information of the users and the companies, only the test application will be covered here. The test application was designed to have similar database and file structure as the main project but to only include the minimum amount of data to keep it easy to manage and debug.

4.3.1 Database

The main project Hintajahti uses quite complex database to support all the features and still stay dynamic and stable. The test applications' database was kept to bare minimum, just to test out the functionality of the user management.



Picture 3. Database structure of the test application.

The main components in the database are *users, groups, companies* and *posts.* Relationships between *users* and *companies* is managed by *companies_users* join table. The user access is checked by ACL tables *aros*, *acos* and join table *aros_acos* as explained earlier. *Users* always belong to *groups* that are connected to *aros* table. *Users* are also connected to *aros* table, in case they need to have user specific permissions. The *posts* table was used just to check the functionality of the ACL. In CakePHP, relationships are not defined in the database itself, instead they are defined in Models.

Program 6. Company model database definitions.

```
class Company extends AppModel {
   var $name = 'Company';

   var $validate = array(
      'name' => array(
         'notempty' => array(
            'rule' => array('notempty'),
         ),
      ),
   );

  var $hasAndBelongsToMany = array(
    'User' =>
      array(
        'className'              => 'User',
        'joinTable'              => 'companies_users',
        'foreignKey'             => 'company_id',
        'associationForeignKey'  => 'user_id',
        'unique'                 => true,
      )
    );
}
```

CakePHP generally requires every database table to have a Model, except join tables. Join tables are defined in a Model of first element in join table, as *companies_users* is defined in a Company Model. In this case users can belong to multiple companies, and companies can belong to many users, therefore CakePHP variable $hasAndBelongsToMany is used.

4.3.2 Project structure

The basic core of the project was done by following CakePHP tutorial “Simple ACL controlled application” that is presented in a CakePHP Cookbook, found on the official CakePHP website.

The project had three Controllers: companies, users, groups and posts. The contents of the Controllers were pretty much the same as in the tutorial, except the users Controller had the additional ACL related functions. User specific rights handling functions were tested by allowing and denying access to “Posts” controller. The functions also check if the user being managed belongs to the same “Company” as the manager.

Program 7. User specific ACL definitions in Users Controller.

```
function allowThisUserPosts() {
     $user =& $this->User;
     $loggedUserCompanyId = $this->Auth->user('company_id');
     $userCompanyId = $user->read('company_id');
     if($loggedUserCompanyId == $userCompanyId['User']['company_id']){
        $this->Acl->allow($user, 'controllers/Posts');
        echo "Rights given";
     }else{
        echo "You don't have the rights to do this";
     }
       exit;
}

function denyThisUserPosts() {
     $user =& $this->User;
     $loggedUserCompanyId = $this->Auth->user('company_id');
     $userCompanyId = $user->read('company_id');
     if($loggedUserCompanyId == $userCompanyId['User']['company_id']){
        $this->Acl->deny($user, 'controllers/Posts');
        echo "Rights taken";
     }else{
        echo "You don't have the rights to do this";
     }
       exit;
}
```

Every Controller also had a View. For this test application, the CakePHP default Views were used. The Hintajahti uses custom Views, but as the MVC architecture is designed for modular use, the Controller functions of the test application would also work with the Hintajahti's views.

Program 8. Sample of a Users/index view.

```
<div class="users index">
  <h2><?php __('Users');?></h2>
  <table cellpadding="0" cellspacing="0">
  <tr>
      <th><?php echo $this->Paginator->sort('username');?></th>
      <th><?php echo $this->Paginator->sort('group_id');?></th>
      <th><?php echo $this->Paginator->sort('company_id');?></th>
      <th class="actions"><?php __('Actions');?></th>
  </tr>
  <?php
    $i = 0;
    foreach ($users as $user):
      $class = null;
      if ($i++ % 2 == 0) {
        $class = ' class="altrow"';
      }
  ?>
  <tr<?php echo $class;?>>
    <td><?php echo $user['User']['username']; ?>&nbsp;</td>
    <td>
      <?php echo $this->Html->link($user['Group']['name'], ar-
ray('controller' => 'groups', 'action' => 'view', $us-
er['Group']['id'])); ?>
    </td>
```

In the above sample of the Users/index view, HTML and PHP are used to create list of the users that are registered. CakePHP Paginate- and HTML-helpers are used to create links and pagination/sorting.

4.3.3 Issues

The main issue was the CakePHPs ACL to only support one "Role" per "User". One objective of the project was to find out if one user could belong to multiple groups. The idea was to create separate groups for product managers, user managers, sales managers etc., each of the group would have different rights. The need for the multiple groups per user was to enable users to be for example product- and sales managers at the same time. After a thorough investigation, it was discovered that it is not possible to do it in the CakePHP version used in project. The possibility to give user specific rights to each of the users was used to circumvent this project. For example user that belongs to product managers, could be given rights to access controllers related to sales managers.

Other issue was the users also belonging to companies. As the users already belong to one ACL “Role” called groups, the possibility of using company as a “Role” was out of question. Instead manual checking functions were created to check users company.

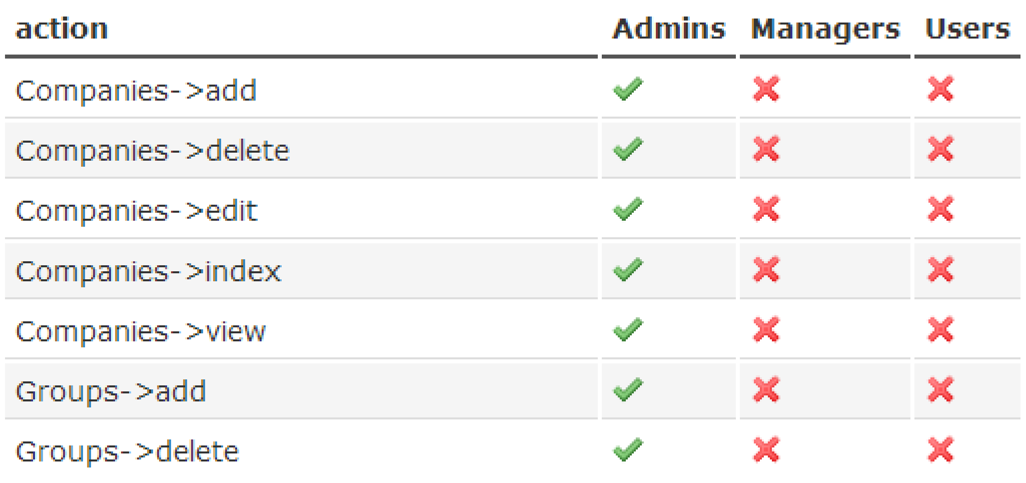
Program 9. Company editing function that also checks if the user that is editing belongs to the company.

```
function edit($id = null) {
     //check if logged user has the same company_id as the company that
is being edited
     $companyId = $this->Company->field('id');

     $userCompanyArray = $this->Company->CompaniesUser-
>findByUserId($this->Auth->user('id'));
     $userCompanyId = $userCompanyArray['CompaniesUser']['company_id'];

     if ($userCompanyId == $companyId){
     //check end
        if (!$id && empty($this->data)) {
           $this->Session->setFlash(__('Invalid Company', true));
           $this->redirect(array('action' => 'index'));
        }
        if (!empty($this->data)) {
           if ($this->Company->save($this->data)) {
              $this->Session->setFlash(__('The Company has been saved',
true));
              $this->redirect(array('action' => 'index'));
           } else {
              $this->Session->setFlash(__('The Company could not be
saved. Please, try again.', true));
           }
        }
     if (empty($this->data)) {
           $this->data = $this->Company->read(null, $id);
        }
     }else{
        $this->Session->setFlash(__('You dont belong to this company',
true));
        $this->redirect(array('action' => 'index'));
     }}
```

Each function that is related to editing something that belongs to some company needs to check if the user belongs to that company. The functions could be contained into separate method that would be easier to use in development, but this was enough to prove the concept.

4.3.4 Alaxos ACL plugin

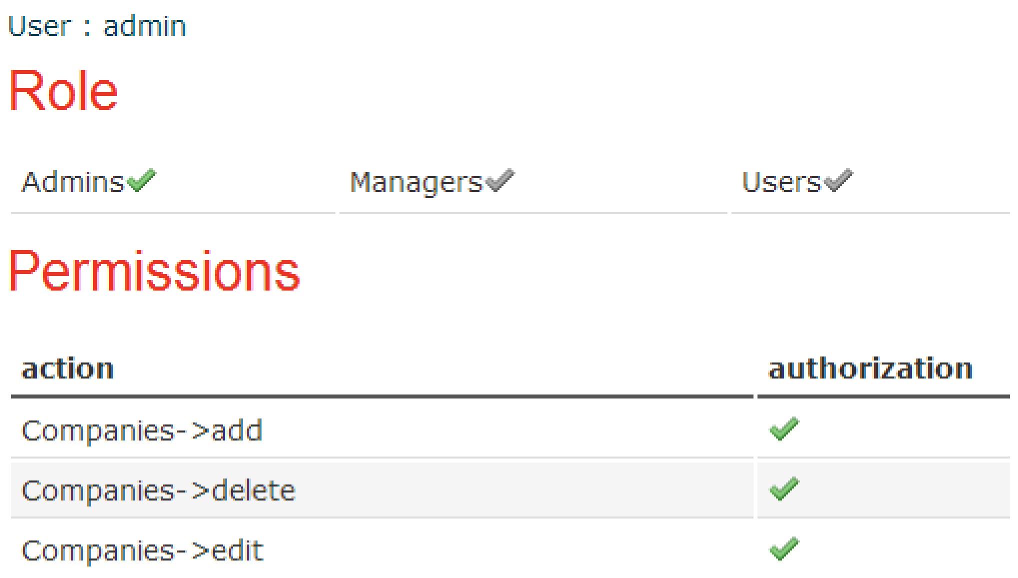
CakePHP doesn’t include any interface to manage ACL relations. It is possible to do all the ACL managing with separate functions like demonstrated in “Program 7”. But during the initial project research, third party plugin was discovered that was designed for this purpose. The Alaxos ACL plugin basically uses the same ACL functions that are described in CakePHP tutorials and documentation, but ties them all together into one simple to use interface. The plugin allows to automatically create new Aros and Acos when needed, it visualizes the Aros and Acos relations and enables to change the permissions. All the permission handling requests are transferred via AJAX –calls, so the managing is fluid and effective.



action	Admins	Managers	Users
Companies->add	✓	✗	✗
Companies->delete	✓	✗	✗
Companies->edit	✓	✗	✗
Companies->index	✓	✗	✗
Companies->view	✓	✗	✗
Groups->add	✓	✗	✗
Groups->delete	✓	✗	✗

Picture 4. Group rights management interface of Alaxos ACL plugin.

The plugin interface lists all the Acos (Controllers and their functions) on the left and the Aros (Groups) on the right, the rights are given and taken by clicking the corresponding icons.


User : admin

Role

Admins✓	Managers✓	Users✓

Permissions

action	authorization
Companies->add	✓
Companies->delete	✓
Companies->edit	✓

Picture 5. User specific rights management interface of the Alaxos ACL plugin.

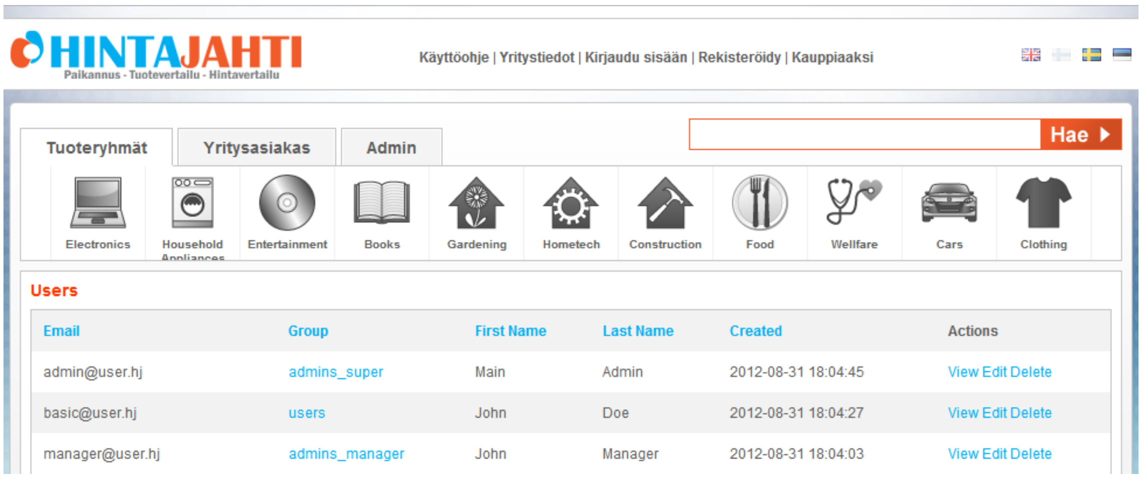
User management works the same way, except the changes only affect one user. It's also possible to change users "Role" from the same page.

4.4 Test application summary

Test application was successful, it had an easy to use user interface for user rights managing for admins. It wasn't possible to make CakePHP ACL to accept multiple "Roles" per "User" but it was possible to give user specific rights to users. It was also possible to create custom functions that enable company managers to give rights to users that belong to the same company.

4.5 Implementing the test application logic into Hintajahti

The CakePHP has a strict naming policy and conventions that has to be followed. Hintajahti and the test application were both coded maintaining the CakePHP policies and conventions. That made integrating the test application code into Hintajahti quite straightforward operation, some configuration related issues were faced but nothing serious.



Users

Email	Group	First Name	Last Name	Created	Actions
admin@user.hj	admins_super	Main	Admin	2012-08-31 18:04:45	View Edit Delete
basic@user.hj	users	John	Doe	2012-08-31 18:04:27	View Edit Delete
manager@user.hj	admins_manager	John	Manager	2012-08-31 18:04:03	View Edit Delete

Picture 6. Sample of Hintajahti’s user listing page.

The user management user interface was modified to match Hintajahti’s visual style. The modular nature of the CakePHP and the MVC framework enables the work made in the sample project to be used in other projects than Hintajahti as well.

5 SUMMARY

The goal of the project was to create a user management system for the online price/specification comparison website Hintajahti. The final product wasn't exactly the way it was supposed to be, due to some framework related issues. The project was still a success, in the end all the needed requirements were fulfilled.

The CakePHP framework and especially the Access Control Lists needed some time to get used to. But after becoming familiar with them, the development was straightforward and rewarding. CakePHP certainly is a powerful tool for web developers that want to get the job done and not reinvent the wheel for every new project. The thing that impressed the most was the modularity of the framework. It was possible to create a sample application that included only the minimal amount of functionalities and move the functions and plugins to the main project without much re-coding.

CakePHP documentation wasn't always the best place to learn the framework. More often than not the useful information was found from other users' websites rather than the official CakePHP documentation. Especially the Access Control Lists were documented quite poorly in the official documentation. Access Control Lists were the one single most demanding part of the project, and because of the poor documentation most of the learning was done through trial and error.

CakePHP definitely should be considered using on all medium to large scale web application. It enables web developers to concentrate on the development and makes the applications dynamic and modular.

REFERENCES

[1] “HTML & CSS – W3C”, [online], Saatavilla: http://www.w3.org/standards/webdesign/htmlcss
(Luettu: 3.8.2012)

[2] “What is CSS – What are Cascading Style Sheets?”, [online], Saatavilla:
http://webdesign.about.com/od/beginningcss/a/aa021607.htm
(Luettu: 5.8.2012)

[3] “PHP: General Information - Manual”, [online], Saatavilla:
http://fi2.php.net/manual/en/faq.general.php
(Luettu: 5.8.2012)

[4] Nixon R, P. 2009. Learning PHP, MySQL, and JavaScript. Sebastopol:

O’Reilly Media, Inc.

[5] “JavaScript Introduction”, [online], Saatavilla: http://www.w3schools.com/js/js_intro.asp
(Luettu: 5.8.2012)

[6] “MySQL :: Why MySQL?”, [online], Saatavilla: http://www.mysql.com/why-mysql/
(Luettu: 6.8.2012)

[7] “Ajax: A New Approach to Web Applications”, [online], Saatavilla:
http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications
(Luettu: 7.8.2012)

[8] “What is web server?”, [online], Saatavilla:
http://www.webdevelopersnotes.com/basics/what_is_web_server.php
(Luettu: 10.8.2012)

[9] Dalibor D, *Installing, configuring, and developing with XAMPP* [online], Saatavilla:
http://dalibor.dvorski.net/downloads/docs/InstallingConfiguringDevelopingWithXAMPP.pdf
(Luettu: 10.8.2012)

[10] “Sublime Text 2 Documentation”, [online], Saatavilla:
http://docs.sublimetext.info/en/latest/index.html
(Luettu: 10.8.2012)

[11] “Git Book”, [online], Saatavilla: http://git-scm.com/book
(Luettu: 10.8.2012)

[12] “CakePHP Cookbook”, [online], Saatavilla: http://book.cakephp.org/2.0/en/index.html
(Luettu: 12.8.2012)

[13] Anupom S, Ahsanul B, P. 2008. CakePHP Application Development. Birmingham:

Packt Publishing Ltd.

[14] “Understanding how ACL works”, [online], Saatavilla:
http://book.cakephp.org/1.3/en/view/1243/Understanding-How-ACL-Works
(Luettu: 14.8.2012)

[15] “Auth and ACL, an end to end tutorial”, [online], Saatavilla: http://mark-story.com/posts/view/auth-and-acl-an-end-to-end-tutorial-pt-1
(Luettu: 17.8.2012)

[16] “CakePHP Core Components”, [online], Saatavilla:
http://book.cakephp.org/1.3/en/view/1241/Core-Components
(Luettu: 17.8.2012)

[17] “CakePHP Core Behaviors”, [online], Saatavilla:
http://book.cakephp.org/1.3/en/view/1319/Core-Behaviors
(Luettu: 17.8.2012)

[18] “CakePHP Core Helpers”, [online], Saatavilla:
http://book.cakephp.org/1.3/en/view/1357/Core-Helpers
(Luettu: 17.8.2012)

[19] “CakePHP Code Generation with Bake”, [online], Saatavilla:
http://book.cakephp.org/1.3/en/view/1522/Code-Generation-with-Bake
(Luettu: 17.8.2012):