# BUILDING A WEB CONTENT MANAGEMENT SYSTEM

A Thesis

Presented to the

Faculty of

San Diego State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

by

Rima Vyomeshbhai Shah

Fall 2012

**SAN DIEGO STATE UNIVERSITY**

The Undersigned Faculty Committee Approves the

Thesis of Rima Vyomeshbhai Shah:

Building a Web Content Management System

Alan Riggins, Chair
Department of Computer Science

Joseph Lewis
Department of Computer Science

Michael E. O'Sullivan
Department of Mathematics and Statistics

August 1, 2012
Approval Date

# DEDICATION

I dedicate this thesis to my parents and family who have been a great source of inspiration in my life. I would also like to dedicate this thesis to my thesis chair, Professor Alan Riggins and my graduate advisor Prof. Joseph Lewis who have provided me full support and guidance throughout my journey in pursuing graduate studies.

# ABSTRACT OF THE THESIS

Building a Web Content Management System
by
Rima Vyomeshbhai Shah
Master of Science in Computer Science
San Diego State University, 2012

This thesis investigates different existing open-source Web Content Management Systems (WCMS) and their features. The main objective of WCMS is to allow non-technical users to easily update and manage a website's content. But WCMS products available on the market today have integrated many other complex functionalities, which makes the user interface difficult to understand and use. All WCMS require some amount of training on installation and where to start. After integration users need to read the manual or take lessons to use other available advanced features. The research question explored here is whether to implement a WCMS from existing options compared with the feasibility of building a new one from scratch. This paper discusses the considerations of developing a new WCMS, including the advantages and disadvantages relative to an off-the-shelf product.

# TABLE OF CONTENTS

PAGE

APPENDIX

# LIST OF TABLES

PAGE

# LIST OF FIGURES

PAGE

# ACKNOWLEDGEMENTS

I would sincerely like to thank my project advisor and thesis chair Prof. Alan Riggins for his continuous guidance and motivation during my thesis research. I would also like to thank my other committee members Prof. Joseph Lewis and Prof. Michael E. O’Sullivan for being in my committee and encouraging my work.

# CHAPTER 1

# INTRODUCTION

Content management (CM) is the set of processes and technologies that support the collection, managing, and publishing of information in any form or medium. Content may take the form of text (such as documents), multimedia files (such as audio or video files), or any other file type that follows a content lifecycle requiring management. Content management is an inherently collaborative process. It often consists of the following basic roles and responsibilities:

- Creator: responsible for creating and editing content.
- Editor: responsible for modifying the style of delivery, including translation and localization.
- Publisher: responsible for releasing the content for use.
- Administrator: responsible for managing access permissions to folders and files, usually accomplished by assigning access rights to user groups or roles.
- Consumer, viewer, or guest: the person who reads or otherwise takes in content after it is published or shared [1].

## 1.1 WHAT IS A CONTENT MANAGEMENT SYSTEM (CMS)?

A content management system (CMS) is a computer system or an application that allows publishing, editing, or modification of content, as well as site maintenance, from a central page [2]. The content on the World Wide Web consists of HTML, XML, and other documents and media files. This content can be published manually by editing and organizing files on a file system exposed to the web through a web server, requiring much technical expertise and tedious work. Content management systems store the actual content which can be text or images in a database. The system then automatically pulls the content out and shows it on appropriate pages based on the rules. Content management systems were created to help people publish documents and media with less technical intervention and in a more consistent and automated fashion. Anything from a document management system, to a media asset management system, to a portal, to a blog system could be considered a Content Management System [3].

Content management system speeds up the content updating process and makes it easy for a non-technical user. This helps in keeping the content up-to-date and timely. They are frequently used for editing, controlling, versioning and publishing specific documentation. They are used for different types of applications to manage its content. Below are some of the CMS applications.

- Blogs: comments on a particular topic;
- News: reading newspapers online instead of reading the hard/physical copy;
- Wikis: these are open to public editing.

The core features of Content Management Systems vary widely from system to system; many simpler systems provide only a handful of features, while others, especially enterprise systems, are much more complex and powerful.

- Allow for a large number of people to share and contribute to stored data;
- Control access to data based on user role (i.e., define information users or user groups can view, edit, publish, etc.);
- Facilitate storage and retrieval of data;
- Control data validity;
- Simplify report writing;
- Define data as almost anything: documents, movies, texts, pictures, phone numbers, articles etc. [2].

## 1.2 Project Definition and Purpose

In this thesis, I have built a Simple CMS which is a web content management system that can be used to dynamically manage the content of a simple static HTML website. For.eg. the news section of the Computer Science Department needs to be updated very often and since it is static HTML page, a technical person having knowledge of HTML and CSS is required to update even a small portion of the section. The primary objective of this thesis is to demonstrate how Simple CMS engine could be integrated with the “News & Events” section page of the San Diego State University’s Computer Science department, so that the process of updating the content becomes faster and easier for the department. The main purpose of this project is to have a user-friendly content administration interface that includes most common CMS functions appropriate for small and simple websites, so that a novice user can manage the website content. A user having less coding knowledge can easily add, edit and format the website's content using the rich text editor integrated in the Simple CMS engine without having to deal with the HTML and CSS code.

## 1.3 CONTENT OVERVIEW

The remainder of the thesis is organized as follows. Chapter 2 describes different types of CMS, compares the most popular open source CMS and discusses their advantages and disadvantages. It also discusses the advantages and limitations of this project. Chapter 3 describes different tools and technologies used to build this project- Simple CMS. Chapter 4 describes the common CMS features implemented in this project. In addition to that, it also describes the high level architecture, database architecture and implementation of Simple CMS. Finally Chapter 5, a conclusion and description of future work in this project to help contribute more for websites requiring more advanced functions in addition to the common basic CMS features available in the project.

# CHAPTER 2

# BACKGROUND

## 2.1 A SHORT HISTORY OF CMS DEVELOPMENT

Claims about who wrote the first CMS are varied, and include Roxen (1994) and Blitzen (mid-1990s), Ingeniux (1999), and Vignette. The main features were a very structured development environment and one had to use tags and templates because there was no WYSIWYG (What You See Is What You Get) editor. If one could not use HTML, they probably were not going to edit their site. Most of those websites were written by web design agencies rather than by software companies, and every agency had their own web designers who could write HTML code.

The second stage of CMS development was led by software houses who took over the functionality and started to build today’s CMS. Key features which were slowly built in included WYSIWYG text editing, search, and the addition of features like survey tools and podcasts. An early stage leader RedDot, and others, led the growth of specialists such as DotNetNuke and Mambo (which later changed to Joomla and DotNetNuke), which are still used today. CMS became feature-rich and web agencies needed both the technical skills, as well as designers, to adapt each client site into the CMS frameworks [4].

## 2.2 WHO NEEDS A CONTENT MANAGEMENT SYSTEM?

Organizations need a content management system if they:

- Use the same content over and over again in one publication (e.g., a warning in an instruction manual may be used several times within that manual).
- Publish the content to more than one media channel (e.g., the content in a printed instruction manual that ships with the product may also be used in the online help information).
- Publish the content in multiple languages.

The types of publications that these organizations produce might include:

- Technical documentation (parts catalogs, software documentation, user's manuals)
- Reference materials (encyclopedias, dictionaries, standards guides)

- Testing and training materials (e-learning programs, testing booklets)
- Marketing and educational materials (packaging, promotional flyers and ads, brochures) [5].

## 2.3 Types of Content Management Systems

CMS come in all shapes and sizes, and can manage anything a team of individuals is working on. From managing simple static website content, to allowing collaborative documentation across the Internet (wiki), CMSs perform many functions. CMS packages can generally be classified into two categories: Enterprise CMS and Web CMS.

### 2.3.1 Enterprise CMS

Enterprise Content Management (ECM) is a formalized means of organizing and storing documents, and other content, that relate to the organization's processes. These high-powered software packages are usually comprehensive solutions, delivering effective content management for use on an enterprise, or corporate level [6].

They are designed to help a corporation become more efficient and cost-effective, and increase accuracy and functionality. It also helps decrease human error and customer response times. They can integrate corporate functions such as shipping and delivery systems, invoicing, employee and human resource issues, customer relations, and document management and sales systems. Enterprise CMS brings data management down to the user’s level, so many users can add their individual piece of information to a very large integrated system data. Software companies deploying these complex systems offer highly customized company-wide solutions, which means the software usually comes with a relatively high price tag [7].

This type of CMS is generally used by large e-Commerce organizations. E-commerce websites are those that have a shopping cart designed to sell items online. The examples of such CMS used for e-Commerce websites are OSCommerce, Magneto etc.

### 2.3.2 Web CMS

Today, we are all used to the idea that we can create our own documents and publish it on the web. But if we go back fifteen or more years, the only way one could create a website was by understanding 'HTML’. Products like Dreamweaver and Frontpage were not around; this meant that if one wanted to build a website then he/she needed someone with technical skills to write it. Also once they had written it, one still needed someone with technical skills to change it, as changes involved reading HTML code to determine where to add content [8].

A Web Content Management System (WCMS) is a software system that provides website authoring, collaboration, and administration tools designed to allow users to create and manage website content with ease. A presentation layer displays the content to the website visitors based on the set of templates [9]. WCMS were developed to resolve the issue of having highly experienced technical staff adding low-level content to a website. WCMS is a web application used to create, manage, store and deploy content on web pages [8]. It is mainly a website maintenance tool that allows non-technical users to make changes to the website, and therefore the website's authoring and administrating tools. A user with little knowledge of programming languages can easily create and maintain the site's content. Web content types can include text, graphics and photos, video or audio, and an application code that renders other content or interacts with the visitor. Web content may be created, organized, and managed in an unlimited number of ways. Therefore a wide variety of WCMS systems have been built to handle many different situations. Some of these applications are general-purpose, providing a consistent general structure to any type of content. Others are more specific to types of content, intended audiences, or workflows [3].

In order to characterize a WCMS we could say the following:

- It manages small units of information (web pages); each unit of information is interconnected via a navigation structure or path.
- It is focused primarily on page creation and editing; it facilitates content creation, content control, editing, and many essential web maintenance functions by presenting the non-technical user with an interface that requires no knowledge of programming languages or markup languages to create and manage content.
- It provides a publishing engine that allows created or amended content to be made available to a website visitor.
- It often provides an approval process or workflow that ensures that content is validated before it is released or published to a website [8].

One of the first steps to managing web content is to decide what kind of content one wants to have on their site. The type of content to be published directly influences what type of authoring tools will be needed. For example, if one wants to post blog updates and link to other sites, the architecture of the site will differ from a company who wants to post videos or interactive tutorials. Keeping content organized will not only make it easier for visitors to navigate the site, but will help avoid content overlap or repetition in the future. There are a number of WCM tools on the market that can handle most of these issues. Choosing the right WCMS can be tricky, considering the dozens of available options. When looking for a WCMS, a company should first determine whether their website will require a basic CMS, mid-range CMS, or complex CMS [10].

## 2.4 What is Open Source?

The term “open source” is a key distinction. It means the software’s source code is freely available for everyone to see and change, but it also has many wider implications. While proprietary software is created, distributed and maintained by a business, with open source software these tasks are handled by a community of developers and users. Just how effective that community is at its job is an important consideration when choosing an open source CMS.

Open source content management systems are free in many ways. A user can do what he/she wishes with the product and the code behind it, extending and integrating it as they see fit. There’s no license cost for the software, and anyone can download and install them on a web server without cost, though it is likely they will have to pay for the server, or pay someone to install the system. An open source CMS takes work. A user either needs to put a lot of time into implementing and maintaining his/her system, or hire someone to do it for them [11].

## 2.5 Commercial CMS

Commercial CMS systems overcome some of the disadvantages of open source CMS systems. A distinct advantage of a commercial CMS system is that the vendor has control over the entire environment and most modules that are created for it, so it creates a more user friendly and secure end product. Since commercial CMS systems are not open source, the community does not have access to the Source Code and so possible hacker exploits are much less likely. There are usually fewer core CMS security updates from commercial vendors than open source vendors. Some commercial CMS systems are “cleanly coded” and well-documented, with an extensible architecture which allows a third party developer to read and follow the source code, understand the Application Programming Interface (API) and create custom enhancements to the CMS, without extensive issues.

There are also some commercial CMS systems that are good tools for specific purposes. For. eg. Microsoft SharePoint was designed to quickly create an internal Intranet, especially for clients who are using Microsoft tools. However SharePoint is not appropriate for implementing public Internet sites, as the tools are geared toward Intranets. Moreover customizing SharePoint for Internet applications can be a very expensive and inflexible process. In addition, expensive license fees for SharePoint, is another disadvantage. Other commercial CMS systems are highly complex and priced extremely high. An example of this would be Open Text, who formerly marketed “RedDot CMS” and now market a new Web Content Management system as part of their Web Solutions group. Licensing alone for a typical OpenText project begins at approximately $125,000 [12].

## 2.6 Comparative Study of Open Source CMS

Table 2.1 compares different existing Web CMS. It tells about Programming languages, databases and web servers used to build these CMS [13]. It also provides information on FTP support and UTF-8 support for these CMS.

**Table 2.1. Comparision of Different Web Content Management Systems**

| Product | Programming Language | Database | Web Server | FTP Support | UTF-8 Support |
|---|---|---|---|---|---|
| Joomla | PHP | MySQL | Apache | Provided as a free add-on | Limited support available |
| Drupal | PHP | MySQL, PostgreSQL | Apache, IIS | Limited FTP support | Available |
| WordPress | PHP | MySQL | Apache, mod rewrite | Available as a free add-on | Available |
| Plone | Python | Zope | Apache, IIS, Zope | Available | Available |
| TYPO3 | PHP | MySQL, PostgreSQL, Oracle, MSSQL | Apache, IIS | Available | Available |
| Open CMS | Java 1.4 | MySQL, PostgreSQL, Oracle, MSSQL | Tomcat, Apache | Not available | Available |

Source: InternetTips. Comparison With Other Web Content Management Systems, 2012. http://www.internettips.com/departments/joomla-web-content-management-system/5-of-6-comparison-with-other-web-content-management-systems, accessed Jan. 2012.

Joomla, Drupal and Wordpress are the most popular open source CMS being used today. Drupal has a steep learning curve and so it's difficult for a first time developer to understand the modules. It requires more advanced coding knowledge when it comes to customizing. Joomla has a better user interface than Drupal for newcomers. It is quite simple to add and edit pages. The design layer is almost the same as Drupal as they both do not have a lot of design templates. So the issue is that most of the Joomla sites will look the same. Wordpress is the perfect solution for small blogging websites as it has simple and user friendly interface. There are a lot of great design templates available for the websites. The issue with Wordpress is when one wants to do something outside the blogging world. Figure 2.1 shows the recent google trends for these CMS. The blue scale stands for Drupal, red for Joomla and yellow for Wordpress. As it's seen from the figure Wordpress leads the way due to its user friendly interface and large number of design templates.

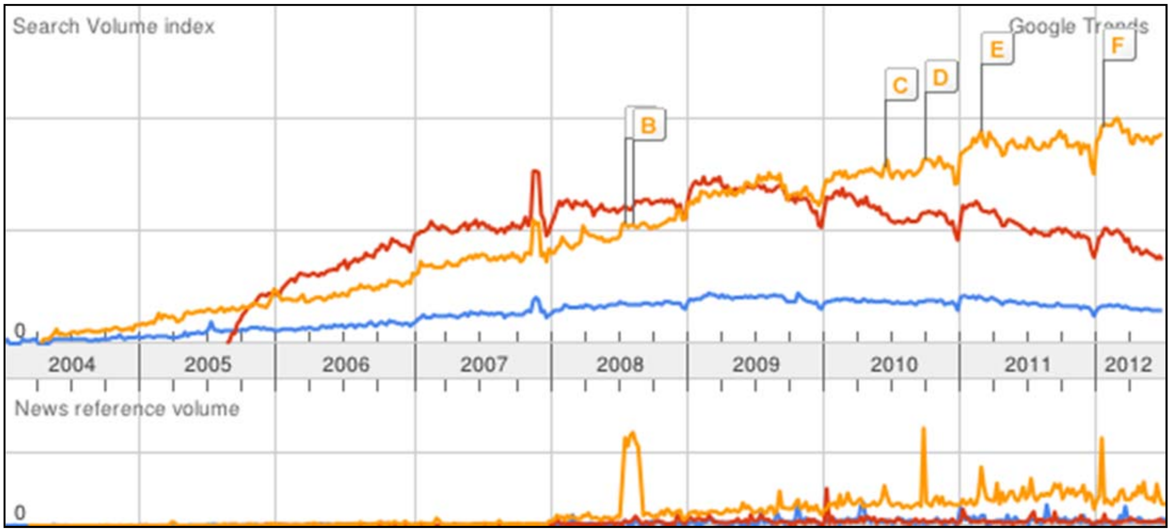


**Figure 2.1. Google Trends: Drupal, Joomla, Wordpress.**

## 2.7 Problems in the Existing Open source CMS

The main purpose of a Content Management System is to make it easy for even a novice computer user to maintain and manage a site. There are many CMSs available in the market, but deciding which is the most suitable will largely depend on the website's requirements and the budget.

Both Drupal and Joomla have a very bloated and confusing administration [14]. This is because both these packages have many advanced features in addition to the normal basic content management features. For example, polls providing the capabilities to capture votes on different topics in the form of multiple-choice questions, or news feeds, which provide syndicated content (RSS, RDF, and Atom feeds). This makes the CMS complex and difficult to use. Generally there will be some trade-offs between the complexity of the requirement and the simplicity in choosing a CMS. For example, if the website only requires a series of text based pages with a spattering of images, then a simple, basic CMS will be easy to use. However, if the requirements for the website include having multiple blogs, video and audio uploads, a forum, an events calendar, and an e-commerce facility, then it is good to choose a CMS capable of providing those facilities. Inevitably though, that CMS is going to be a little more complex to maintain [15]. If the website requires complex features, the best thing to do is to hide these complex features. A good user interface should make most common tasks the most prominent and hide rare tasks so that they do not get in the way [16]. There was research done by University of Minnesota Office for Information Technology's usability lab which identified many usability problems with Drupal's administration [17]. As of this writing, Drupal's administration interface is confusing and not user friendly. Joomla's administration usability and learning curve is better than Drupal's, but not enough to provide a noticeable advantage to the end-user over Drupal.

WordPress has a much better and very intuitive administration design [14], which makes it easier to learn.  It includes features such as drag and drop, resulting in the generation of code without technical intervention. It would be more correct to describe such products as 'website builders' than Web Content Management Systems.

The main feature that is not seen in most of today’s complex CMSs is intuitive and user-friendly website administration. Although there are no license fees with open source CMS, one still needs to pay a vendor to create an interface design or a theme, install the CMS, configure it for use, implement the “theme”, and also pay for ongoing support [12].

## 2.8 PERFORMANCE FOR DEPLOYED APPLICATIONS

Figure 2.2 [18] shows a graph that compares the page load time for textual information for different open source CMSs with Simple CMS. It shows that Joomla and Drupal have higher page load times than WordPress. SimpleCMS has nearly similar response

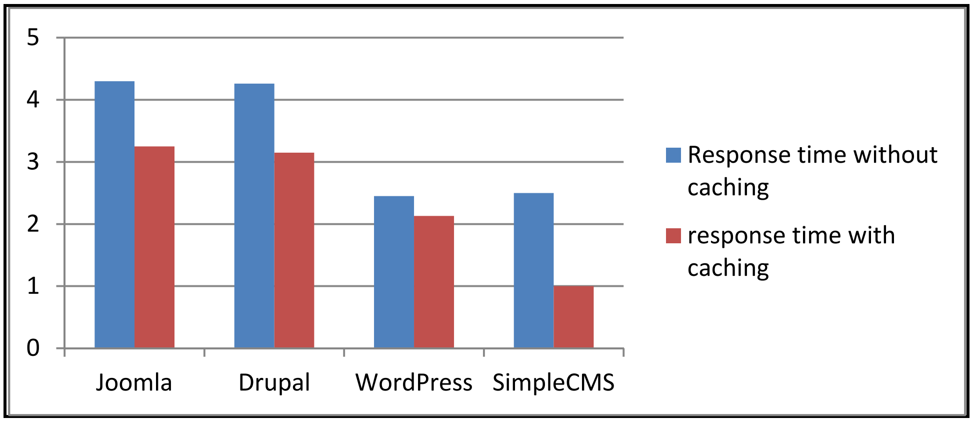


**Figure 2.2. Comparison page load times for different deployed CMS. Source: S. K. Patel, V. R. Rathod, and S. Parikh. Joomla, Drupal and WordPress – A Statistical Comparison of Open Source CMS, 2011. http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=06169111&tag=1http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=06169111, accessed Jan. 2012.**

time to WordPress. When any page gets loaded from the server to a client machine, the client machine stores some data on its site so when that page gets loaded again it takes less time to load compared to the first time. In case of textual information, Joomla and Drupal take 1.10 seconds less time than without the caching, while WordPress only reduces 0.32 seconds in PLT. It seems that it caches less data in memory [18].

## 2.9 To Build or To Buy?

There is no single best way to implement a CMS. When choosing between writing/creating a new CMS for yourself and using an existing software there are three key pieces of information need to be taken into account.

1. The functionality and features required which will decide how complex it will be.
2. How closely does an existing system match the requirements and what would it cost to implement it versus building a new CMS?
3. How soon is the CMS system needed?

If the requirements are not very complex, and if it is not needed immediately, it would be faster and less expensive to create a new Simple CMS. Otherwise, researching the different existing CMSs is required before starting to use one. In addition, one can also contribute to one of the existing open CMSs and can get feedback from experienced programmers who can help to configure it according to one's needs.

### 2.9.1 Using an Existing CMS

Pros:

- Lots of hosted solutions and online help is available for the off-the-shelf products. These products also provide live help and a dedicated resource to help the website builder. There are blogs which discusses the issues faced while installing or using such a product and can help a new user to overcome similar problems.
- Others can serve as administrator it if the user who integrated it is unavailable. Once the website is integrated into the CMS, any user can read the manual and use it to manage the content.

Cons:

- Research time in selecting the one that is closest to user requirements. There are lots of CMSs available in the market, but each of these products has different features. So one needs to research each of them and their features and select the closest one.
- An open source CMS is likely to have several pieces of functionality that are not required by a simple website, slowing the overall speed of the site [19].
- Steeper learning curve. These off-the-shelf products have a lot of advanced features and so it’s difficult for a first time user to understand all the modules.
- Bug fixes and security updates are released by the vendor. These CMS systems issue new versions of their own core CMS system quite often, normally to close security hole, fix bugs and add new features. And installing these new versions of CMS systems is required or the website will potentially be easy to hack if security holes are not fixed. Installing a new version of these CMS systems is complex and convoluted and requires disabling all existing modules prior to an upgrade [12].
- More limited in terms of customization

### 2.9.2 Building Your Own CMS

Pros:

- Total control over the features: The developers are familiar with the system and are able to build new features quicker without having to navigate through a certain design pattern set out by an open source solution [19].
- Low cost. It will be cheap to build one meeting your requirements than to get the off-the-shelf product to do what is needed.
- Easy to understand. Considering the requirements first, it will be easier for everyone in the firm to understand it.
- No bug fixes from other developers unlike the existing open source CMS.

Cons:

- Implementation time:  it is difficult for companies to make their own CMS because they might not have the appropriate in-house professionals to do so.

- Only you understand it and can fix it. There are problems in retaining the CMS knowledge when the individuals who created it leave the organization.

## 2.10 Advantages of This Project

- Simple CMS is easy to install and administer.
- The user interface is more intuitive.
- It is able to import and use static webpages in the CMS.
- Easily create editable regions for clients within the front-end of the website.
- It features easy menu page creation and a nice WYSIWYG editor, which allows easy content formatting, image uploading and image resizing.
- It also gives the ability to add a new stylesheet or modify the existing ones.
- The workflow will only allow the newly created page to be visible to the users when the administrator approves and publishes it.
- The audit trail report displays the information regarding all changes made to the website pages and thus helps in deterring fraud by keeping a record of which pages were modified or deleted.

## 2.11 Project Limitations

You cannot create/build a new template using Simple CMS. There is only one editable region on a page. One can edit a simple website that has a homepage with all the other pages as the tabbed menu/navigation bar. There can be only one navigation bar whose pages can be modified.

# CHAPTER 3

# TOOLS

Several technological tools were used in this project; each is summarized in the sections that follow.

## 3.1 HTML

HTML stands for Hyper Text Markup Language. A markup language is a language that annotates text in a way that is syntactically distinguishable so that the computer can manipulate it. It is a set of markup tags used to describe web pages. The tags are what separate normal text from HTML code. They are the words between the <angle-brackets>. Different tags will perform different functions, like rendering images or tables. It is a combination of words and symbols which give instructions on how the document will be presented. The tags themselves don't appear when you view your page through a browser, but their effects do.

Markup is what HTML tags do to the text inside them. They mark it as a certain type of text (italicized text, for example) [20]. HTML documents contain HTML tags and plain text. The content on a HTML page will be static. In order to change the content, the editor needs to have some knowledge about HTML and change the content accordingly.

## 3.2 CSS

CSS stands for Cascading Style Sheets. It is used to control the style and layout of multiple web pages all at once. Styles define how to display HTML elements. CSS overrides the browser's default settings for interpreting how tags should be displayed, letting you use any HTML element indicated by an opening and closing tag to apply style attributes defined either locally or in a stylesheet.

External Style Sheets can save a lot of work. They are stored in CSS files. Stylesheets contain rules, composed of selectors and declarations that define how styles will be applied. The selector (a redefined HTML element, class name, or ID name) is the link between the HTML document and the style. There are two different kinds of selectors: types (HTML element tags) and attributes (such as class and ID names) [21].

## 3.3 JAVASCRIPT

JavaScript is a client side scripting language designed to add interactivity to HTML pages. A scripting language is a lightweight programming language. It is usually embedded directly into HTML pages. It is an interpreted language which allows the scripts to execute without preliminary compilation.

JavaScript can react to events. A JavaScript can be set to execute when something happens, like when a page has finished loading or when a user clicks on an HTML element. JavaScript can read and write HTML elements and can also change it’s content and properties. A JavaScript can be used to validate form data before it is submitted to a server. This saves the server from extra processing. A JavaScript can be used to detect the visitor's browser, and depending on the browser, load another page specifically designed for that browser. Finally, JavaScript can be used to create cookies and to store and retrieve information on the visitor's computer [22].

## 3.4 PHP

PHP (Hypertext Preprocessor) is a widely-used open source server-side scripting language for web development and can be embedded into HTML. PHP pages contain HTML with embedded code [23]. PHP offers several advantages:

- PHP runs on different platforms (Windows, Linux, Unix, etc.).
- PHP is compatible with almost all servers used today (Apache, IIS, etc.).
- PHP is free to download from the official PHP resource: www.php.net.
- PHP is easy to learn and runs efficiently on the server side [24].

What distinguishes PHP from something like client-side JavaScript is that the code is executed on the server, generating HTML, which is then sent to the client. The client receives the results of running that script, but does not know what the underlying code was [23].

## 3.5 MYSQL

MySQL is a database server is ideal for both small and large applications. It supports standard SQL and compiles on a number of platforms. It is free to download and use [24].

## 3.6 JQUERY

jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. jQuery is designed to change the way JavaScript is written. Therefore jQuery greatly simplifies JavaScript programming [25].  This project specifically uses jQuery for form validation and to implement tabbed menu.

It is free, open source software.  Dual-licensed under the MIT License or the GNU General Public License, jQuery's syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications. jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, theme-able widgets. The modular approach to the jQuery library allows the creation of powerful dynamic web pages and web applications.

jQuery includes the following features:

- DOM traversal and modification
- DOM manipulation based on CSS selectors that uses node elements name and node elements attributes (id and class) as criteria to build selectors
- Events
- Effects and animations
- Ajax
- Extensibility through plug-ins
- Utilities - such as user agent information, feature detection
- Compatibility methods that are natively available in modern browsers but need fallbacks for older ones (for example, the inArray() and each() functions)
- Cross-browser support [26].

# CHAPTER 4

# FEATURES, DESIGN AND IMPLEMENTATION

This chapter discusses the various features, architecture and design components of the project. Screen images of the graphical user interface for select functions are in the Appendix.

## 4.1 FEATURES

The following are some of the key features of web based CMS and its implementation in this project.

### 4.1.1 Default Templates

Standard output templates built using HTML and CSS that can be automatically applied to new and existing content, allowing the appearance of all content to be changed from one central place. This separates the content from the presentation.

This project has 3 default templates that can be used according to the requirement. The user can also integrate their own template (HTML and CSS) files into the Simple CMS engine and can set it as the default theme. Figure A.1 shows the screen used to configure the template and select a start page.

### 4.1.2 Access Control

Session management and user accounts are typically included in a CMS. Roles can then be created and assigned any number of permissions. Users are then assigned roles, giving them the permissions included in those roles. For example, an administrator might create an "Editor" role, assigning editing permissions of all content to that role, rather than granting that role to the appropriate user accounts. When users with that role log into the CMS, they automatically have access to edit content [3].

This project has three user defined roles:  admin, editor and designer. Different roles can be assigned to different user accounts and thus each has access to different sections of the CMS. The admin user has access to all the sections, whereas the editor can modify existing pages and the designer can manage stylesheets. All the users can modify their own user profile.

### 4.1.3 Easily Editable Content

Content is separated from the visual presentation of a site, which makes it much easier and quicker to edit and manage. The standard web element for entering large blocks of text is the HTML textarea. To format text using a textarea, a user needs to include HTML elements or some other application-specific markup. [3]. To overcome this difficulty, most WCMS software includes a rich text editor for editing rich text within web browsers, which presents the user with a WYSIWYG (what-you-see-is-what-you-get) editing area. WYSIWYG editors allow non-technical individuals to create and edit content. These editors allow the users to see what their content will look like while they are in the process of editing it. The aim is to reduce the effort for users trying to express their formatting directly as valid HTML markup [27]. Rich text editors manage the content formatting.

There are different rich text editors available in the market. TinyMCE is one of these, but it does not have a free file/image upload functionality. This project uses CKEditor, which is an open source WYSIWYG text editor from CKSource that can be used in web pages. It brings to the web common editing features found on desktop editing applications like Microsoft Word and OpenOffice. It is lightweight. CKFinder is a powerful and easy to use Ajax file manager for Internet browsers. This project uses CKEditor integrated with the demo version of CKFinder [28]. Figure A.2 shows the CKEditor used in this project and Figure A.3 shows the CKFinder integrated with CKEditor for managing server files.

### 4.1.4 Workflow Management

Workflow features are implemented to assist teams with the process of editing and publishing content. The content publication is generally immediate or scheduled. For example, the simplest type of workflow is used in this project: add new content (page) to a publishing queue, where it awaits final approval from a site administrator before the content appears on the web site. More robust workflow might delegate work to separate types of users, often based on security roles. An editor, for example, might automatically be notified when new content is posted, letting him/her edit and approve the content before publishing [3].

This project uses a simple workflow where the newly added page is first entered into the publishing queue and, once the administrator approves and publishes it, the new page appears as of the menu items on the homepage.

### 4.1.5 Audit Trial Report

The Audit Trail Report shows the date and time that every page was created or modified. In addition to the date and time, the name of the user who made the entry or modification is also listed, along with the type of action performed: Created, Modified, or Published/Unpublished. This report can help deter fraud by keeping a record of which pages were modified or deleted; and it can let you know what data needs to be reentered in the case of data corruption. Only the admin user can view this report. Figure A.4 shows the audit trial report screen.

## 4.2 DATABASE ARCHITECTURE

There are two basic approaches for storing the data and other system information in the CMS, such as users and permissions: (1) storing whole documents as XML in a single field, and (2) storing documents of different types in separate tables. The first approach assumes that you store all documents in a single database table that has a single field for holding the whole document. However, when it is necessary to display more than one XML document on the page there are severe problems. Just consider a product catalog with dozens of items:  when you need to display this list of products, you need to go through all database records, retrieve XML, parse and render it. This process is not straightforward and the performance will not be optimal. Also the XML approach does not support typed data such as integer, datetime, etc. as all of the content is serialized to text. So you will need to ensure that such typed values or decimal numbers are parsed and displayed in the correct format.

This project uses the standard relational database schema used by most of the designers today. This traditional relational database approach allows you to easily retrieve data from the database and display them on the website using standard HTML controls, such as table, while having complete control over the format [29]. The admin and the user interface use the same database engine. In order to manage the content of the site the user does not need to edit the HTML content or manually go to write the query, but he/she can login to the CMS and make the required changes. There are five tables required to be created in the database on the server where the application is hosted. Table 4.1 describes each of the tables used in this project.

**Table 4.1. Simple CMS Tables and Descriptions**

| TABLE | Description |
|---|---|
| Content | Stores information of website menu pages |
| Users | Stores user information: first name, last name, email, username, password and roleID |
| Roles | Stores roleID and role name |
| Configurations | Stores the configurations of website like the template and the stylesheet used |
| audit_trial | Stores information about the modifications made to the web page to generate the report |

Figure 4.1 describes the data types for each of the fields defined in all of the above-described tables. It also defines the primary key and the foreign keys for each of those tables, as well as the type of relationships existing between them.

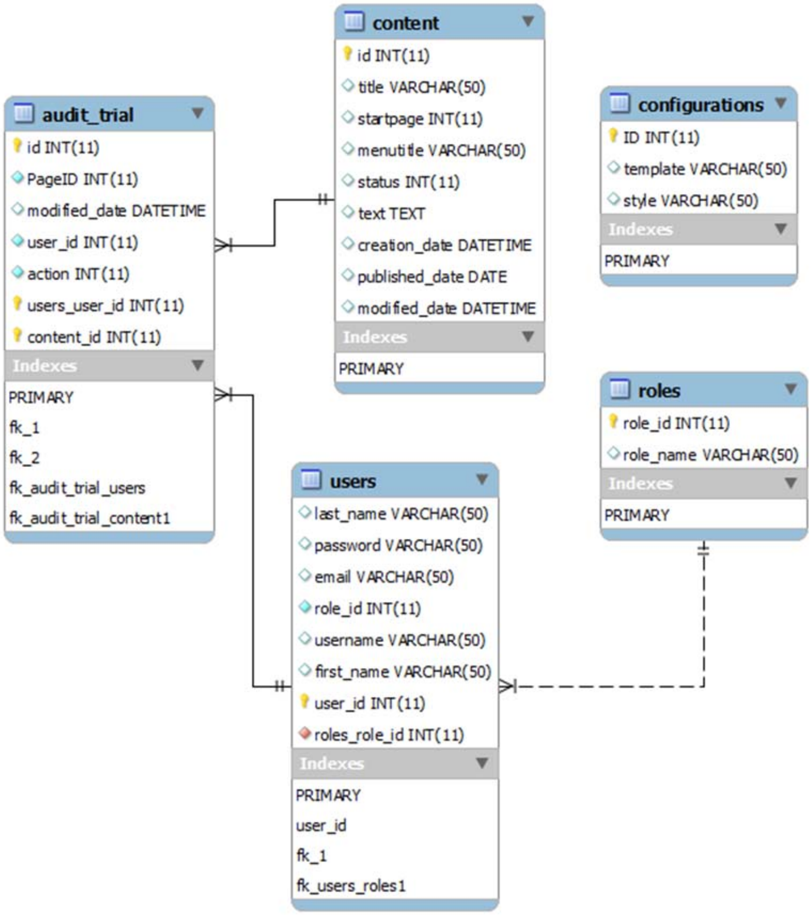


**Figure 4.1. Database architecture of Simple CMS.**

## 4.3 High-level Overview (System Architecture)

Figure 4.2 shows the high level architecture of the Simple CMS engine. Simple CMS is designed to allow users with specific roles to access certain sections of the CMS admin interface. The administrators have access to manage users, configure homepage, publish/unpublish a page. The editors can only modify the page content. The designers can only modify the stylesheets of different templates. General users directly access the web page. When the CMS engine receives the webpage request, it reads the settings, then runs an SQL query to select data from the database. It then binds the data to the selected template and displays the page to the visitor. This separates the content from the presentation allowing the administrators to just change the design template with the same content.

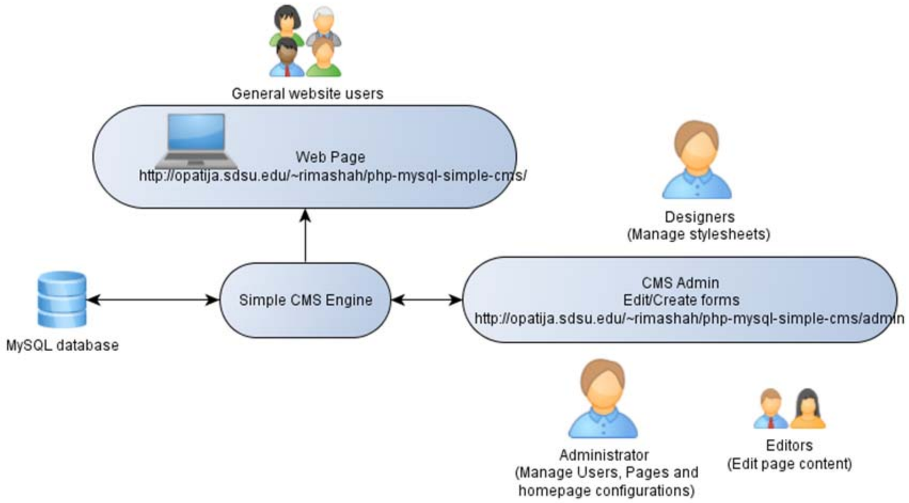


**Figure 4.2. High-level overview of Simple CMS.**

## 4.4 Use Cases

Use cases for several core functionalities are briefly described in this section.

### 4.4.1 Login

Figure 4.3 shows that any authorized user can go to the administration interface and modify their own profile.

### 4.4.2 Manage Pages

Figure 4.4 describes that a user with admin role can manage all pages, but the editor can only modify the existing pages.

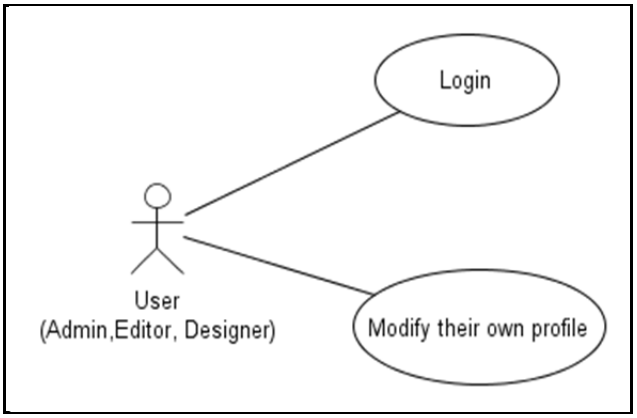


**Figure 4.3. Login use case.**

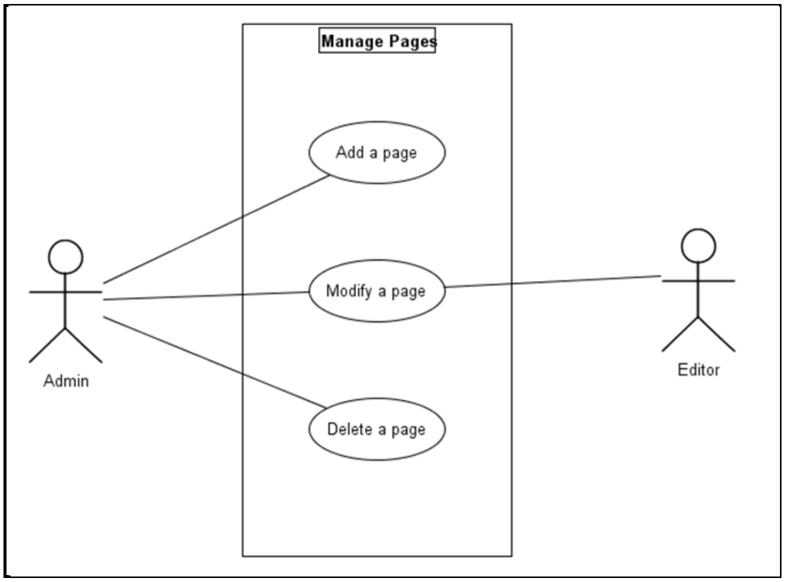


**Figure 4.4. Manage pages use case.**

## 4.4.3 Manage Users

Figure 4.5 show that only a user with admin role can manage other users.

## 4.4.4 Manage Stylesheets

Figure 4.6 show that a user admin can manage all stylesheets associated with different templates, while the user with designer role can only modify existing stylesheets.

## 4.4.5 Website Configuration

Figure 4.7 show that a user admin can configure the website which includes setting up a home page and a template for the website.

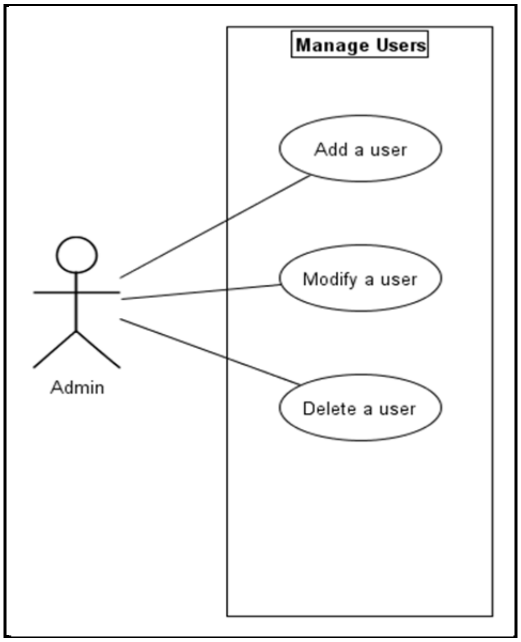


**Figure 4.5. Manage users use case.**

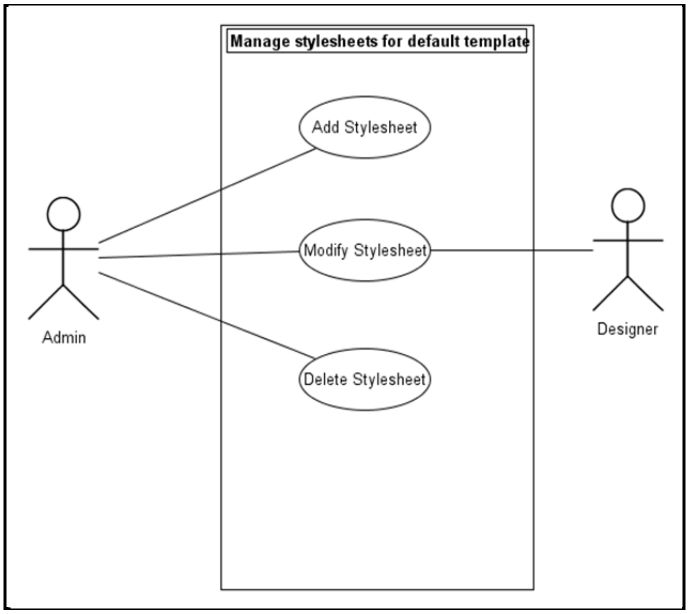


**Figure 4.6. Manage stylesheets use case.**

## 4.5 BASIC OPERATIONS

After your template is integrated into the CMS, you can modify its content.

### 4.5.1 Login Process

Figure 4.8 shows how the login and the authentication works for the Simple CMS engine. It uses the md5() function to store an encrypted password in the database. This

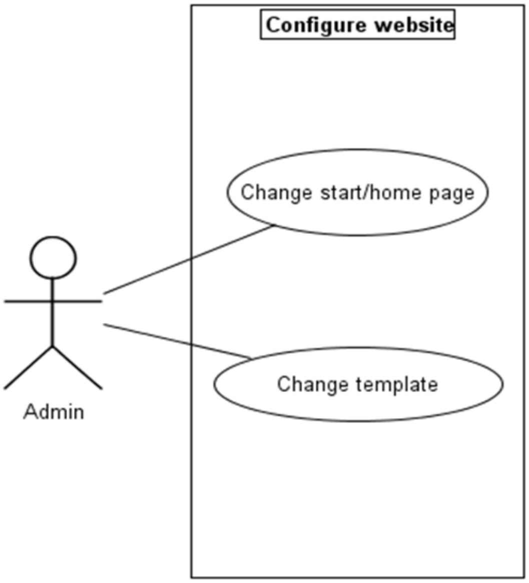


**Figure 4.7. Website configuration use case.**

function calculates the MD5 hash of the string passed into it. On the user end when the password is entered, it is again hashed and compared to the one stored in the database. The following piece of code validates the input login details and redirects to the website.

```
    <?php if(isset($_POST["submit"]))
    {
        $username = $_POST['formlogin'];
        database_connect();
        $credentials = mysql_query("select user_id,role_id,username, password from
users where username = '$username'");
        if(mysql_num_rows($credentials)>0)
      {
        $row = mysql_fetch_row($credentials);
        if(isset($_POST['formpass']) && isset($_POST['formlogin'])) {
            if($row[3] == MD5($_POST['formpass']))
            {
                    $_SESSION["userid"] = $row[0];
                    $_SESSION["roleid"] = $row[1];
                    $_SESSION["username"] = $row[2];
```

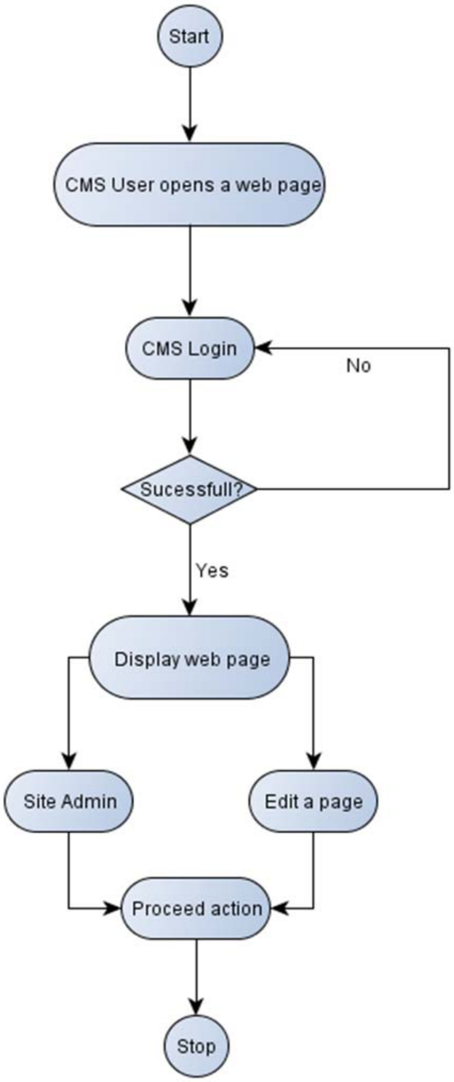
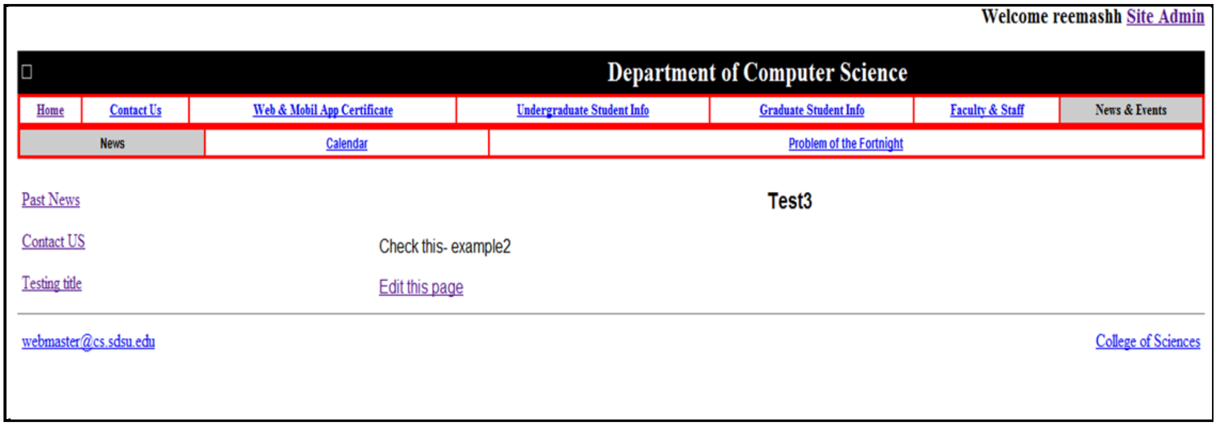


**Figure 4.8. Login activity.**

```
                $_SESSION["loggedin"] = "1";    ?>
        <script language="javascript">
            location.replace("/~rimashah/php-mysql-simple-cms");
        </script>
<h4><a href='admin'>you are now logged in, continue to the admin section</a></h4>
        <?php }
            else
echo "<font color='#990000'><center>Wrong Password</center></font>";
        }
        }
        else
        echo "<font color='#990000'><center>User does not exist</center></font>";
}
```

Once the user logs in it is redirected to the website's user interface. Figure 4.9 shows the News and Events section of San Diego State University's Computer Science department, integrated into the Simple CMS engine. To edit any page in the site, click on "Edit this page" link which redirects directly to the "Manage Pages" section or just click on "Site Admin" to go to the administration home page.



Test3

Check this- example2

**Figure 4.9. User interface.**

## 4.5.2 Add/Modify Page

Figure 4.10 shows how the page creation and editing works in the Simple CMS. When the user adds a new page it is unpublished by default. The administrator needs to publish it in order for the page to appear on the site. It will appear as one of the menu options in the sidebar navigation and clicking on that will take the user to the newly added page. Editing the page content does not change the state of the page. As shown in Appendix Figure A.2 the add/edit screen has the following input elements: Menu title which appears on the sidebar navigation, Page title and the page content. There are simple input text fields for menu title and page title but the page content has a text area integrated with a rich text editor i.e. CKEditor that allows easy content formatting without having to deal with the HTML and CSS code. It also has the file upload tool to allow the users to upload a file to server and use that uploaded file in the page content. To remove any of the menu pages from the website the

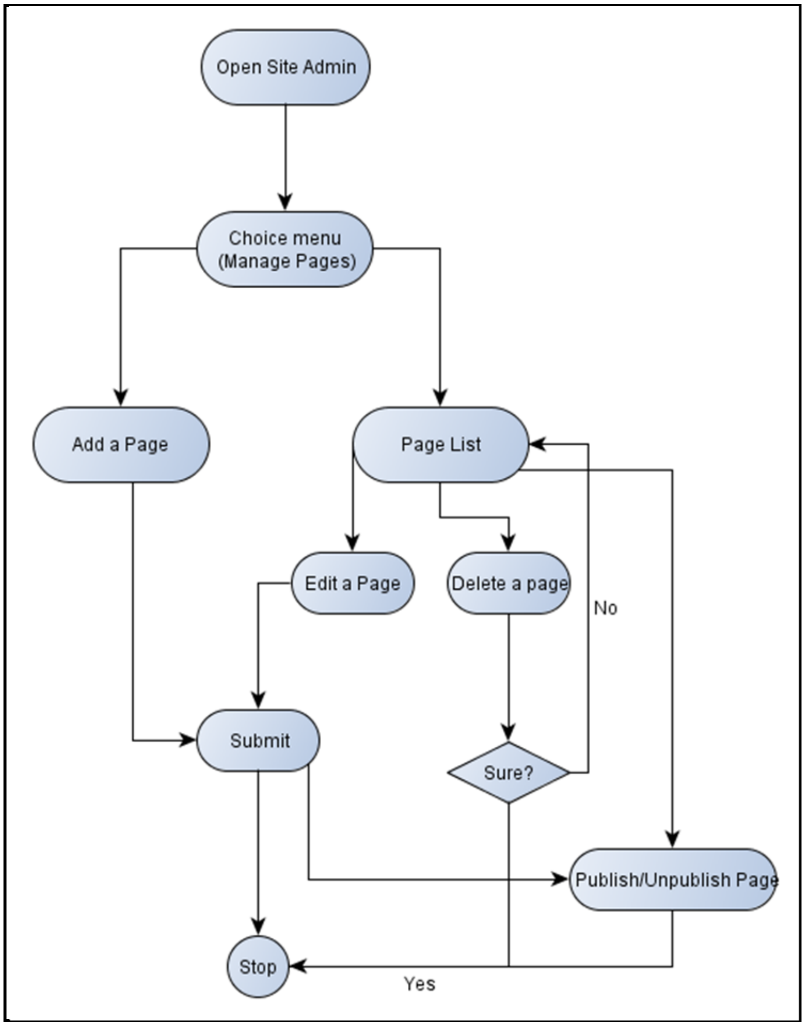


**Figure 4.10. Add/modify page activity.**

administrator has to manually click on the unpublish button. Figure A.5 shows the “Manager Pages” section of the Simple CMS.

### 4.5.3 Add/Modify User

Figure 4.11 shows workflow to create a new user in the Simple CMS. Only the admin user is allowed to add a new user, modify other user’s profile or delete an existing user. Fig. A.5. shows the screen which has the required fields for first name, last name, username, password and email. Only when the admin adds a user, he/she will be able to login to the site admin and perform respective action. Figure A.6 shows the “Manage Users” section of the Simple CMS.

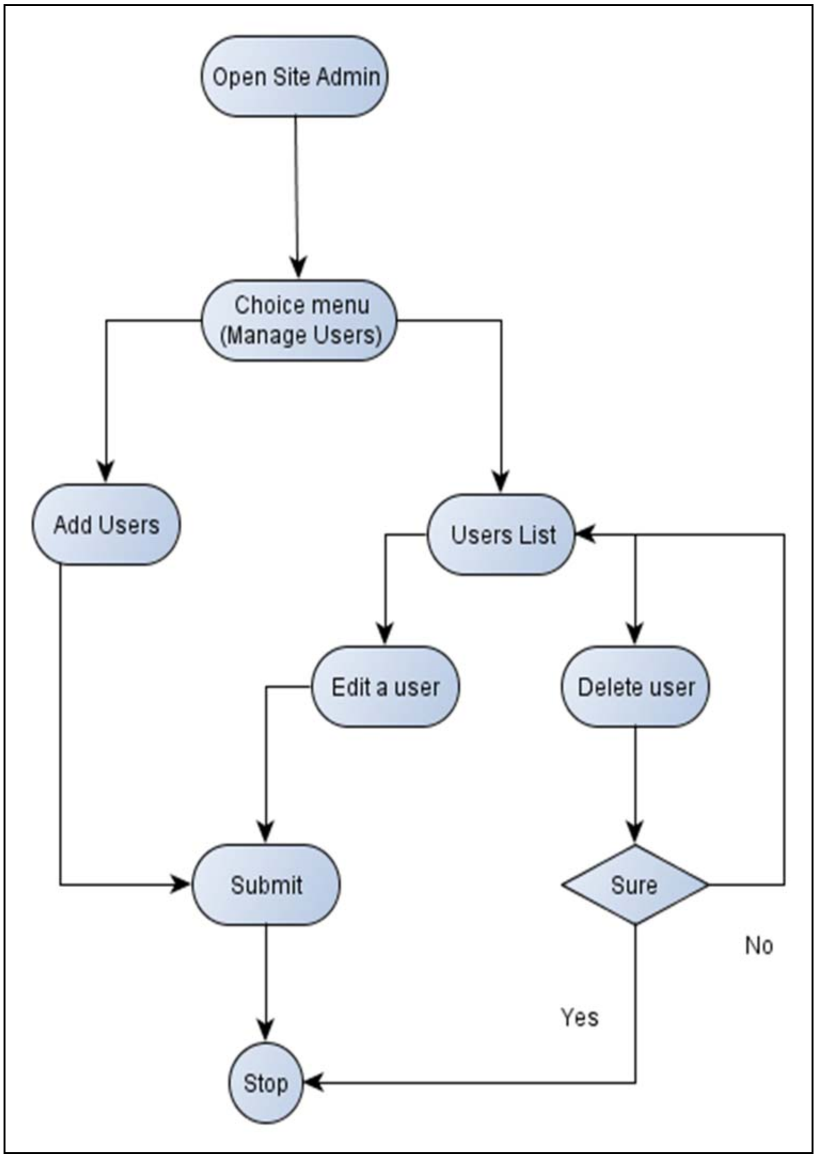


**Figure 4.11. Add/modify users activity.**

### 4.5.4 Add/Modify Stylesheet

This activity has a similar workflow to the add/modify users activity. Creating a new stylesheet adds a new CSS file to the “css” folder. To modify a css file, click on the edit link related to that particular stylesheet. The content of the stylesheet is then available to modify on the text area in the adjacent section. The following code creates a new stylesheet in the “css” folder.

```
if(isset($_POST['submit'])) {
        $title = $_POST['style_title'];
        $text = $_POST['text'];
        $myFile = $title.".css";
        $fh = fopen("../css/".$myFile, 'w') or die("can't open file");
        $stringData = $text;
        fwrite($fh, $stringData);
        fclose($fh);
        if($fh) echo "<br>The sylesheet is successfully added.";
    }
```

### 4.5.5 Delete Process

The same delete activity works for pages, users and stylesheets. There is a single file which performs deletion based on the page parameter. When the user clicks on delete, a query string along with the page parameter is passed to the file. The parameter can either be menu, user or stylesheet. If it is a menu page or a user, it deletes that particular entry from the related database table and redirects the user back to the same page. In case of stylesheets it unlinks or removes the file from the “css” folder. The following code shows how it works:

```
        if($page=="menu")
        {
            $sql = "DELETE FROM content WHERE id = $id;
            $query = mysql_query($sql)or die("There was a problem while
            deleting: ". mysql_error());
        }
        elseif($page == "style")
        {
            $filename = $_GET['id'].".css";
            unlink("../css/".$filename);
        }
        elseif($page == "user")
        {
            $sql = "DELETE FROM users WHERE user_id = $id;";
            $query = mysql_query($sql)or die("There was a problem while
            deleting: ". mysql_error());
```

```
}
```

### 4.5.6 Upload/Download File Process

There is no separate screen for this activity, but one can access these features using the CKFinder (File upload) tool integrated with CKEditor. Figure A.4 shows the screenshot of the CKFinder interface. To upload a file to the server, the user needs to go to the “Manage Pages” section. Click on “Link” in the rich text editor for uploading any kind of file or click on “Image” for uploading an image, then go to “Browse server”. Click on “Upload” then “Add Files” which pops up the file open dialog box to browse local files. Select the files to be uploaded and click “Upload” again. This will upload all the selected files to the CKFinder’s “userfiles” folder inside “_samples” directory on the web server. To download a file, “Browse server” as above and select the file to download, right click on it and hit “Download”.

## 4.6 How to Integrate the Simple CMS with Your Template

- Step1: Unzip the project file and copy the folders to your web server.
- Step 1: Copy the HTML file in the project’s “templates” directory, the css file into the “css” directory and images to “img” directory of php-mysql-simple-cms folder
- Step 2: Edit the HTML file to replace the dummy content with the dynamic variable $text which gets the content from the database. Also change the location of the css file to “css/style.css” and the src for all images to “img/image.jpg”.
- Step 3: Edit the css file to change the image path location to: “img/image.jpg”.
- Step 4 (optional): If your template has a dynamic sidebar menu. You can replace it with the following code:

```
while ($row = mysql_fetch_assoc($navresult)) {
                                    $navid = $row['id'];
                                    $menutitle = $row['menutitle'];
                                    $startpage = $row['startpage'];
         if ($startpage == 1) {
         $href = "http://opatija.sdsu.edu/~rimashah/php-mysql-simple-cms/index.php";
         }else{  $href = "page.php?id=$navid";
              }
```

- Step 5: To create the admin databases simply run the “install.php” found in the website’s main directory.

- Step 6: Once all the above steps are completed, you can login to the admin section and click on “Configure Homepage”, select your template and save the configurations. The user end will have the editable content replaced by the test content which can be changed from the admin section.

# CHAPTER 5

# CONCLUSION

Depending on the requirements and the budget, CMS required for a system can either be built from scratch or one can use an existing open source or commercial product. The key feature to keep in mind while building a Web based CMS is intuitive and user-friendly administration. It should include important CMS functions like separation of layout and content, formatting the content with “WYSIWYG” editor and managing the workflow for the content approval before it is published to the users. I have successfully implemented the Simple CMS for the “News and Events” section of the San Diego State University’s Computer Science department.

The primary features of Simple CMS are: (1) easily editable content, (2) templates (default/auto), (3) user authentication, and (4) workflow management. The administration interface is relatively simple. It features one navigation bar that has different tabs for respective functions, such as Manage Pages, Manage Users, Manage Stylesheets, Configure homepage and Audit trial report. It has a workflow which will only allow the publishing of the new content when it is approved by the administrator. It has defined users and roles for users to add, delete or update content within the website. Each user can modify data according to his access rights. Only admin can assign roles to users and has full control over each user and his activity.

The user can easily integrate a new template, which is basically the HTML and CSS files, into this application. Once it is integrated, the user can easily format and edit the page content using the rich text editor without having to deal with the HTML or the CSS code. Thus this project will be useful to the users with less technical expertise, allowing them to easily manage the content of their page.

There are opportunities for future development in this project on at least three specific functions:

1. Multilingualism: this feature will allow displaying the content in multiple languages, potentially allowing users to view the content in their own native language.

2. Versioning: CMS software may allow the process of versioning by which pages are checked in or out of the WCMS, allowing authorized editors to retrieve previous versions and to continue work from a selected point. Versioning is useful for content that changes over time and requires updating, but it may be necessary to go back to or reference a previous copy.
3. Content distribution: CMS software often assists in content distribution by generating RSS and Atom data feeds to other systems. They may also e-mail users when updates are available as part of the workflow process [9].

## REFERENCES

[1] Wikipedia. Content management, 2012. http://en.wikipedia.org/wiki/Content_management, accessed Dec. 2011.

[2] Wikipedia. Content management system, 2012. http://en.wikipedia.org/wiki/Content_management_system, accessed Dec. 2011.

[3] DocForge. Content management system, 2011. http://docforge.com/wiki/Content_management_system, accessed Dec. 2011.

[4] Labyrinth Solutions. A (brief) history of CMS development, 2012. http://www.contegro.com/info-center/designers-blog/blog-article/_disc_/engage-the-web/_thread_/a-brief-history-of-cms-development, accessed Dec. 2011.

[5] Vasont Systems. Crash course for content management, n.d. http://www.vasont.com/resources/who-needs-cms.html, accessed Dec. 2011.

[6] Wikipedia. Enterprise content management, 2012. http://en.wikipedia.org/wiki/Enterprise_content_management, accessed Dec. 2011.

[7] A. Bach. Types of CMS's, 2006. http://www.webdotdev.com/nvd/content/view/205/, accessed Dec. 2011.

[8] Contentmanager.eu.com. What is web content management (WCMS), 2011. http://www.contentmanager.eu.com/wcms.htm, accessed Dec. 2011.

[9] Wikipedia. Web content management system, 2012. http://en.wikipedia.org/wiki/Web_content_management_system, accessed Dec. 2011.

[10] E. Mullan. What is web content management (WCM)?, 2012. http://www.econtentmag.com/Articles/ArticleReader.aspx?ArticleID=79727, accessed Dec. 2011.

[11] Project BIG Fish. Open source content management, 2010. http://www.projectbigfish.com/opensource, accessed Dec. 2011.

[12] D. L. Nye. Content management system comparison, 2012. http://www.jkdesign.com/content-management-system-comparison-jkdesign.pdf, Jan. 2012.

[13] InternetTips. Comparison with other web content management systems, 2012. http://www.internettips.com/departments/joomla-web-content-management-system/5-of-6-comparison-with-other-web-content-management-systems, accessed Jan. 2012.

[14] Comentum. Drupal vs Joomla vs WordPress, 2010. http://www.comentum.com/drupal-vs-joomla-cms-comparison.html, accessed Jan. 2012.

[15] Frieze Design. Content management systems (CMS), 2012. http://www.friezedesign.co.uk/website_content_management_system.html, accessed Dec. 2011.

[16] T. Tate. Minimizing complexity in user interfaces, 2009. http://uxdesign.smashingmagazine.com/2009/10/07/minimizing-complexity-in-user-interfaces/, accessed Jan. 2012.

[17] Drupal. Report from the University of Minnesota Drupal usability testing, 2011. http://drupal.org/node/1175694, accessed Jan. 2012.

[18] S. K. Patel, V. R. Rathod, and S. Parikh. Joomla, Drupal and WordPress – A statistical comparison of open source CMS, 2011. http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=06169111&tag=1http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=06169111, accessed Jan. 2012.

[19] Jaggy Snake. Use an open source CMS or a custom built one?, 2009. http://www.jaggysnake.co.uk/blog/use-an-open-source-cms-or-custom-built-one, accessed Jan. 2012.

[20] R. Shannon. What is HTML?, 2012. http://www.yourhtmlsource.com/starthere/whatishtml.html, accessed Jan. 2012.

[21] Teachbook. CSS guide, n.d. http://blog.teachbook.com.au/index.php/computer-science/web-development/html-guide, accessed Dec. 2011.

[22] W3Schools. JavaScript introduction, 2012. http://www.w3schools.com/js/js_intro.asp, accessed Dec. 2011.

[23] PHP. What is PHP?, 2012. http://www.php.net/manual/en/intro-whatis.php, accessed Dec. 2011.

[24] W3Schools. PHP introduction, 2012. http://www.w3schools.com/php/php_intro.asp, accessed Dec. 2011.

[25] jQuery. Homepage, 2012. http://jquery.com/, accessed Dec. 2011.

[26] Wikipedia. JQuery, 2012. http://en.wikipedia.org/wiki/JQuery, accessed Dec. 2011.

[27] Wikipedia. Online rich-text editor, 2012. http://en.wikipedia.org/wiki/Online_rich-text_editor, accessed Jan. 2012.

[28] CKEditor. Homepage, 2012. http://ckeditor.com/, accessed Feb. 2012.

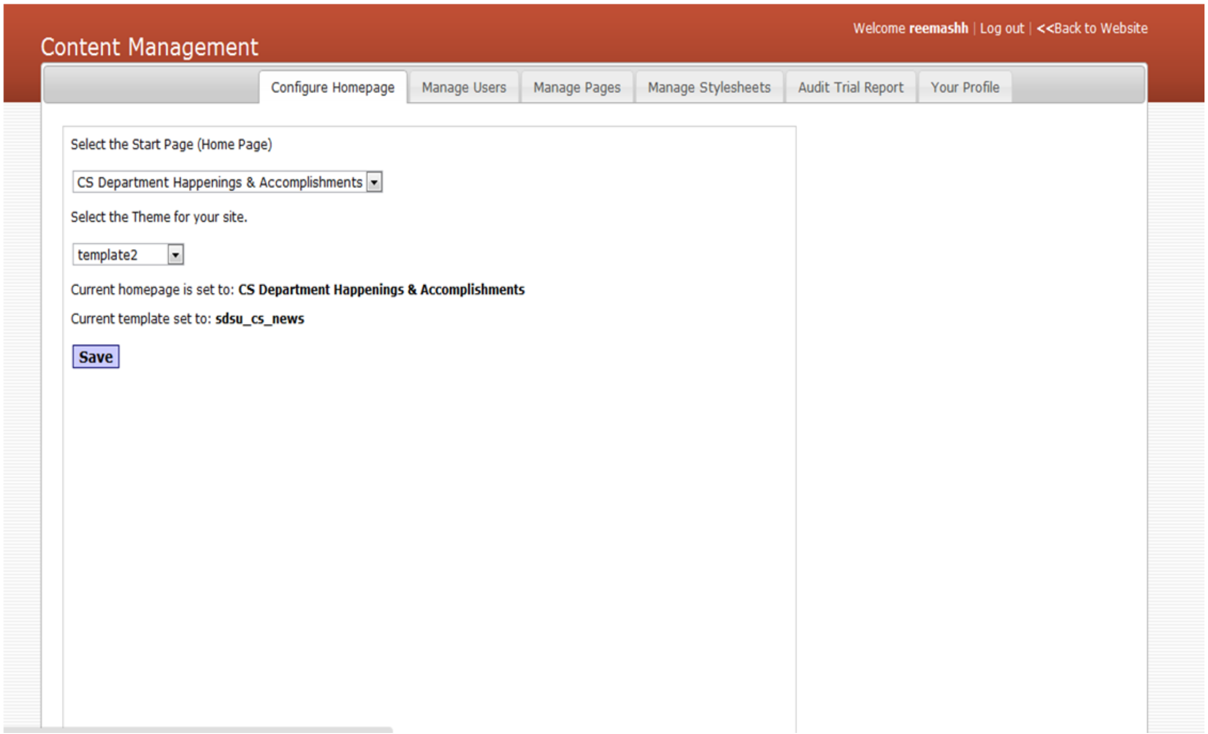
[29] P. Palas. Choosing between XML and SQL in .Net content management systems, 2006. http://www.codeproject.com/Articles/13991/Choosing-between-XML-and-SQL-in-NET-Content-Manage, accessed Jan. 2012.

# APPENDIX

# GRAPHICAL USER INTERFACE

### Configure homepage:

Figure A.1 shows the screenshot of the “Configure Homepage” section of the administration interface. A user with proper privileges can change the home page, as well as select a different template or theme for their web page.



Select the Start Page (Home Page)

CS Department Happenings & Accomplishments

Select the Theme for your site.

template2

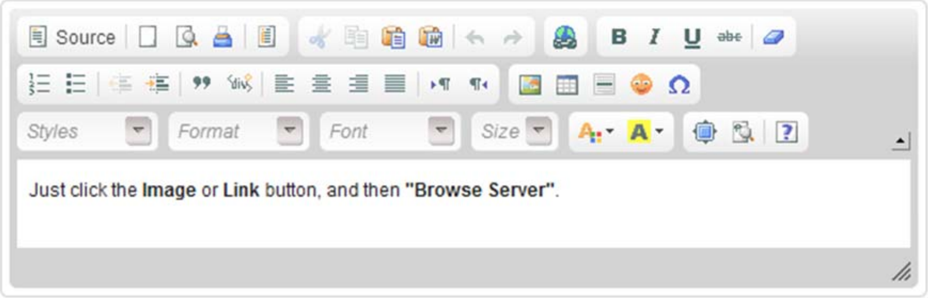
Current homepage is set to: **CS Department Happenings & Accomplishments**

Current template set to: **sdsu_cs_news**

**Figure A.1. Website configuration.**

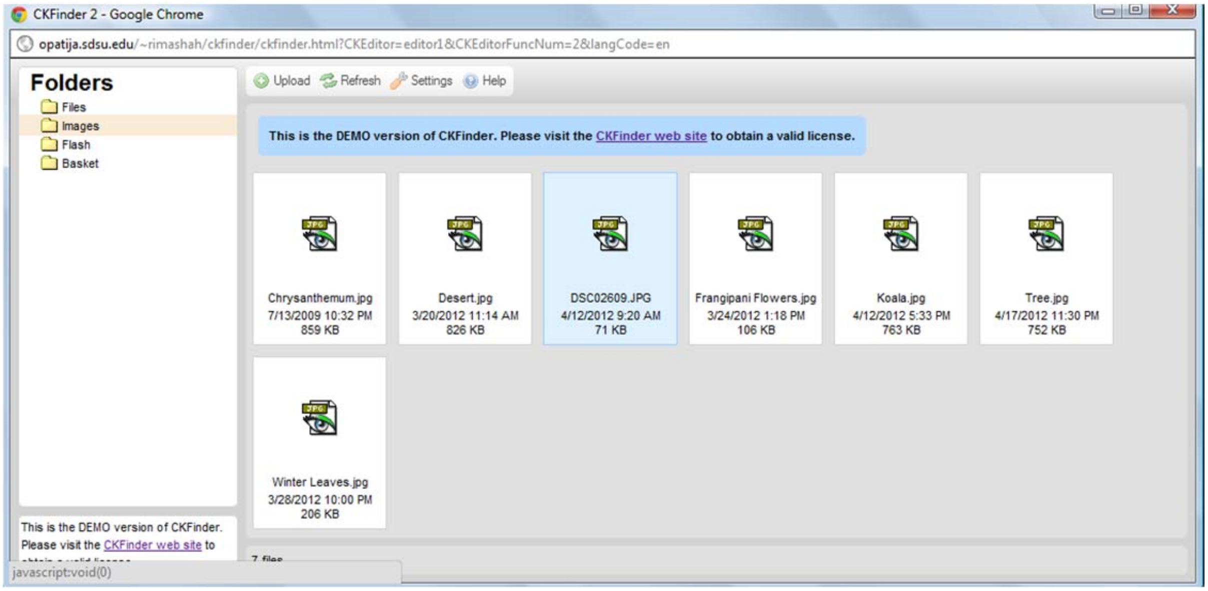
**CKEditor and CKFinder tools:**

Figure A.2 shows the CKEditor used for content formatting (change text color, font, size etc.). This tool has an Image and Link button, which is used to insert image URLs or browse the server for existing images. Figure A.3 shows the CKFinder tool, which is used as a File Manager to upload and browse server files and use those in the content.



Just click the **Image** or **Link** button, and then **"Browse Server"**.

**Figure A.2. Rich text editor (CKEditor).**


CKFinder 2 - Google Chrome

opatija.sdsu.edu/~rimashah/ckfinder/ckfinder.html?CKEditor=editor1&CKEditorFuncNum=2&langCode=en

This is the DEMO version of CKFinder. Please visit the CKFinder web site to obtain a valid license.

Chrysanthemum.jpg
7/13/2009 10:32 PM
859 KB

Desert.jpg
3/20/2012 11:14 AM
826 KB

DSC02609.JPG
4/12/2012 9:20 AM
71 KB

Frangipani Flowers.jpg
3/24/2012 1:18 PM
106 KB

Koala.jpg
4/12/2012 5:33 PM
763 KB

Tree.jpg
4/17/2012 11:30 PM
752 KB

Winter Leaves.jpg
3/28/2012 10:00 PM
206 KB

This is the DEMO version of CKFinder.
Please visit the CKFinder web site to

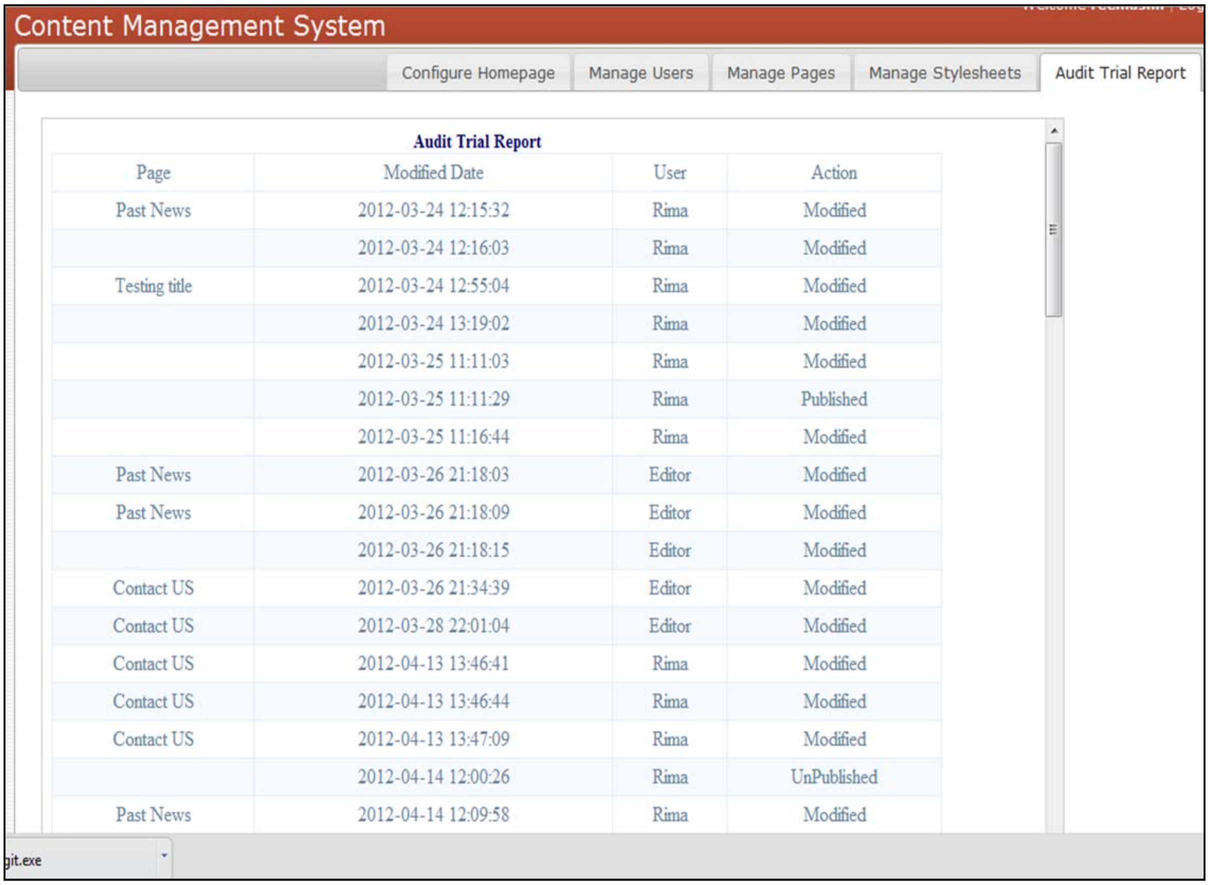
7 files

javascript:void(0)

**Figure A.3. File manager (CKFinder integrated with CKEditor).**

### Audit Trial Report:

Figure A.4. shows the screenshot of the Audit Trial Report that publishes every transaction that takes place in any of the pages in the website. The transactions can be adding a new page, modifying/deleting an existing page or publishing/un-publishing the page.



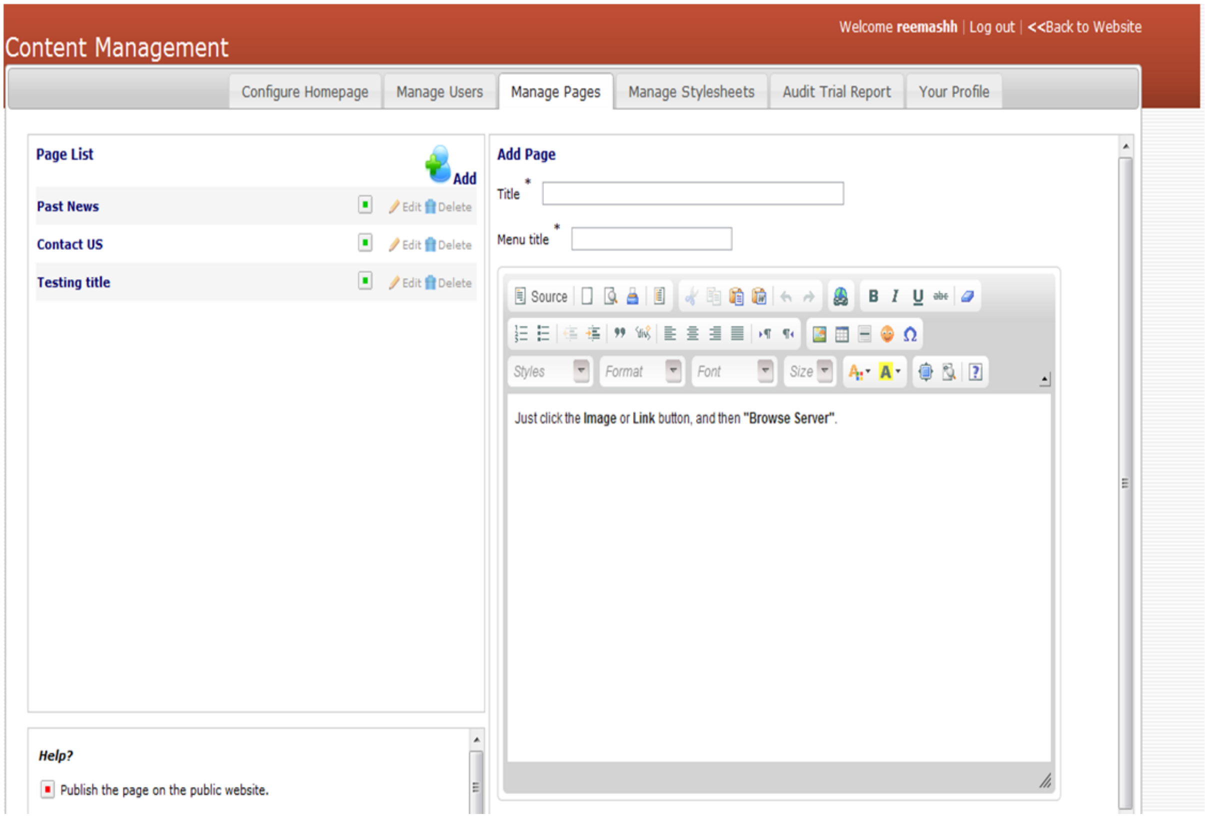
**Audit Trial Report**

| Page | Modified Date | User | Action |
|---|---|---|---|
| Past News | 2012-03-24 12:15:32 | Rima | Modified |
|  | 2012-03-24 12:16:03 | Rima | Modified |
| Testing title | 2012-03-24 12:55:04 | Rima | Modified |
|  | 2012-03-24 13:19:02 | Rima | Modified |
|  | 2012-03-25 11:11:03 | Rima | Modified |
|  | 2012-03-25 11:11:29 | Rima | Published |
|  | 2012-03-25 11:16:44 | Rima | Modified |
| Past News | 2012-03-26 21:18:03 | Editor | Modified |
| Past News | 2012-03-26 21:18:09 | Editor | Modified |
|  | 2012-03-26 21:18:15 | Editor | Modified |
| Contact US | 2012-03-26 21:34:39 | Editor | Modified |
| Contact US | 2012-03-28 22:01:04 | Editor | Modified |
| Contact US | 2012-04-13 13:46:41 | Rima | Modified |
| Contact US | 2012-04-13 13:46:44 | Rima | Modified |
| Contact US | 2012-04-13 13:47:09 | Rima | Modified |
|  | 2012-04-14 12:00:26 | Rima | UnPublished |
| Past News | 2012-04-14 12:09:58 | Rima | Modified |

**Figure A.4. Audit Trial Report.**

**Manage Pages:**

Figure A.5 shows the screenshot of the "Manage Pages" section. The administrator will have access to perform all actions on the pages. This section uses CKEditor, as shown in Figure A.2, in order to manage the content formatting without having to deal with HTML/CSS code.



| Page List | | |
|---|---|---|
| Past News | Edit | Delete |
| Contact US | Edit | Delete |
| Testing title | Edit | Delete |

Just click the **Image** or **Link** button, and then **"Browse Server"**.

Publish the page on the public website.

**Figure A.5. Manage pages section.**

## Manage Users:

Figure A.6 shows the screenshot of “Manages Users” section. The admin user can add a new user, modify an existing user’s profile, or delete a user. Any user can modify her or her own profile from the “Your Profile” section.

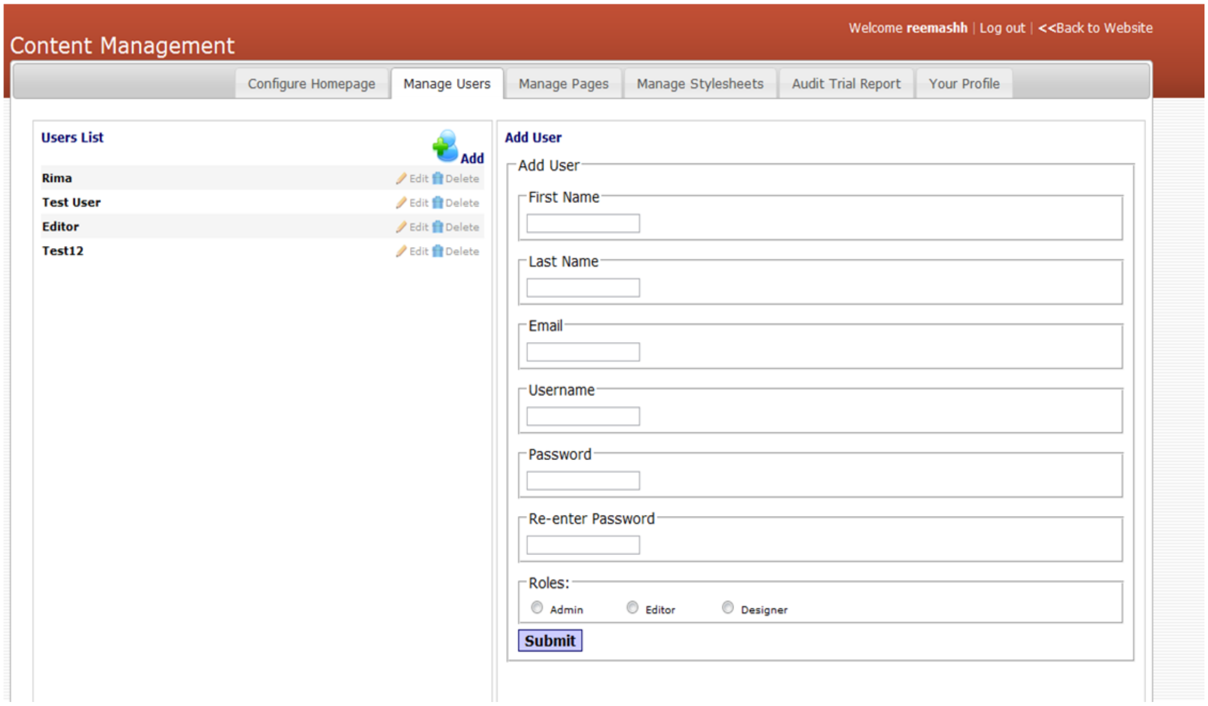


**Figure A.6. Manage users section.**