

California State University, San Bernardino

CSUSB ScholarWorks

Theses Digitization Project

John M. Pfau Library

2011

Trigeiawriter: A content management system

Mohan Prabhakara Ram

Follow this and additional works at: <https://scholarworks.lib.csusb.edu/etd-project>

 Part of the [Databases and Information Systems Commons](#)

Recommended Citation

Ram, Mohan Prabhakara, "Trigeiawriter: A content management system" (2011). *Theses Digitization Project*. 3331.

<https://scholarworks.lib.csusb.edu/etd-project/3331>

This Project is brought to you for free and open access by the John M. Pfau Library at CSUSB ScholarWorks. It has been accepted for inclusion in Theses Digitization Project by an authorized administrator of CSUSB ScholarWorks. For more information, please contact scholarworks@csusb.edu.

TRIGEIAWRITER: A CONTENT MANAGEMENT SYSTEM

A Project
Presented to the
Faculty of
California State University,
San Bernardino

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
in
Computer Science

by
Mohan Prabhakara Ram

September 2011

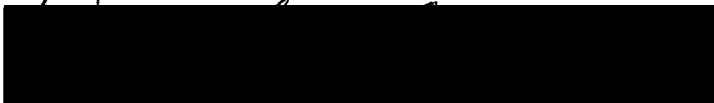
TRIGEIAWRITER: A CONTENT MANAGEMENT SYSTEM

A Project
Presented to the
Faculty of
California State University,
San Bernardino

by
Mohan Prabhakara Ram

September 2011

Approved by:



David Turner, Advisor, Computer Science
and Engineering

6/13/2011
Date



Richard J. Botting



Ernesto Gomez

© 2011 Mohan Prabhakara Ram

ABSTRACT

TrigeiaWriter is a Content Management System (CSM) for Trigeia.com, a web based magazine site. The purpose of this master's project is to design and implement a CMS. Since TrigeiaWriter is used for a web based magazine, it incorporates different roles for the users and these roles are authors, editors, and administrators. Authors create the content for the web site, which can range from technology based story/opinion to entertainment gossip. This content needs to be reviewed by the editor before they can be published on the site. The administrator is in charge of maintaining the site; this consists of creating users, giving them roles, change their status, such as making them inactive or deleting them entirely.

ACKNOWLEDGEMENTS

I would like to acknowledge and thank Dr. David Turner for his help, support, and guidance throughout this project. Also it was generous of him to share his knowledge and input; Dr. Richard Botting for his extremely informative input and sharing his knowledge; Dr. Ernesto Gomez for knowledge and input.

TABLE OF CONTENTS

ABSTRACT	iii
ACKNOWLEDGEMENTS	iv
LIST OF FIGURES	vii
CHAPTER ONE: INTRODUCTION	
Project Overview.....	1
Project Scope	2
Compatibility	3
Tools and Software	4
Project Limitations	5
Standards Compliance	5
Acronyms, Abbreviations, and Definitions	7
CHAPTER TWO: OVERVIEW OF TRIGEIA WRITER	
Overview	12
Administration Section	12
Public Site	14
Install Process and Restore	16
Architecture	19
Database Structure	24
Includes	27
Plug-ins	30
Really Simple Syndication	31

File Transfer Protocol	32
Backup	33
CHAPTER THREE: TECHNOLOGIES	
Server-side Technologies	35
Client-side Technologies	37
Search Engine Optimization	40
CHAPTER FOUR: FUTURE WORK AND CONCLUSION	
Future Work	43
Mobile Site	43
Hyper Text Markup Language 5	44
No Structured Query Language	49
Conclusion	51
REFERENCES	52

LIST OF FIGURES

Figure 1.1. Article Keyword Relationship	2
Figure 1.3. Error Message for Older Browser	4
Figure 2.2. Menu Structure	13
Figure 2.3. Public Site	15
Figure 2.4. Install Welcome Screen	16
Figure 2.4.1. Basic Information	17
Figure 2.4.2. List of Restore Files	18
Figure 2.5. Model View Controller	19
Figure 2.5.1. Code Structure that Handles the Pages	20
Figure 2.5.2. Code Structure of How Options Work in TrigeiaWriter	22
Figure 2.5.3. Site Layout	24
Figure 2.6. View All Post by Authors SQL Query	25
Figure 2.6.1. Data Dictionary	27
Figure 2.7. Code for sending Out a notice	28
Figure 2.7.2. Configure.php	29
Figure 2.8. Plug-in Architecture	30
Figure 2.8.1. Admin Component Structure	31
Figure 2.9. RSS Icon	31
Figure 2.10. Code that Uploads User's Avatar to a Particular Directory	33

Figure 2.11. Backup Code	34
Figure 3.1. Administrators are Able to See the Inactive Users	35
Figure 3.1.1. Non-Administrators are Greeted With an Error Message	36
Figure 3.1.2. SQL Code to Retrieve Users From the Users' Table	36
Figure 3.2. JavaScript Drop Down Menu	37
Figure 3.2.1. jQuery Code for New User Form	38
Figure 3.2.2. jQuery Validation Form	39
Figure 3.2.3. CSS Code that is Used for Body Text in a Web Page	40
Figure 4.1.1. Mobile Site Mock-up	44
Figure 4.2.1. Required Filed for Input Box in HTML5	45
Figure 4.2.2. HTML 5 Required Field	46
Figure 4.2.3. HTML5 Code for Checking Email	46
Figure 4.2.4. HTML5 Code for Checking URL	47
Figure 4.2.5. HTML 5 Email Validation	48
Figure 4.2.6. Auto Focus in HTML 5	48
Figure 4.2.7. Placeholder in HTML 5	49
Figure 4.3.1. Sample NoSQL Command	50
Figure 4.3.2. Sample Output of NoSQL	50

CHAPTER ONE

INTRODUCTION

Project Overview

TrigeiaWriter is a content management platform (CMS) that is built for Trigeia, a small Internet start-up. In order to make TrigeiaWriter, WordPress blogging engine was chosen as the CMS; as the project work commenced it was evident that not everything could be done according to the software specification/business proposal. A decision was made to stop using WordPress and create a custom CMS from scratch that would meet all the requirements. TrigeiaWriter is programmed in PHP server-side scripting language, jQuery (JavaScript engine) for client-side validation, and SQL is used as the database language to interact with MySQL database. Web pages are constructed by using HTML and CSS. Since TrigeiaWriter is a web program there the user can run any environment on their computer, as long as they have a standard compliant web browser.

TrigeiaWriter is specifically built from the ground up for Trigeia's web based magazine site. TrigeiaWriter has been built with scalability in mind, this means it can be used for not just web based magazines, but also for a corporate site. TrigeiaWriter has two components to it, administrative section and public site. The administrative section is where the content is created, managed and maintained. The public site is viewed by site visitors. Each article that is created

has an associated keyword (tag) attached to it, so that they are sectioned off and organized by these keywords. An example of how many keywords can be associated to articles can be seen in Figure 1.1.

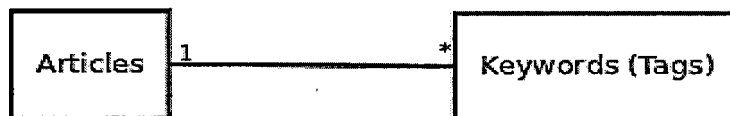


Figure 1.1 Article Keyword Relationship

Keywords can be created, and edited if needed. Since this is a magazine based site, there are three types of roles for site contributors. These three roles play a major part of the site. These roles are *author* who creates the content, the *editor* who is in charge of editing the content, and the *administrator*, who manages the site.

Project Scope

The project scope consists of the architecture, look & feel, and implementation of the CSM using Model View Control (MVC) architecture. There are many advantages of using MVC, for instance if a part (such as the look & feel) of the architecture is swapped out then the whole CMS will still function normally. Another advantage for using MVC is the whole project is structurally organized. So each component (for example *view-posts*) are broken up into view

and task. Also within a task there can be a sub category called option, which is used to pass arguments. For example a file, such as *user_status.php*, can be used for not only deleting a user, but also to put that user into an inactive state. By passing *option=0* this will invoke delete user, but with *option=1* will put the user in an inactive state. This will greatly reduce the repetition of code.

The look & feel can be changed with minimal effort. Only a few pages of code need to be updated in order for a new theme to take place. The web application is programmed in an adaptable fashion. Meaning that it does not have to be used just as a web based magazine, but it could also very well be used for other things, such as a weblog (blog), news site, or just general all purpose web site. The public site can have its own theme, so the site can look like a generic blog, a magazine, a newspaper just to name a few.

Compatibility

Since TrigeiaWriter is a web application, it is platform agnostic; this enables it to run on any computer as long as few requirements are met. In order to have TrigeiaWriter run successfully, the computer needs Apache 2.x web server (but any web server is sufficient, such as lighttpd for instance), PHP 5.5, and MySQL 5 database server. Included in the installation process is a version checker, which will display a green check mark if the versions match, or a red x-mark if they do not. On the client side the user needs to have a modern web browser installed. TrigeiaWriter incorporates newer technologies, such as jQuery

JavaScript engine for form validation. The drop down menus found in the administration portion of the site is done with JavaScript. It is highly recommended that a newer web browser should be used, as older web browsers such as Internet Explorer version 6 might not have the support for certain CSS elements and Javascript. If an obsolete browser is used on the site, then a message is printed as shown in Figure 1.3.

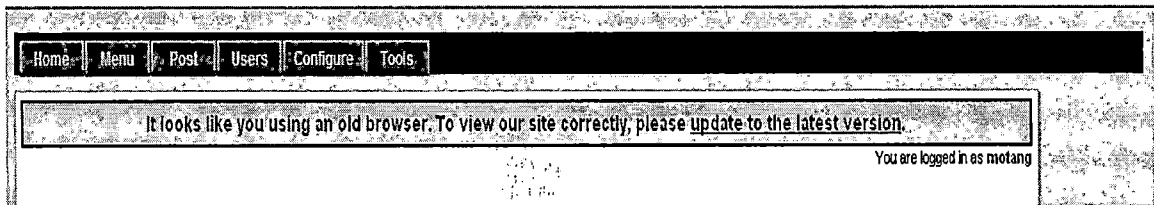


Figure 1.3 Error Message for Older Web Browser

Tools and Software

TrigeiaWriter is written in the PHP web programming language, which does the server-side scripting. The client-side rendering is done with HTML, CSS and jQuery. The name jQuery represents a JavaScript library, which TrigeiaWriter uses for validating the information being inputted. For data query SQL is used and primarily developed in the Ubuntu (a variant of Linux) platform using Geany IDE. MySQL Query Browser was used as a GUI front-end for SQL queries. MySQL Server Administrator was used as a GUI front-end to manage the database and its tables.

Project Limitations

The web application will not work the way it was intended if the user is not using a modern browser. Older versions of browsers will not render the site properly as they do not have support for newer version of CSS. The older version of JavaScript library that ships with these older browsers will also hinder jQuery as they do not have support for it. That being said, the site is still usable as it will load, and visitors can still read the stories, but it would not look the way it was intended. If for some reason, JavaScript validation is not working (in case of older web browser or the user has JavaScript turned off) then server-side validation is used.

The majority of the work is done on the server, so if the server happens to go down, then the website will be inaccessible. Also it is critical to make sure that the web server meets the requirements to have TrigeiaWriter run at optimal speed.

Standards Compliance

TrigeiaWriter has been programmed with W3C's standard guideline for building web pages using HTML [8], as well as web page accessibility policy [11]. With W3C's standard guideline, web applications run and look the way they are intended. With a standard compliant browser TrigeiaWriter will look the same, irrespective of the platform it is running on. This means TrigeiaWriter is written in XHTML instead of HTML, certain tags such as the image tag (``) is

closed off with '>' as it does not have a closing tag of its own in regular HTML. TrigeiaWriter is programmed to adhere web page accessibility policy, it is programmed to allow people with disability can navigate the public site without any problem [11]. Here are some of the guideline which states that a site must have to be accessible. All images should have an *alt* attribute, the site must support font resizing up to 3 times the default size, if tables are used on the site they must contain *label* tags so that a screen reader can navigate. These are just a few of the guidelines, more can be found at California State University, San Bernardino's web page accessibility site [11].

Acronyms, Abbreviations, and Definitions

- 404 error: 404 error or not found is a error code that is generated by the server when a requested document is not found on the server.
- Apache HTTP Server: Web server software that is made by the Apache Foundation
- Back-end: This refers to server-side languages like PHP, where all the computation is done on the sever itself and then the result is then shown to the user
- CMS: Content management system, a cohesive web application that is used to manage and create content.
- Codec: Stands for compression/decompression, used for video and audio files, where the file is compressed by a program with a certain technology and the appropriate player has that technology to decompress it and play back the video or audio file on the user's computer.
- CSS: Cascading Style Sheet, is used to make the look and feel of a web pages. So this mens sections of the pages can be placed in the center or the right. Is used to define how a web page will be presented to the visitor to the web site.
- Database: Software that is loaded on a computer that holds data that is input using SQL
- Database Dump: the database dump contains a record of the table structure and/or the data from the database.

- **Feed readers:** Programs that are used to access or RSS feeds. This can range from Desktop applications such as Thunderbird Mozilla Mail/News client to web based reader like Google Reader.
- **Front-end:** Refers to client-side languages like JavaScript, where the users' computer is used to make the computation
- **GUI:** Graphical User Interface
- **HTML:** Hyper Text Markup Language, used to markup web pages. Everything from bold to italic to new paragraphs are done using this language.
- **jQuery:** A cross-browser JavaScript library for JavaScript that used for real time events on the client's computer.
- **Machine-readable:** Is way of storing content that can be accessed by a machine and can be turned into binary.
- **MD5:** Message-Digest algorithm 5, is a widely used cryptographic hash function with a 128-bit hash value.
- **MVC:** Model View Controller is an architecture where there are three separate things but are interchangeable. The pattern isolates the domain logic from user interface, allowing an independent development, testing, and maintaining the project.
- **MySQL:** A rational database management system (RDBMS) that runs as a server providing multi-user access to number of databases. It is made by MySQL AB company

- NoSQL: A database system that differs from the traditional relational database management systems. No tables are associated with the data, and join operations are avoided and usually scales horizontally.
- PHP: PHP Hypertext Processor, is the server-side scripting language that is used to do the heavy lifting from talking to the database with embedded SQL statements, to getting the data and inputting that in the database and also retrieving it and displaying it to the users.
- Plug-ins: a collection of software components that will add features to a particular software application.
- RDBMS: Rational database management system is a database system that is based on rational model that was defined by E. F. Codd.
- Regular expression: regular expression provides a concise and flexible means for matching strings of text, such as words, characters, or pattern of characters.
- Robots.txt: Is a common protocol that prevents cooperating web spiders and other web crawlers from accessing parts of a website.
- RSS: Really Simple Syndication, is a web technology that used to published frequently updated works in a standard format (XML). These feeds can be accessed with Feed Readers such a Google Reader or RSSOwl.

- SEO: Search Engine Optimization is a way to format URL so that it would be easy for search engines such as Google to crawl the web site and index it.
- SQL: Structured Query Language is used to insert, retrieve, and manipulate the data that is stored in a database.
- User Agent: an implementation that is used in communications with the server to identify a computer system.
- URL: Uniform Resource Identifier is what specifies where an identified resource is available and the mechanism for retrieving it.
- W3C: The World Wide Web Consortium is the main international standard organization for the World Wide Web.
- W3C: World Wide Web Consortium is the main international standards organization for the World Wide Web.
- Web browser: Software that is used for retrieving, presenting, and traversing information resources from the World Wide Web.
- Web server: Software that is loaded on a computer that servers up web pages upon request.
- Web spiders: Also known as web crawlers is a computer program that browses the World Wide Web in an automated manner. They are used to gather data, index websites for search engines, etc.

- **Web Crawler:** A computer program that surfs the Internet and indexes it in orderly fashion.
- **WordPress:** Open source content management system developed by a company that goes by the same name.
- **XHTML:** eXtensible HyperText Markup Language, is a family of XML that extend HTML.
- **XML:** Extensible Markup Language is a set of rules for encoding a document in a machine-readable form.

CHAPTER TWO

OVERVIEW OF TRIGEIAWRITER STRUCTURE

Overview

There are two parts to TrigeiaWriter, similar to that of other content management system (CMS). These parts are “admin” (administration section) and “public.” The administration part of TrigeiaWriter is very important. This is where the content is created and the public part is where the content is consumed by site visitors. TrigeiaWriter has an automated install process similar to that of WordPress, where only a few key information is required by the user and rest of the task is handled automatically. TrigeiaWriter has been built from the ground up to incorporate MVC architecture, where the view is separate from the task.

Administration Section

The administrator part of TrigeiaWriter is very important. This is where content, users, and menu items are created. Within the administration section there are options to create a backup of the database, configure RSS feed, and plug-ins can be enabled or disabled. The UI for Administration is kept simple. All of the navigation is at the top with drop down menus. By having the menu at the top, this gives more screen real estate for the main content, as shown in Figure 2.2.

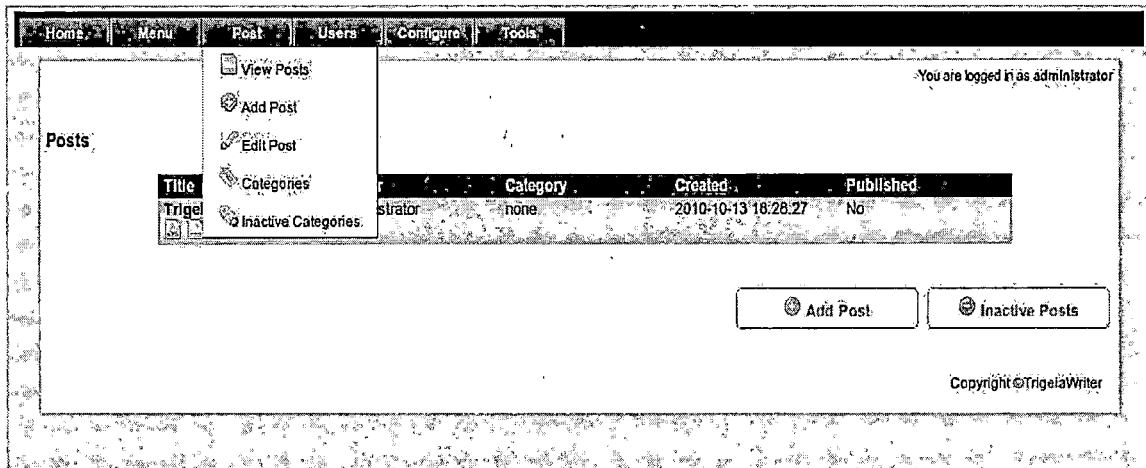


Figure 2.2 Menu Structure

Each major section of the administration is broken down into its own menu. These sections are menu, posts, users, configure and tools. In menu section menu items are created and managed. These menu items are shown on the public site, for example, a contact page. The post menu contains links to create and manage articles, as well as categories. One important thing about the posts (articles) is that they cannot be deleted, but only made inactive, this is done to keep the posts in archive mode as these content are vital for the site. Categories can be deleted but it is encouraged not to, because articles are filtered with categories. If a particular category is deleted then the articles that depend on it are orphaned. By default two categories are created upon installation, which are system and menu. These are important categories and the user does not have access to them. System category is used as a default category when posts without any specific category is selected are in saved

status. Menu category is used to generate the menu on the public site.

Next item in the menu is users; this is where users are created, edited and even deleted. By default the administrator is created by the system upon installation. This account cannot be deleted and only certain aspect of it can be edited. Users are not deleted upon first clicking on the delete button, instead they are placed into an inactive state. Users' status can be changed back to active status by an administrator or can be deleted completely from the database. Also only users with administrative privileges can edit other user accounts, and as well as create and delete users.

Configure menu item is where the option for RSS, plug-ins, and setup is found. The RSS can be turned on or off as well as the option to have short or long RSS feeds. Plug-ins is used to install, remove, enable and disable plug-ins. Setup option is a dummy link, it is used to give information to the administrator to go to the installation process and choose the restore option.

Public Site

The Public site is viewed by site visitors, as shown in Figure 2.3. Upon visiting the site, the visitors views the homepage, which has summaries of the latest articles. From there visitors can choose which article to read (there will be a list of most recent articles from each category), or they can choose to visit a particular category from a drop down menu. Also visitors can search for an article with a built in search engine. If the social plug-in has been enabled then

after reading the article, visitors have the choice of sharing the article via popular social networking sites, such as twitter, facebook, etc. to promote an article they liked.

Sunday, August 2, 2009 / 790 Users online Simple Magazine Network site Network site 2

Next size

Simple Magazine / News

News | Politics | Culture | Sport | Debate | Entertainment

Frontpage | Style Demo | Two Columns | Single Column | Archives | Empty Page

Template Information

11:51: Saturday, August 2, 2009 by Viktor Perissov



This is a free website template by Arcsin, built using tableless XHTML and CSS.

This template is distributed under a Creative Commons Attribution 2.5 License, which allows you to use and modify it for any purpose (personal and commercial), under the condition that you keep the provided credit links in the footer.

The latest template version and CMS conversions for platforms such as Wordpress and Blogger can be found at the official Simple Magazine website template page.

For more templates, questions and comments please visit Arcsin Web Templates.

[Read more »](#)



Curabitur justo arcu
Bibendum ac bibendum in
semper nec bibi

Donec sodales
Duis risus lectus, gravida
eu scelerisque.

Donec rhoncus
Donec eget congue,
libero.

Article title



Integer diam elit,
condimentum ac semper ut,
tincidunt non diam. Ut congue
rutrum justo ac commodo.
Aenean euismod tincidunt
lorem scelerisque euismod.

[Read more »](#)

Second Article Title



Sed congue facinia leo, sed
dignissim odio pharetra vel.
Fusce dignissim dui. Fusce
semper porttitor enim dapibus
venenatis.

[Read more »](#)

Third title

Sed auctor hendrerit eros eu
eleifend. Cras hendrerit laculis
andales. Phasellusque
interdum rhoncus magna.

[Read more »](#)

Latest News

Aenean tempor arcu.	20:49
Juste interdum rutrum	20:49
In nec justo in urna	19:55
Accumsan condimentum	19:55
Etiam conmodo bibendum	19:55
Mauris euismod justo	19:51

[Browse all »](#)

Most Viewed

1. Integer diam elit
2. Condimentum ac semper
3. Tincidunt non diam
4. Ut congue rutrum
5. Enim dapibus venenatis
6. Cras hendrerit laculis
7. Duis in lectus
8. Eleifend nectortor

[Browse all »](#)

Network News

Nullam eros
Eleifend nectortor
Duis in lectus
Integer diam elit
Enim dapibus venenatis
[Visit Network Site »](#)

About Simple Magazine

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

[Learn more »](#)

Follow Us (RSS)

- ◉ Lorem ipsum
- ◉ Dolor sit amet
- ◉ Consectetur
- ◉ Adipiscing elit

Help & Support

Quam velit dapibus quam, ornare
suscipit tector digni ut tellus.

[Frequently Asked Questions \(FAQ\) »](#)

Get in touch

Phone: +45 7152 5412
Email: info@simplemagazine.com

[Online contact form »](#)

© 2003-2009 Simple Magazine → Home | News | Politics | Culture | Sport | Debate | Entertainment

Website template by Arcsin

Fig 2.3 Public Site

Install Process and Restore

The install process for TrigeiaWriter is fast and simple; in total it is only couple of steps. It can be invoked by going to the site URL with */install* following it. The "Welcome to Install" page has two choices, if it's a first time install then it is urged for the administrator to go on to the second screen, if the site needs to be recovered then the recovery option can be chosen which will initiate the recovery process, as shown in Figure 2.4.

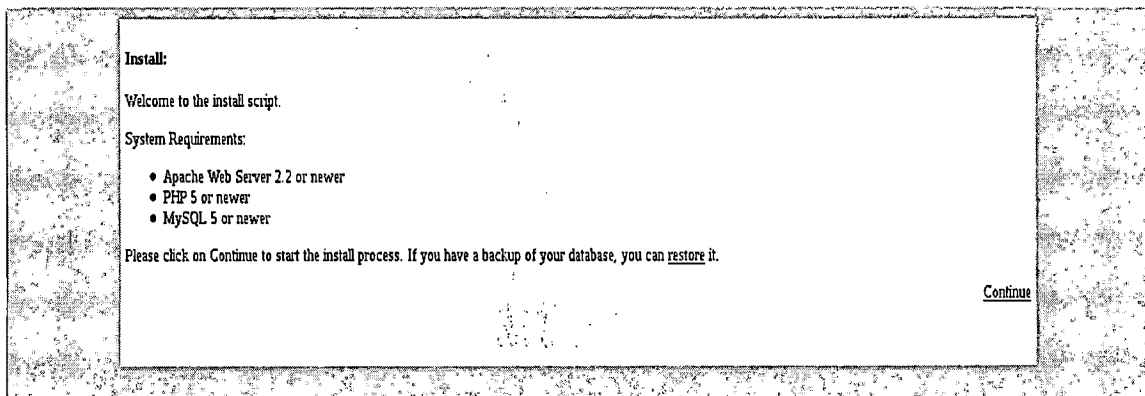
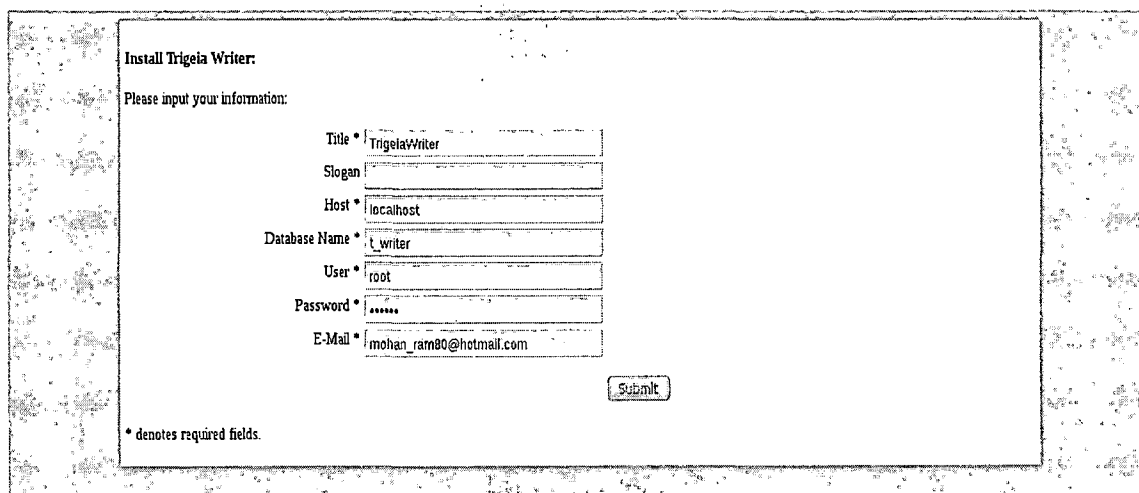


Fig 2.4 Install Welcome Screen

Both the install and recovery processes need basic information such as the name of the site, MySQL server address, user name and password. Email is also needed for the administrator (as this is used for the contact us form on the public site), since the administrator also plays the role of the web master. After the user specifies this information, the install process creates the database tables and views that are used by TrigeiaWriter, as shown in Figure 2.4.1. These basic

information are saved in a *config* file, if this file is not found in a particular directory then it is created and the appropriate data is written on to it. This step is done in the background and if the administrator chooses to he can edit the config file in any text editor.



The screenshot shows a web form titled "Install Trigeia Writer:". Below the title is the instruction "Please input your information:". The form contains several input fields, each with an asterisk indicating it is required. The fields and their values are: Title (TrigeiaWriter), Slogan (empty), Host (localhost), Database Name (t_writer), User (root), Password (masked with asterisks), and E-Mail (mohan_ram80@hotmail.com). A "Submit" button is located at the bottom right of the form. A note at the bottom left states "* denotes required fields."

Figure 2.4.1 Basic Information

The recovery process is very simple. After the basic information has entered, a screen with all the back-ups is shown and they are listed by date with the latest at the top and the oldest at the bottom (refer to Figure 2.4.2). Once the desired back-up is chosen, the automated system will restore the data into the database and the system will be restored to what it was on that particular date the back-up was done. Note, a back-up must be done through the administration

section regularly to ensure that the latest back-up can be used in case the data needs to be restored.



Figure 2.4.2 List of Restore Files

Architecture

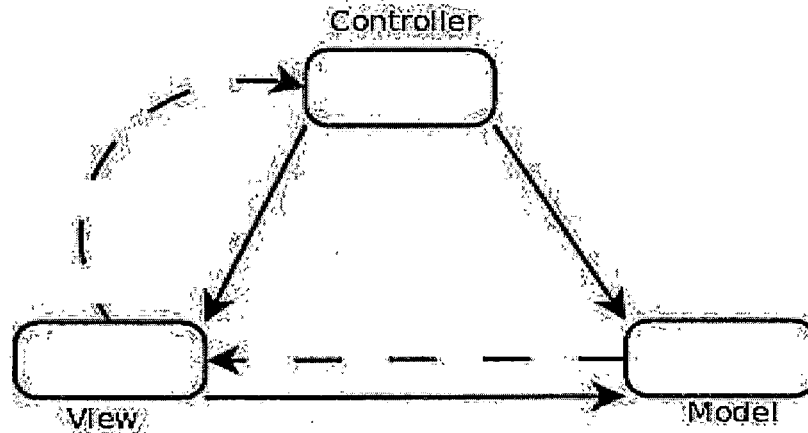


Figure 2.5 Model View Controller

The architecture of TrigeiaWriter is setup in Model View Controller (MVC), as seen in Figure 2.5. For example, a component that adds new user to the database is split up into that architecture. This means the forms that are used to add information about the user is placed in the *view* folder, but all the back-end process such as creating the user and adding the data into the database is placed in the *task* folder. TrigeiaWriter incorporates its own internal URL scheme. Each component can have its own folder but is required to have *index.php* in that folder. This *index.php* is what controls everything. From the *index.php* file views and tasks are parsed. Also *index.php* file generate a 404 page not found error. This function can be seen in Figure 2.5.1.

```

if (isset ($_GET['view']))
{
    $page = $_GET['view'];
    if (file_exists("view/$page.php"))
        include_once ("view/$page.php");
    else
        include_once ("../includes/404.php");
}
else if (isset ($_GET['task']))
{
    $page = $_GET['task'];
    if (file_exists("task/$page.php"))
        include_once ("task/$page.php");
    else
        include_once ("../includes/404.php");
}
else
    include_once ("view/default.php");

```

Figure 2.5.1 Code Structure that Handles the Pages

The view and task are folders within a particular component. The view folder is required to have a file called *default.php*. This file is used as the default page upon the first visit. See “else” statement in the above code as an example. The purpose of the view is to construct the web pages for the user’s browser. For example, a view might include a form to add a new user. The task is where all the business logics are performed. The PHP script for adding a new user into the database is done via the task.

Specific options can also be passed on through option variable, which is done by the task. For example take the file called *user_status.php*. Within this file are options that can be used to modify the user status. For instance an option could be passed on to make the user inactive, and another one can be

used to delete the user, or even make the user active again, an example shown in Figure 2.5.2.

```
$userID = intval($_GET['id']);
#option = intval($_GET['option']);
if($userID != 1)
{
    #this is the actual SQL statement that inactivates or activates the use
    if($option == 0)
        $sql="UPDATE tw_users SET status = 0 WHERE id = '$userID' ";
    else
        $sql="UPDATE tw_users SET status = 1 WHERE id = '$userID' ";

    #this is error statement if there was any and an option for the user to go
back and correct that mistake
    if (!mysql_query($sql,$conn))
    {
        message("notice", "Something went wrong, please try again later");
        echo ('<center><a href="index.php?view=users" class="button">Go
back</a> ');

        //die("error: " . mysql_error()); //uncomment for debug purpose
    }
    else
    {
```

```

if($option == 0)
{
    message("notice", "User has been deleted");
}
else
{
    message("notice", "User status has been changed");
}
include_once ('view/users.php');
}
}

```

Figure 2.5.2 Code Structure of How Options Work in TrigeiaWriter

TrigeiaWriter is structured in a way that only the core aspects of a component need to be programmed. This means only a portion of the page needs to be coded (the main content of a page). Since the header and footer are consistent throughout the site, they are called from an external source that is located in the *includes/template* directory.

With this structure repetition of code is cut down, and codes that are used many times are centralized. Another advantage of this structure is, if a change needs to be done, then it can be done in one place and the changes will take effect globally. Using global changes are not encouraged, but in this case this is the template of the whole project. Nothing major related to variables, or settings

etc. are not global, those are more localized to particular components. So basic HTML code is placed in a central place directory called *template*, with php code that separates the HTML code into three sections. These sections are header, `upper_html` which consists of menu for the administration section, and `lower_html` which has the footer, as seen in Figure 2.5d. The reason the header and `upper_html` are separated because not all administration section need the top menu (for example the log-in page).

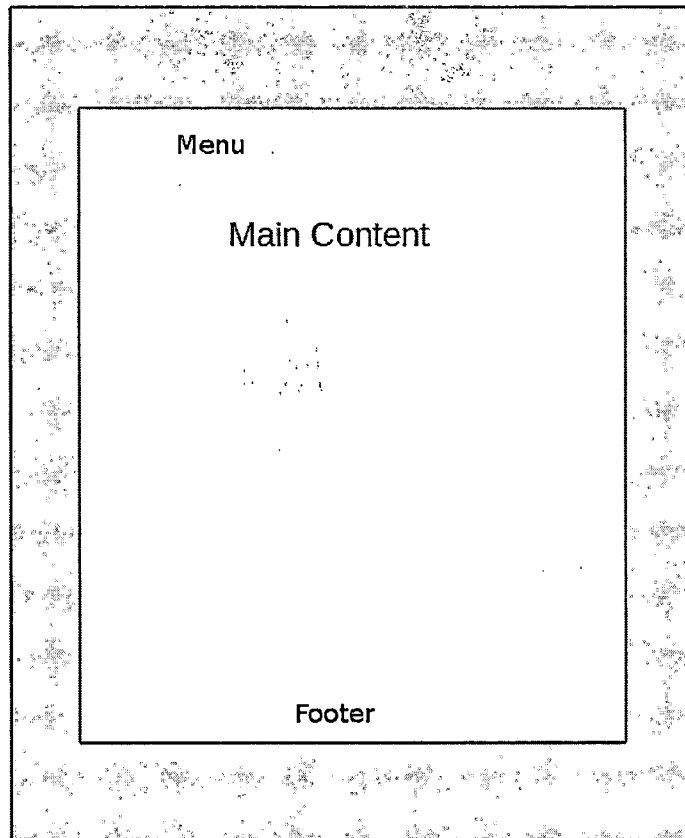


Figure 2.5.3 Site Layout

Database Structure

TrigeiaWriter's database structure is setup so that all major components of TrigeiaWriter has it's own database tables. This means plug-ins, articles, other components have their own database tables. Because of this structure each major part of the CMS is self contained and can be altered without having to worry if a change will effect anything else. Depending on the query, data can be pulled from tables and mashed up together to present a view to the user. One such query is the view that used to show the articles, who the author was, if it is published and publication date.

There is also a log table which is used to log major transactions that are made. For instance when a new author is created, then that gets logged with a time stamp and who initiated the process. This is done to keep a log and if needs to be audited for some reason it can easily be done. The log table is populated by a script that is included in key files, and every time that file is run it also calls particular functions, so when a new user is created then the new user script for log is called, that function enters in the time, and what was done to the database and by who, as shown in Figure 2.6.

```
CREATE VIEW v_posts AS
    SELECT tw_articles.id AS aid, tw_articles.title,
    tw_articles.time_stamp, tw_articles.category, tw_articles.published,
    tw_articles.uid, tw_users.id AS usrid, tw_users.username, tw_category.catId,
    tw_category.categories
    FROM tw_articles, tw_users, tw_category
    WHERE tw_articles.uid = tw_users.id AND tw_articles.category =
tw_category.catId
    ORDER BY time_stamp ASC
```

Figure 2.6 View All Post by Authors SQL Query

Data dictionary has been structured to scale, more tables can be added to it if needed. In Figure 2.6.1 shows the data dictionary along with the views. Views

are used to store complicated queries such as the one in Figure 2.6a. By implementing views the core PHP code can be kept minimal when it comes to embedding the SQL code and also makes it easier for future use as it's just one line of code and for more in-depth view the database can be referenced. Also once views were implemented for complex queries, PHP pages with those queries seems to load faster.

User database contains a column called password. Password that is entered into the database is encrypted with MD5 hash function. On top of MD5, salt is added which is a random generated characters (up to 26). The MD5 and salt are separated by a ':' and this is taken into account with an algorithm to decrypt the password whenever a user needs to log into the system.

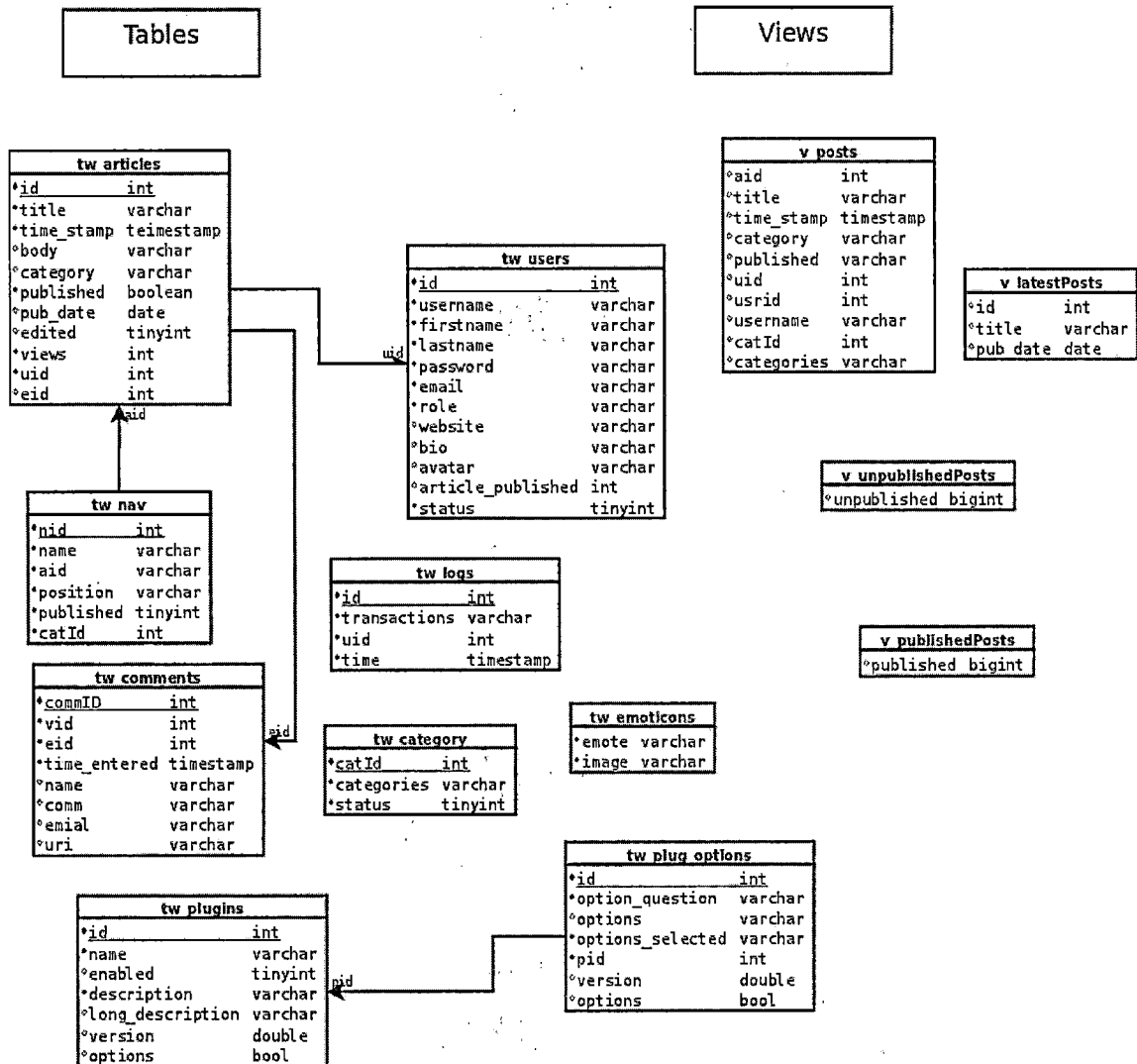


Fig 2.6.1 Data Dictionary

Includes

Includes is a core folder that contains many files that are used repeatedly by TrigeiaWriter. For example, a file called *connect.php* is in this folder that is used to connect up to the SQL database. A folder called *template* is in this

directory that is used to make the common structure and layout. The includes folder also has the error page that is generated when a page is called but does not exist, this can be seen in lines 7 and 13 in Figure 2.5.1.

TrigeiaWriter also has the capability to generate error messages. This is used when something goes wrong, or something is not found, or just to alert the user. For example the 404 error can be seen in Figure 2.7.

```
message('notice', '404 - Something went wrong, the page you have been looking  
for isn\'t here');
```

Figure 2.7 Code for Sending Out a Notice

The example illustrates a message called notice, it is passed on and printed out on the screen. There are three types (or level of severity) of messages. These messages are notice, message, and warnings. Notice is used to give alerts to the user, but it's not necessarily very important. Message are used to send out messages to the user, for instance an SQL query ran and it was successful. Warning is usually used to warn the user when something has gone wrong, or when they tried to do something but it is wrong action. For instance, they inputted a wrong type of content like an email address without an '@', or letters into a phone number field.

Configure is a file that is created when TrigeiaWriter's install process to run, this file has important information such as the address of the database server, name of the database, and name of the tables. This file is important as it contains important information about connecting to the database, and it is suggested that this file can be edited from the install or restore process by novice users. An example of the configure file is shown in Figure 2.7.2.

```
<?php
$title = 'TrigeiaWriter';
$slogan = "";
$host = 'localhost';
$database = 't_writer';
$user = 'root';
$password = 'mohanp';
$email = 'mohan_ram80@hotmail.com';
?>
```

Figure 2.7.2 Cofigure.php

Plug-ins

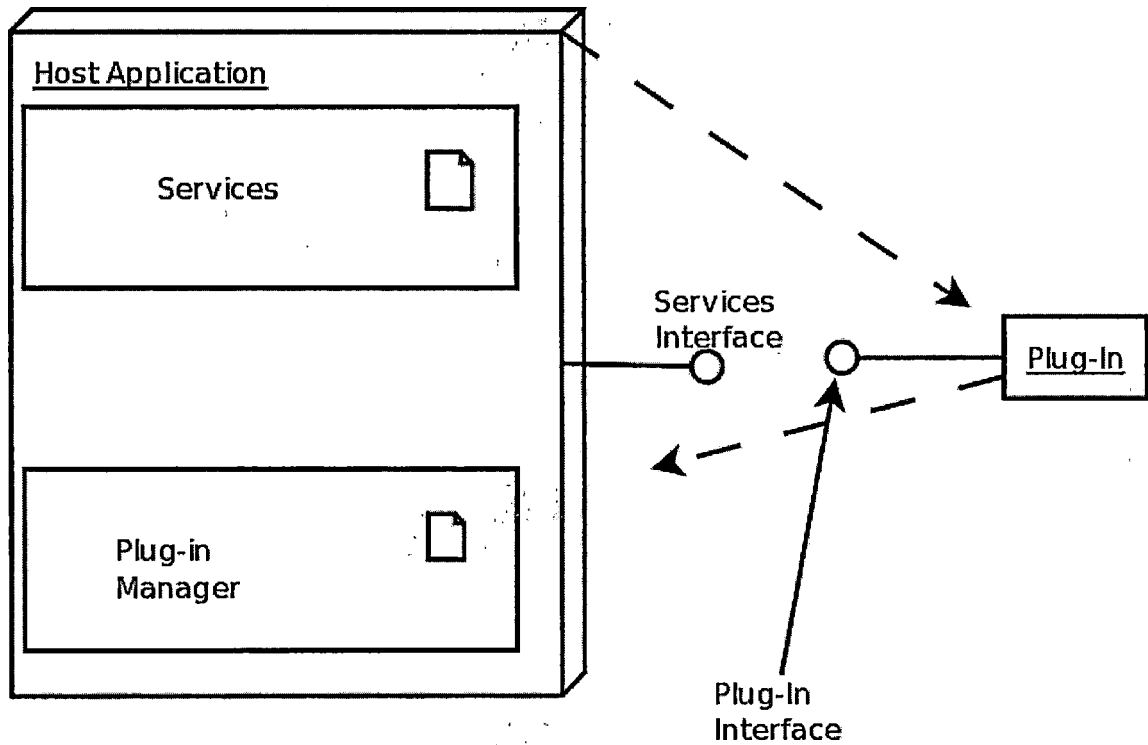


Figure 2.8 Plug-in Architecture

TrigeiaWriter has been built to support plug-ins. Plug-in structure is shown in Figure 2.8. So it has the potential to expand with new features. These features can range from having social links on the articles, or have stars to rate the quality of the article. Plug-ins are required to follow the MVC architecture. So within the folder of the plug-in a task and view folders are required as shown in Figure 2.8b.

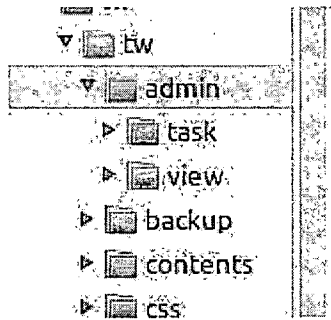


Figure 2.8.1 Admin Component Structure

Really Simple Syndication

RSS (Really Simple Syndication) is a feature that is offered by the majority of the web sites on the Internet. It is used by the site visitors to keep up-to-date on the content as they are published. These contents are pushed out to the user without having them visit the site. Majority of modern web browsers support this feature, and there are many web services that also support it. The RSS is built on XML technologies and can be generated with PHP.

With TrigeiaWriter the user has a choice to enable or disable the RSS feed. If RSS is enabled then an RSS icon will be shown with compatible browser. The icon looks similar to which is shown in Figure 2.9.



Figure 2.9 RSS Icon

File Transfer Protocol

File transfer is used to transfer files from one place to another over the Internet. TrigeiaWriter has a built in File Transfer Protocol (FTP) to upload the images, sound files, or movie files from the users' computer to the web server. The directory that holds these media files are uploaded to */images/tw-uploads*. Figure 2.10 is a sample PHP code that used to transfer avatar image.

```
$filename = $_FILES["avatar"]["name"];
    $file_basename = substr($filename, 0, stripos($filename, '.')); // strip
extention
    $file_ext = substr($filename, stripos($filename, '.')); // strip name
    $filesize = $_FILES["avatar"]["size"];

    if (($file_ext == ".png" || $file_ext == ".jpg") && ($filesize <
200000)) {
        // rename file
        $newfilename = $_POST['id'] . $file_ext;

        /*if (file_exists("../images/avatars/" . $newfilename)) {
            // file already exists error
            $error = "You have already submitted this file.";
        } else {*/
            move_uploaded_file($_FILES["avatar"]["tmp_name"],
```

```

"../images/avatars/" . $newfilename);

        //echo "File uploaded successfully.";
        $avatar = $newfilename;

    // }

} elseif (empty($file_basename)) {

    // file type error

    $error = "Please select a file to upload.";

} else {

    // file selection error

    $error = "Only jpg and png files can be submitted online.";

    unlink($_FILES["avatar"]["tmp_name"]);

}

```

Figure 2.10 Code that Uploads User's Avatar to a Particular Directory

Backup

TrigeiaWriter has a built in back-up tool, that is found under *Tools* menu. This option will automatically initiates a MySQL Dump of the database and it's content into a flat file into */backup* directory. The administrator then has the choice of backing that file up in a different location but is encouraged to leave a copy in the */backup* directory as that is the directory that is used when the Restore option is invoked.

```
$pathdump = "../backup/";  
$backupFile = $database . date("Y-m-d-H-i-s") . '.sql';  
  
$command = "mysqldump --host=$host --user=$user  
--password=$password $database > $pathdump/$backupFile";  
system($command);
```

Figure 2.11 Backup Code

The Figure 2.11 is an illustration of how the backup script is done. First the path to the backup directory is given (in future this can be made dynamic), then we have the code for the name for the backup file with current date appended on to it. Finally the MySQL command for the SQL dump that takes the content from the database and writes that out to a flat file. This can easily be placed in a secure place, and used to restore back manually or through the automated restore.

CHAPTER THREE

TECHNOLOGIES

Server-side Technologies

Server-side programming languages are used to write programs that run on the web server and not on the client's computer. For TrigeiaWriter the server-side programming languages that were used are PHP and SQL. The PHP language is the primary scripting language for TrigeiaWriter. The majority of all the server-side computation is done in this language. For example, there are occasions where certain actions such as making a user inactive can only be done with user accounts that have an administrative role. If a user is logged into an account other than administrator, then that action is not even shown. If the user happens to go that certain page (from web browser history) then they are greeted with an error message. That is shown in Figures 3.1 and 3.1.1.

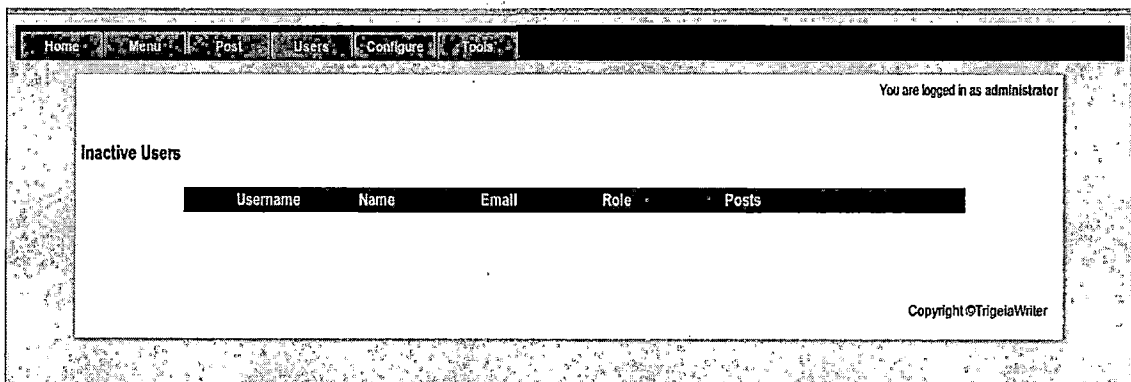


Figure 3.1 Administrators are Able to See the Inactive Users

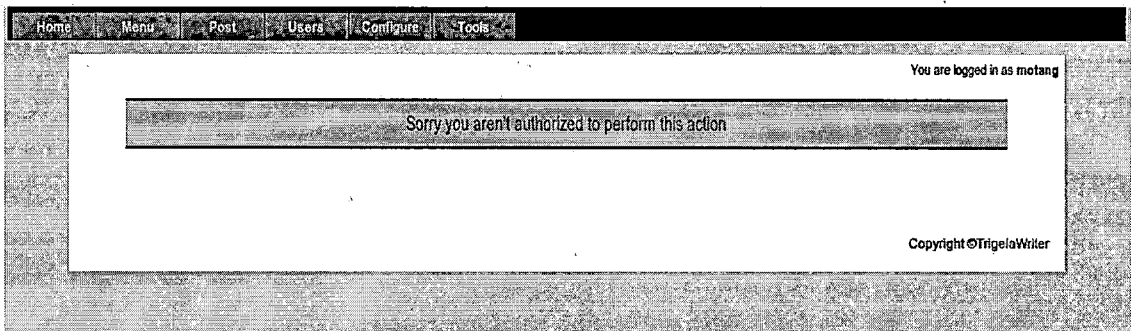


Figure 3.1.1 Non-Administrators are Greeted With an Error Message

Another example of how server-side programming language are used in TrigeiaWriter is to insert HTML code into a webpage template. This makes it easy to manage and make changes to the site, because the code is in a centralized location and used where needed by using an insert directive rather than copying code. This reduces code redundancy.

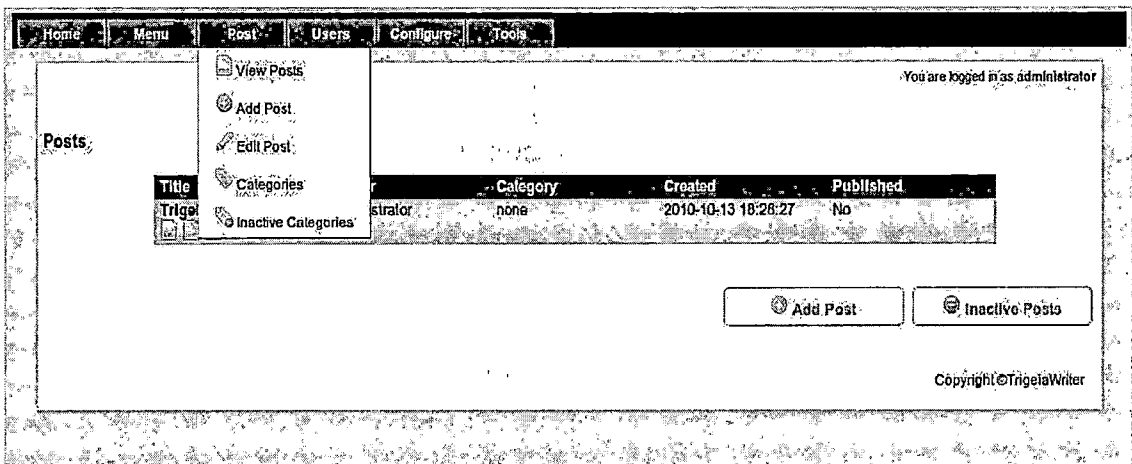
The second server-side programming language that is used for TrigeiaWriter is SQL, which is a database language. The SQL is used to insert, manipulate, and retrieve the data from the MySQL database server. This is shown in Figure 3.1.2.

```
SELECT * FROM tw_users  
  
WHERE status = 1  
  
ORDER BY username ASC
```

Figure 3.1.2 SQL Code to Retrieve Users From the Users' Table

Client-side Technologies

Client-side programming languages such as JavaScript, HTML, and CSS are processed on the client's computer. Because of this, the load is taken off the web server. For example the drop-down menu in the administrator section is done with JavaScript, as shown in Figure 3.2. This enables the menu to drop-down when the mouse is moved over it. This gives it more of a native application feel rather than a web page feel.



3.2 JavaScript Drop Down Menu

The various form validations use jQuery. jQuery is not necessarily a language by itself but rather an extension for JavaScript. By using jQuery a real time actions can be evoked without having to refresh the web page. A sample is shown in Figure 3.2.1.

```

$("#addProfile").validate({
  rules: {
    username: "required", // simple rule, converted to {required:true}
    firstname: "required",
    role: {
      selectNone: true
    },
    website: {
      url: true
    },
    email: { // compound rule
      required: true,
      email: true
    },
    password: {
      required: true,
      minlength: 6
    },
    password_again: {
      equalTo: "#password"
    }
  }
});

```

Figure 3.2.1 jQuery Code for New User Form

The code blocks with *“required”* denote required fields. The user cannot

go further without having required text in those fields. The “*minlength*” code is used to make sure at least a certain number of characters are inputted, and “*equalTo*” is used to compare two fields. The errors can be seen in the Figure 3.2.2.

The screenshot shows a web application interface with a navigation menu at the top containing 'Home', 'Menu', 'Post', 'Users', 'Configure', and 'Tools'. The main content area is titled 'Profile' and contains a form with the following fields and errors:

- Username:** An empty text input field with an asterisk (*) and the error message "This field is required."
- First Name:** A text input field containing "Mohan".
- Last Name:** A text input field containing "Ram".
- Role:** A dropdown menu with "Select a role" selected and an asterisk (*).
- E-mail:** A text input field containing "ramm@" with the error message "Please enter a valid email address."
- Password:** A text input field containing "*****" with the error message "Please enter at least 6 characters."
- Re-enter Password:** A text input field containing "*****" with the error message "Please enter the same value again."

At the bottom left of the form, there is a legend: "* Denotes required fields". At the bottom right, there are two buttons: "Cancel" and "Add User". A copyright notice "Copyright ©TigeiaWriter" is located at the bottom right of the page.

Figure 3.2.2 jQuery Validation Form

The HTML and CSS are used to present the webpage to the user. HTML is a mark up language, which can be used to format a web page. Everything from bold to italics can be done. The CSS is used as a style sheet, this means a block of text can be shown in a particular place, or even have it shown with a different font type. The CSS can also be used for having different colors, different

text size, etc. as shown in Figure 3.2.3. All of these can be done in one centralized file(s).

```
body
{
    background-color: #e5e5e5;
    font-family: Georgia, "Times New Roman", Times, Ubuntu, sans;
    font-size: 14px;
}
```

Figure 3.2.3 CSS Code that is Used for Body Text in a Web Page

In the above code the body block of the web page has a specific background color which is light gray, and the font family is Georgia, and if that is not available on the client's computer then Times New Roman and so on is used. Also the font size is set to 14 pixels:

Search Engine Optimization

Search Engine Optimization (SEO) is way of formating the URL, so that search engines such as Google can easily index websites. With server-side scripting variables are passed around all the time; the most common way to do that is with the URL. Sites usually have URL that end with *page*, *id*, *article*, or

user; for example:

<http://www.example.com/index.php?page=1&id=123>

This type of URL makes it hard for search engines to index the site. In this case web crawlers tend to skip over sites that have these types of URL. This is where SEO comes into action. To make it easier for search engines URL is formatted to look like this:

<http://www.example.com/index.php/page/1/id/123>

How is this done? Well there are few ways to do this. One way to have search engine friendly URL is to have Apache web server do a mod rewrite. Mod rewrite is way to rewrite the URL into having more friendly look. The other way of having friendly URL is to implement a custom version of mod rewrite, and that is exactly what TrigeiaWriter has been programmed to do. A custom version of mod rewrite works by getting the URL, and using PHP function called `explode` to parse the URL, take the bits from it and store that into an array. From the array pieces are passed on to appropriate places, so for example, *page=1* will be *page/1* and *id=123* will be *id/123*. These SEO links are only used on the public site. The administrative site does not include any SEO links for security purpose and a file called *robots.txt* has been included in the root directory of TrigeiaWriter to tell the search engines to ignore specific directories.

Why does TrigeiaWriter implement it's own mod rewrite when Apache web server has it built it? The reason TrigeiaWriter was opted into using it's own mod rewrite is so that it can be independent of the web server. This means if

webmasters want to use Lighttpd as the web server instead of Apache, then this would not break the public site, as Lighttpd uses it's own version that is different of Apache's.

CHAPTER FOUR

FUTURE WORK AND CONCLUSION

Future Work

Mobile Site

With usage of smart phones and tablets on the rise, it would be in the best interest for TrigeiaWriter to have the capability to render a mobile version of the site when visited by a such a device. This means that an automatic redirection is needed and can be done by detecting the user agent string with a server-side language such as PHP [12]. Mobile site are made for smaller screen resolution (usually around 320x400 for smart phones) in mind with larger text to make the navigation easier. A mock-up is shown in Figure 4.1.1. The easiest way to implement this is to have a directory called */mobile*. This will offset the domain holder from having a subdomain, as it is not a requirement to have one. The mobile directory will hold a file that will use a different template for the site that is friendly to smaller screens. No additional code is required other than calling a few CSS files, and getting the data from the database.

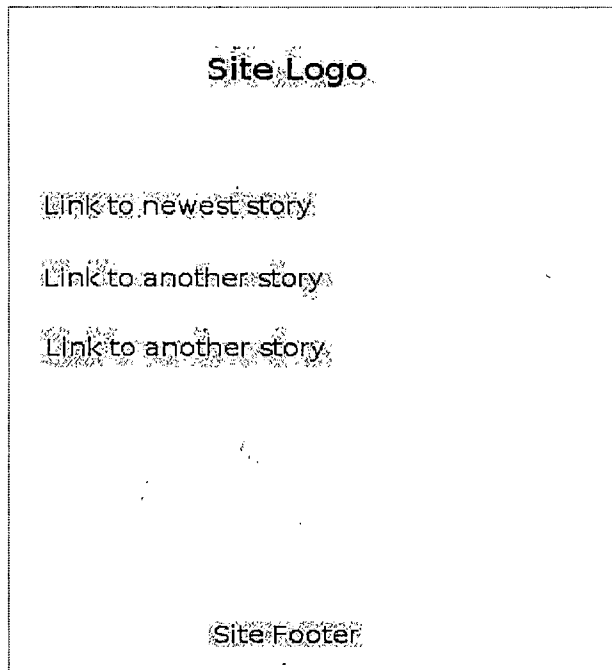


Figure 4.1.1 Mobile Site Mock-up

Hyper Text Markup Language 5

The HTML 5 is the next generation of the web markup language. It brings with it mainly the inclusion of <video> and <audio> tags which can embed video and audio clips respectively. This will help greatly with media rich articles as you can just embed video and audio like an image, and let the browser on the user's computer take care of the rest. This means the browser that is compatible with HTML 5 are going to have the appropriate video and audio codec. So these media files will run on the computer without any other extra software needed on the web server. Since HTML 5 is relatively new, it needs to be standardized by the W3C. Since it is not standardized not all browsers fully support it for now.

Another notable addition to HTML 5 is form validation. Since form validation is important to manage proper data input, having it built into the markup language eliminates the need for an add-on [13]. For TrigeiaWriter jQuery validation has been implemented as a form validation, if the user has JavaScript turned off then a server side validation is used to make sure there is a proper input. With HTML5 these two implementations can be eliminated and have the web browser take care of validation. To test out the HTML 5 validation, *required* is used for fields that are required such as fields that deal with user's name, email address, etc. The sample code can be seen in Figure 4.2.1.

```
<input type="text" id="firstname" required="required" />
```

Figure 4.2.1 Required Field for Input Box in HTML 5

By adding *required* in the input tag, browsers that support HTML5 take this into account and make sure that input field is populated, as shown in Figure 4.2.2.

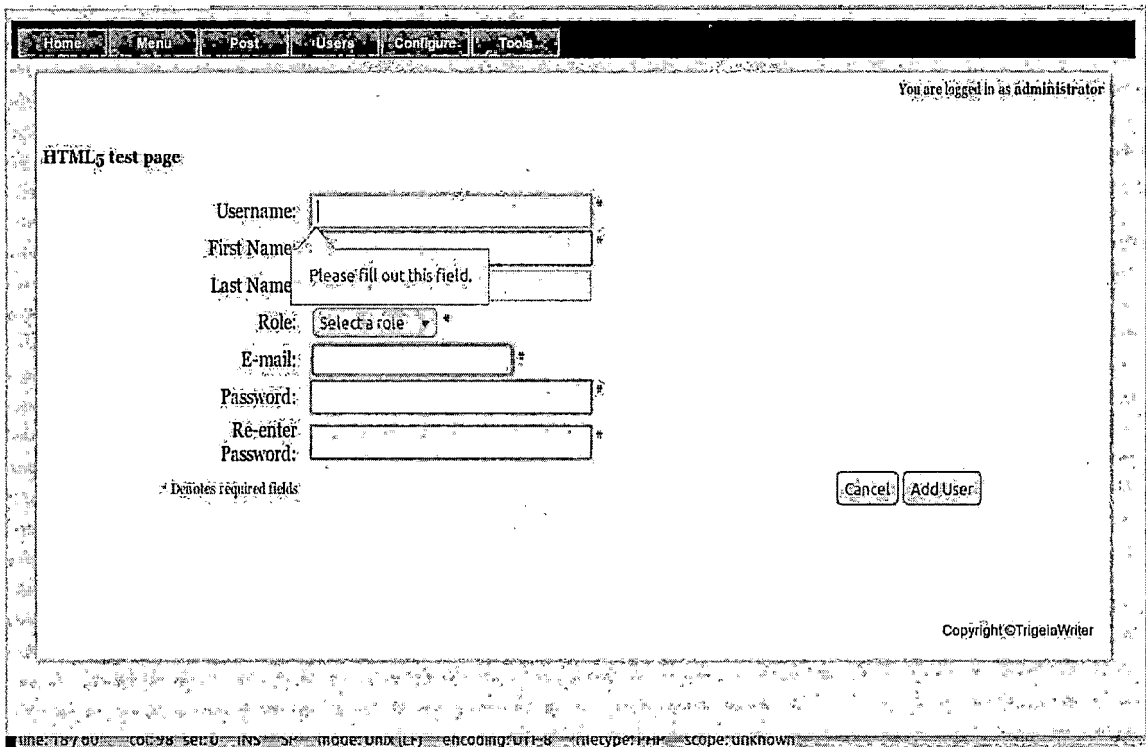


Figure 4.2.2 HTML 5 Required Field

Email is used for the email field input so the validation looks for an '@' and an '.' via regular expression, as these characters are common with all email addresses, as seen in Figure 4.2.3. The code for this would look like:

```
<input type="email" name="email" />
```

Figure 4.2.3 HTML 5 Code for Checking Email

If an URL field for a form is needed, then *url* can be used as the type. With that then HTML 5 will use the built regular expression to validate the URL:

```
<input type="url" name="url" />
```

Figure 4.2.4 HTML 5 Code for Checking URL

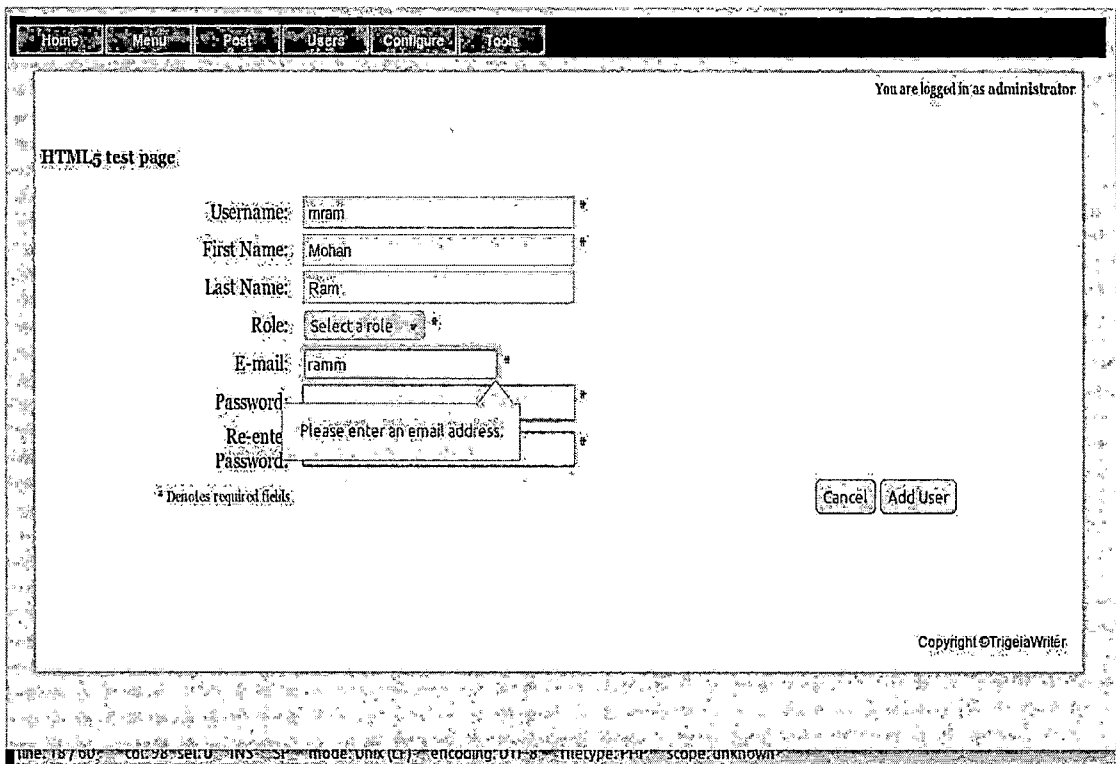


Figure 4.2.5 HTML 5 Email Validation

Also *autofocus* is used on the first field of the form, this enables the cursor to be located on that field waiting for the user to input. This is not crucial, but is a nice touch, and was only possible with addition of JavaScript before [14]. Example of this can be seen in code below.

```
<input type="text" id="firstname" required="required"
autofocus="autofocus" />
```

Figure 4.2.6 Auto Focus in HTML 5

When the user are presented with a search input, often times websites have text in them to give clues, such as Search for User. With HTML 5 *placeholder* tag can be used to have text in the box and upon focus will disappear so input can be taken by the user. Before this type of thing could have only been done using JavaScript. An example of this code is shown below.

```
<input type="url" name="url" placeholder="Enter your URL" />
```

Figure 4.2.7 Placeholder in HTML 5

In HTML 5 form validation works with the latest versions of Firefox, Chrome, and Opera web browsers, but it is not yet supported in the latest version of Internet Explorer and Safari web browsers. Since HTML5 is not properly implemented yet that could be the reason, once it is I am sure all browsers will have support for this.

No Structured Query Language

NoSQL is a fairly new concept that is complete opposite of the classic relational database management systems (such as MySQL). The data that is stored are not required to have a fixed table scheme, and usually avoid join operations and typically scale horizontally. There has been a good measurable movement towards NoSQL, mainly in the social networking field (facebook,

twitter, etc.). An example of NoSQL is Apache Cassandra, which was worked early on by facebook and open sourced with Apache license and is maintained by the Apache Foundation. A sample NoSQL command would look like this[15]:

```
nosql cat articles.rdb
```

Figure 4.3.1 Sample NoSQL Command

Code 4.3.1 illustrates how to get data out of a articles, once the command is invoked in the command line the output should look like something like this:

ID	TITLE	BODY	CATEGORY
001	Hello World	Hello world	misc
002	End of Time	End of time	misc

Figure 4.3.2 Sample Output of NoSQL

There are few other choices in NoSQL databases, such as Google's BigTable, Amazon's Dynamo, which are widely used internally in their respective companies. Like HTML 5 this is also relatively new technology. Currently TrigeiaWriter uses traditional SQL for storing data, but it would be interesting to

see the performance of TrigeiaWriter if it were to use NoSQL for storing data and see how it scales.

Conclusion

TrigeiaWriter has been developed with cross-browser in mind. This means it adheres to the web standard, and no matter what type of browser is used, as long as it is a modern web browser, the user will have the same experience. This means not only for the administration section, but also for the public site. TrigeiaWriter also have the capability to scale, which means even though it was made for Trigeia itself, TrigeiaWriter can be used for any other site that is in need of CMS or a blog. With extension in mind TrigeiaWriter was developed using MVC architecture. It has the potential to grow and morph into whatever the client wants, but leaving the core mechanism in place.

TrigeiaWriter also has room to grow, with newer web technologies being introduced it can be modified to use those new technologies. For instance NoSQL can be used instead of MySQL to store the data.

REFERENCES

- [1] Mike McGrath, PHP in easy steps, Computer Step, 2003
- [2] Mike McGrath, SQL in easy steps, Computer Step, 2004
- [3] Aaron Brzell, WordPress Bible, Wiley Publishing, 2010
- [4] PHP Documentation Home Page, <http://www.php.net/docs.php>
- [5] MySQL Documentation Home Page, <http://dev.mysql.com/doc/refman/5.5/en/>
- [6] jQuery Documentation Home Page, http://docs.jquery.com/Main_Page
- [7] W3C Schools, <http://www.w3schools.com/>
- [8] W3C Web Site Standards, <http://www.w3.org/QA/2002/04/Web-Quality>
- [9] Wikipedia Home Page, <http://en.wikipedia.org>
- [10] WordPress Home Page, <http://www.wordpress.org>
- [11] Web Page Accessibility Policy,
<http://acm.csusb.edu/webaccessibility/default.html>
- [12] Smashing Magazine: How To Build A Mobile Website, November 03, 2010,
<http://www.smashingmagazine.com/2010/11/03/how-to-build-a-mobile-website/>
- [13] HTML 5 Form Validation on SUMO,
<http://blog.mozilla.com/webdev/2011/03/14/html5-form-validation-on-sumo/>
- [14] Dive Into HTML 5, <http://diveintohtml5.org/>
- [15] NoSQL Tutorial, <http://www.linuxjournal.com/article/3294?page=0.1>