

BACHELOR'S THESIS

Degree Programme in Business Information Technology

Business Information Systems Management

2011

Prince Akyereko

DEVELOPING WEB BLOG SYSTEM FOR DEVELOPER'S HELSINKI OY



TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR'S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Business Information Technology | Business Information Systems Management

2011 | 66

Paivi Nygren

Prince Akyereko

DEVELOPING WEB BLOG SYSTEM FOR DEVELOPER'S HELSINKI OY

This Bachelor's thesis project is about web blogs and developing one with MySQL and PHP. The blogging system was built for Developer's Helsinki Oy.

Gathering the user requirements was done through reading of the documentation accompanying the project, reading the manuals of the older blog system and interaction with other developers and owners of the project. The goal was to build a web blog system for the users to send, read and comment on posts.

The blog was built with PHP, CSS, MySQL, JavaScript, and HTML. It has both the administrator's and the registered users' interfaces. The system was tested on the most common browsers.

KEYWORDS: Weblog, PHP, MySQL

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

1 INTRODUCTION	7
2 BACKGROUND	10
2.1 Advantages of Blogs	11
2.2 Disadvantages of Blogs	13
2.3 Making a Blog	13
3 WEB BLOG FOR DEVELOPERS HELSINKI OY	16
3.1 The current system and Problems Associated with It	16
3.2 Gathering User Requirements	16
3.3 Building the Application Process	18
4 HOW THE SYSTEM WORKS	25
4.1 Creating Really Simple Syndication (RSS) Feed for the Blog	37
5 TESTING THE APPLICATION	38
5.1 Layout Testing	38
5.2 Functional Testing	42
6 SUMMARY	47
SOURCE MATERIAL	49

APPENDICES

Appendix 1 Application Codes	50
Appendix 2 Other UML Diagrams of the system	64

SCREENSHOTS

Screenshot 1: Login page	24
Screenshot 2: Sign up form	26
Screenshot 3: Writing post in the system	29
Screenshot 4: After submitting post in the system	30
Screenshot 5: Updating post in the system	31
Screenshot 6: After submitting update post in the system	32
Screenshot 7: A visitor (registered) writing comment on a post	34
Screenshot 8: After visitor (registered) called "peter mensah" has submitted his comment on a post	35
Screenshot 9: Searching for post result in the system by a visitor called "kojo"	36

Screenshot 10: Layout testing in internet explorer	39
Screenshot 11: Layout testing in Google chrome	39
Screenshot 12: Layout testing in Firefox	40
Screenshot 13: Layout testing in Safari	41
Screenshot 14: Layout testing in Opera	41
Screenshot 15: Update post in Internet explorer	43
Screenshot 16: Compose form in Internet explorer	43
Screenshot 17: Compose post form in Safari	44
Screenshot 18: Updating post in Opera	44
Screenshot 19: Testing the search button and results n Opera	45
Screenshot 20: Post selection and comment form on Google chrome	45
Screenshot 21: Post comment form and post selection in Safari	46

FIGURES

Figure 1: Figure 1: Tables of the blog database	19
Figure 2: Use case diagram of the Developers blog system	20
Figure 3: Activity diagram of adding new post	22
Figure 4: Activity diagram of commenting on blog post	23
Figure 5: Use Case description and scenario of commenting on added post	24
Figure 6: Briefly descriptions of roles perform by registered users in the system	27
Figure 7: Account Creation Use Case and Description	64
Figure 8: Activity Diagram of Creating User Account	65
Figure 9: Flagging and banning inappropriate Comment	66

TABLES

Table 1: Event Table of the Blog System	21
Table 2: Layout testing	38
Table 3: Functional testing	42
Table 4: Compose Post Form (HTML)	50
Table 5: Sending Post into the Database (blog_post.php)	51
Table 6: PHP Code that lists all posts on the Page (listing_post.php)	52

Table 7: Commenting on post code	53
Table 8: Deleting Post	54
Table 9: HTML and PHP code that creates the Blog Home page (index.php)	55
Table 10: XML FEED FILE (blogfeedindex.xml)	57
Table 11: CSS File for the Application. (css_blog.css)	58
Table 12: Blog Home Pages (HTML)	62
Table 13: Headers and Footers. It has both header.php and footer.php	63

LIST OF ABBREVIATIONS

HTML	Stands for Hyper Text Markup Language. A markup language is a set of markup tags. HTML uses markup tags to describe web pages (W3school Group, 1999-2010.)
PHP	Stands for Hypertext Pre-processor, is a server-side scripting language and scripts are executed on the server (W3school Group, 1999-2010).
CSS	Stands for Cascading Style Sheet (W3school Group, 1999-2010).
JavaScript	Was designed to add interactivity to HTML pages and is a scripting language (W3school Group, 1999-2010)
ASP	Stands for Active Server Pages and is a Microsoft Technology that runs inside the IIS (W3school Group, 1999-2010)
MySQL	is an open source relational database management system based on the Structured Query Language (SQL)(W3school Group 1999-2010).

1 INTRODUCTION

The invention of the internet in the 19th century paved way for many ways of communication. Websites are integral part of communication in our everyday online activities. Nowadays with the availability of internet in every home blogging is becoming more and more popular.

Today blogs and other websites have become a source of information for many people and organizations all over the world. Schools, government and private organizations, journalists, news houses etc. all depends on a web blog system to tell us the information that we need and inspire us to visit their web pages every day. (Wikimedia Foundation, Florida, USA, 2010.) Therefore, there should always be an easy way anybody can post and edit something into a website that they have right to without needing to edit the programming code. Without a web blog system being incorporated in a web page, it is always going to be difficult for people to post and add information to their websites everyday.

This project focuses on building a web blog system from the scratch for Developer's Helsinki Oy using HTML, CSS, PHP, MySQL, JavaScript etc. MySQL was the database server used to store the blog posts and PHP was the scripting language used to put content into the database and bring it back into the weblog. The building of the application began by reading the project documentation submitted to me by the company and a personal discussion with the project manager and other developers.

The project shows a way of designing a blog that will be user friendly and allows its visitors to be part of the system. The visitors who read the posts will be allowed to contribute and comment on the posts and their comments will be available to the entire audience. The system was designed in two phases. There is the administrator interface and the home page for the blog posts to appear for the visitor. The administration site allows the administrator to post and edit blog posts. Moreover, it allows him/her to list the posts and delete any

post that is no more relevant and must be removed from the system. The administrator interface is where the blog post is composed before it can be posted to the home page for the visitors to see. The home page for the blog is where the most recent posts are shown. It also has the search facility and archives area where the older posts can be seen. The part of the blog that invites the visitor or the user to be part of the system by contributing and leaving comments motivates and attracts some people to visit the page.

There is also the Really Simple Syndication (RSS) feed for the blog. The RSS feed is a way in which websites distribute their contents. 'An RSS feed is an XML file that tells the outside world when you have updated your blog so your readers do not need to keep coming back to your website to find out whether you made any changes. It defines an easy way to share the website updates, headlines and news to regular customers who have subscribed to the RSS feeds of the website'. (Andy Budd et al., 2006, 329.)

In designing the system, the developer has taken into consideration the needs of the final user and how the user will interact with the system. The user and the company's requirements are those implemented here with the view of achieving the client's needs. Every technology used in this project is by the request of the customer. A database of the blog and other tools used in the system conform to the contemporary design rules of the company.

In the future, the blog could be modified and improved to have more functions with time. A feature could be added which will allow one to post and preview his post in the administrator area before they are finally posted onto the blog. In addition, the same preview section could be added to the comments area also. Moreover, in the future, other ready made tools such as Textile and Markdown could be added to the blog to allow good and more advanced formatting. It is possible to add categorization system to the blog posts which will allow the posts to be categorized

A sample of people was selected to test the system and give feedback. Developer's Helsinki Oy has acknowledged the system and has implemented it

and customer feedback has also been encouraging.

2 BACKGROUND

Blogging kicked off big style in 2004. It was around for quite a few years before that, but not really mainstream. The word “blog” certainly couldn’t be found in the pages of a dictionary and was only used in conversations online, by those who were “in the know”. These days, you hear it everywhere. (Andy Budd et al., 2006, 5.)

A blog is website mainly maintained by an individual with regular entries of articles, descriptions of events, or other materials such as photography, art or other media files (Andy Budd et al., 2006, 3). A blog is a personal website that provides updated headlines and news articles of other sites that are of interest to the user; also may include journal entries, commentaries and recommendations compiled by the user; also written web log, weblog; also called blog (Andy Budd et al., 2006, 2). Again, a blog often call weblog is a type of website, normally maintained by an individual with regular entries of articles, descriptions of events, or other materials such as photography, art or other media files. Posts are frequently exhibited in a reverse dated order. Blog can also be used as a verb, meaning to update or add content to a blog. (Wikimedia Foundation, Florida, USA, 2010.)

Majority of blogs are interactive, allowing visitors to post comments and even message each other through gadgets on the blogs and it is this interactivity that differentiates them from other static websites. So many blogs provide a written explanation on a particular theme; others function as more personal online log. A typical blog combines text, other inputs, images, and links to other blogs, websites, and other communications related to its subject. Today, the ability of readers to post comments in an interactive way is an integral and essential part of many blogs. A lot of blogs are mainly textual, notwithstanding some blogs focus on music (MP3 blog), art (Art blog), videos (video blogging), audio (podcasting), and photographs (photo blog). There is another type of blogging which give prominence to very short posts. This way of blogging is called Micro

blogging. (Wikimedia Foundation, Florida, US, 2010.)

Again, a blog can also be described as a website that allows users to reflect, give opinions, and talk about various subjects in the form of an online journal while readers may comment on posts. Posts usually appear in a reverse sequential order. (Mediawiki, Creative Commons Attribution-Share Alike, 2002.)

Apart from those types of blogs mention above which are mainly categorized by the media type, there are other different types of blogs which needs to be mentioned. Among of these, the differences are seen based on the way or the purpose in which the blog is used for. Some blogs are use for personal purpose while others are for political reasons and other reasons (Wikimedia Foundation, Florida, US, 2010).

Personal blog is normally a commentary by a single person and is the most traditional and a regular form of blog that exists. There are also corporate and organizational blogs. These are blogs used or written internally to enhance the communication and a way of life in an organization or outside for public relations or advertising and selling purposes. (Wikimedia Foundation, Florida, US, 2010.)

A blog on the other hand can also be categorized by a genre. Normally, this type of a blog is written purposely with a focus on a particular subject such as political blog, educational blog, and legal blogs. (Wikimedia Foundation, Florida, US, 2010).

A web blog with shorter entries and blend media type is called tumble log. A blog that is written on typewriters and then scanned is called typecast. Also a blog can be described by the device being used to send or to compose it. A blog written by a mobile device like a mobile phone is called a mob log. (Wikimedia Foundation, Florida, USA, 2010.)

2.1 Advantages of Blogs

The basic advantage of having a blog is to enable a user to share their views

and opinions about a particular topic. On the hand, a unique feature of a blog is the ability to let registered users comment on what they read. This lets interested individuals discuss that particular topic. This leads to further exchange of information and ideas. (Wetzel, 2010.)

There are quite a few major blog hosting sites that offer various features that enable users to determine how they wish to express their opinions. For example, a blog hosting site may offer various different templates which would make it attractive or perhaps try to make creative parallels between the writer's ideas. (Wetzel, 2010.) This unique display style implemented by the blogger creates a persona that attracts like minded individuals, thus creating a sense of community with dissemination of interesting information. In essence, blogs are easy to maintain because they are organized in chronological order. For example, a reader would be able to access a blog according to time if they wish, or they can search with respect to the topic titles. (Wetzel, 2010.)

An important thing to note about blogs is that they are easy to edit as well. That is because the only thing required is login information and access to internet. This makes it easy for bloggers to express their views wherever they may be. (Wetzel, 2010.)

Another significant advantage of blogs is that it lets users share their views when they might not be given a voice due to selectiveness of other media (such as newspapers, magazines and other forms of publications) (Wetzel, 2010). That is because these print media have limited capacity and they must choose what might be in the best interest of their organization, hence the selectiveness. On the contrary, in the blogosphere there is virtually no limit as to how much and how well one would like to express his/herself. As a result, people choose to express their feelings and views on the blogosphere depending on what they feel like sharing. (Miller & Pole, 2010.) Interestingly though, blogging also creates an opportunity for bloggers to be picked up by news media as a supplement to their news stories. For instance, when the iPhone was leaked, all the anticipated information was acquired from the Gizmodo technology blog. As a result, Gizmodo enjoyed the fame that it probably would not have been able

to, if it had contacted a news outlet. (Chen, 2010.)

Financially, blogging can increase a firm's return on investment by having people blog about topics that concern the firm (Miller & Pole, 2010). For instance Developers Helsinki Oy would be able to get a better insight as to what concerns the general public when it comes to designing a system. Blogging would also enable them to understand the demographics of their customers and the systems they seek out. With this vital information, they can then re-design their business processes in such a way that would give them a competitive advantage. Moreover, it would also create a sense of affiliation—where the users would gladly be associated within the market place.

2.2 Disadvantages of Blogs

The main product of blogging is the spread of information. When individuals talk about a particular topic, they tend to get into a discussion highlighting the benefits and risks associated. Therefore, the most harmful thing to come out of a blog discussion is negative publicity of a particular product. For example, if one is discussing a health product and someone makes a comment about the side effects, the comment could potentially harm subsequent sales of that particular product. (Wetzel, 2010.)

Another disadvantage of blogging is that people might disguise themselves as an expert. This can cause people to believe in false information and might even cause health related damages. Therefore, it is good to keep track of every blog post. Since that would be very difficult, it would be appropriate to allow users to report inappropriate posts and comments.

2.3 Making a Blog

A web blog like any other website can be programmed using a lot of modern day web technologies. It can be programmed with HTML, CSS, JavaScript, PHP, HTML, CSS, ASP, Python etc and even with XML. On the other hand, a blog can be created without the need to know any of the above web applications and how to code. Today building sites such as Movable Type,

ExpressionEngine, WordPress, and Textpattern have made it much easier for someone who does not know coding could even create a blog without any problem. These tools are also good platform to create a very good user friendly and an interactive weblog. (Acroterion, 2001-2007.) There are different ways one can create or design web blog depends on the person's requirements and how it wants the blog to be.

A blog could be a dynamic one or static depending on the way is programmed and how it functions. The choice between dynamic and static when building a web blog is a critical one that needs to be talked about and must always be taken into consideration when making such a project. (Acroterion, 2001-2007.)

Dynamic websites normally provide the visitor or the user with the ability to interact with the content and give some form of feedback. However, the reason for calling these sites dynamic has to do with how the sites are developed and maintained. In the dynamic web site, all of the content, styling files and related web documents are contained within one or more databases located somewhere on the Web and normally administered by an application called a Content Management System. (Acroterion, 2001-2007.) The content is cascaded on the website using various templates written in a dynamic mark-up language such as PHP, JAVA, and ASP. These languages are like HTML, but they are more complicated and can do a great number of things than common HTML. Creating pages dynamically allows for all sorts of intelligent applications, from e-commerce, random quote generators to full on web applications such as Facebook, Twitter and MSN. (Acriterion, 2001-2007.)

Static weblog or website on the other hand is one that its pages are stored on the server in the format that is sent to a client web browser (Acroterion, 2001-2007). Normally, it is designed in simple HTML and does not use server side languages like PHP, ASP, Python and other languages available in its application building. Usually static websites display the same information to its visitors most of the time. (Acroterion, 2001-2007.)

This project seems to be one of the dynamic blogs around because it allows the

user to interact with blogger by posting comment thus making it interesting for the user to visit the site all the time. The administrator can also remove a dangerous or abusive user from the system.

3 WEB BLOG FOR DEVELOPER'S HELSINKI OY

Developer's Helsinki was founded in January, 2009 to develop, test and market great communication ideas. It is a small company operating in the Helsinki region. 'The company aims to create intelligent and user-friendly websites and services, which bring real value to our customers. Our goal is to play a central part in the next generation of web-based innovation. That is why we research and develop various projects, which give us insight and tools whereby we can create a great web presence for our customers' said by Harry Alanen, co-founder of the company. (Developer's Helsinki Oy, 2010.)

3.1 The current system and Problems Associated with It

This company builds internet applications and web base systems that add value to the existing one. The company current website provides information for its users and customers but does allow the managers of the site to add a message for its visitors so easily. To add a message one needs to go and edits code before adding it. That has called for a modest system that will work along with their website and allow anybody whether somebody with developing experience or without developing experience to post a message to the site without any difficulty. Not that there was no website for the company, there was but it was done at beginning of the company and now needs improvement from the existing one. So there was the need in design a blog for it, in order to make communication between the company and the public easier. The older website was a static one which was not easily to blog everyday. So the company mooted the idea of new blog to respond to the challenges of today environment.

3.2 Gathering User Requirements

The first process of gathering information about the project began by first reading the documentation that accompanies the project. After that I took the old system to look at it and study it carefully. I investigated the good and bad

features of the older blog. Again I read the user manual and documentation of the older system that exist in the company archive.

After reading the documentation of both the older and the new system, and investigated the older system I met the chief operating officer to discuss the documentation further with him. When I met him I discussed the documentation with him and tried to understand some of the things I do not understand in the document in order to gather the requirements well. After the meeting with him and interaction with other developers I gathered what the system needs to do and some of the restrictions that must be applied in the design face.

There must be a database for the blog which should include a database table for post and a database table for comment and also a table for users. Every blog post must have a unique ID and should be time stamped and posts must appear in chronological order with the seven most recent ones appearing first. When an older post is edited, it should not change the original timestamp. Every post written is also associated with the author or the writer. An author must login in order to write and post a blog. The entry or post should consists of; title, date, author name, content, which may include text, styled, links, pictures or embedded video.

When writing a post or entry for the blog, the following elements, restrictions and commands must be applied; the title should be a simple input field. It is required before the text could be submitted. No post or text can be submitted until a title has been written or entered. A maximum of 140 characters, including spaces, can be entered into the input field. This input field should be designed using HTML and CSS should be used to style it. Content is the area where the discussion of the post takes place and it is necessary to construct this area in a way that encourages good communication. The author's comments should be designed in a different way so that they are a little bit different and going through other comments should be easier. The content should be designed with the user in mind not forgetting the last user.

3.2 Building the Application Process

After gathering the user requirements, I began by first drawing UML diagrams for the project. I submitted the UML diagrams to the chief operating for comment and approval. The next day he came to me in the office and discussed the UML diagrams further with me. He gave me the suggestion as how to improve my UML diagrams and after further consultations with other developers let to the drawing of the UML diagrams in both Figure 1, Figure 2, Figure 3 and Figure 4 below respectively.

The application was built with MySQL and PHP. MySQL is the database server used to store the blog posts and PHP is the server-side scripting language used to put content into the database and pull it back out into the home page template for visitors to access it. The forms and layout were built with HTML, CSS and JavaScript. The PHP codes were done in NuSphere 3.0.

The application has two users, the administrator and the visitors of the site. The visitor must log in order to perform other duties apart from just reading the blog entry. In blog development the administration site where the post is compose and edited is very important. So this project will show the logic of composing post and how a registered visitor or administrator can comment on the post here.

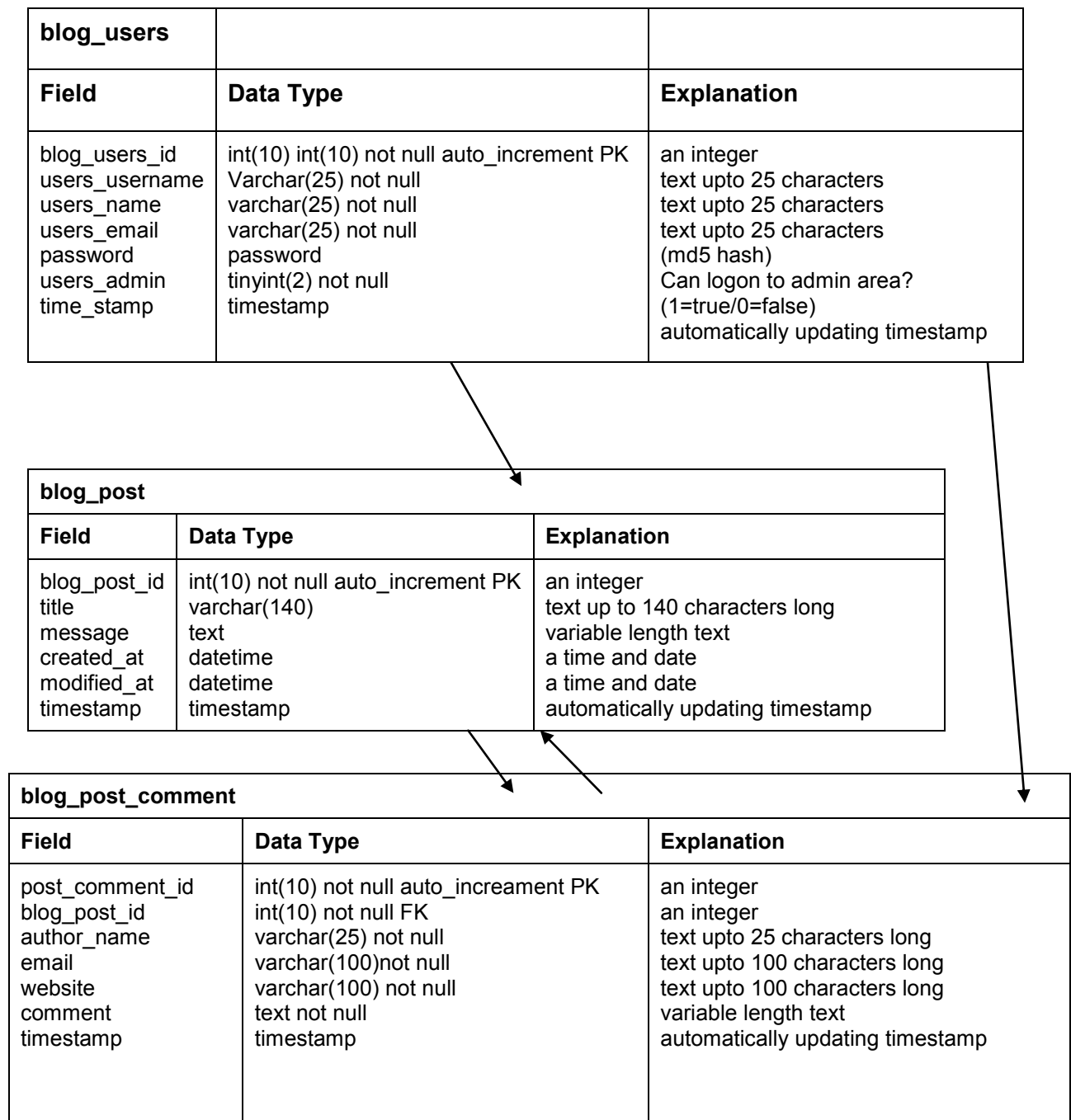


Figure 1: Tables of the blog database

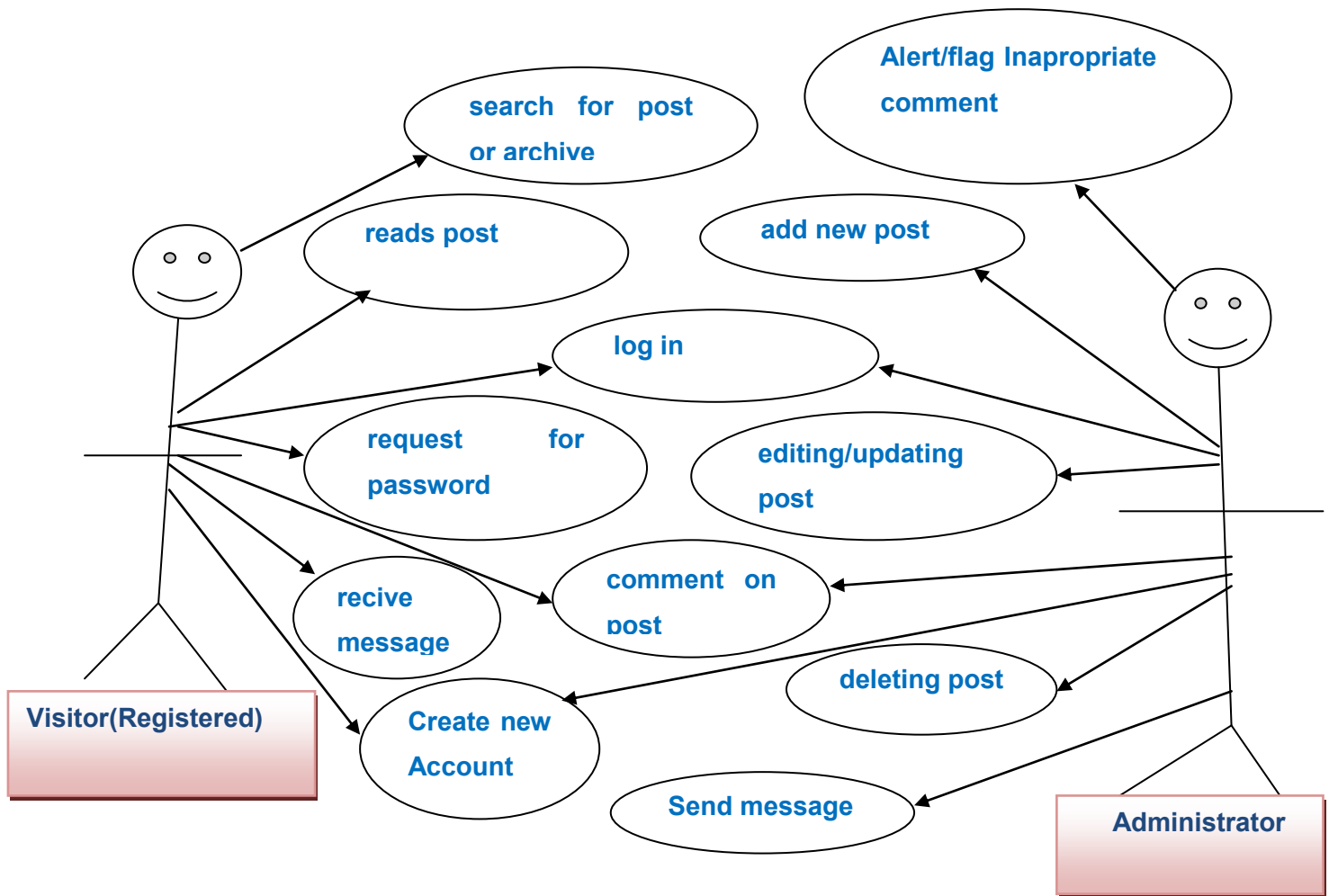


Figure 2: Use case diagram of the Developers blog system

The figure above depicts the system's actors' various roles and how they interact with the system.

Table 1: Event Table of the Blog System; indicating event, trigger, and source of the action, activity, response and destination of the action.

EVENT	TRIGGER	SOURCE	ACTIVITY	RESPONSE	DESTINATION
Unregistered user(visitor) or Administrator wants to create an account	New account creation	Unregistered user	Create new account and submit	Account creation confirmation	System
Administrator wants to send a message	Compose new message	Administrator	Send Message	view message	Registered user
Administrator wants to create or add a new Post	Compose new post	Administrator	Submit new post into the database	Confirmation of post into the system	System
Registered user(visitor) or administrator wants to comment on a post	Compose new comment	Registered user(visitor) or Administrator	Submit comment into the system	Confirmation of comment and comment appear beneath post	System
Administrator wants to edit/update Post	Edit Post	Administrator	Submit updated post into the system	Confirmation of update and post is updated	System
Administrator wants to delete post in the system	Delete post	Administrator	Select and delete post	Confirmation of delete post	System
Registered user wants to search for post	Search for archive post	Registered user	input what to search for and enter	Display the search result	System
Administrator wants to alert inappropriate content	Alert/ Warning Notice	Administrator	Alert inappropriate content to the user	User receive and view the warning	System registered user (visitor)

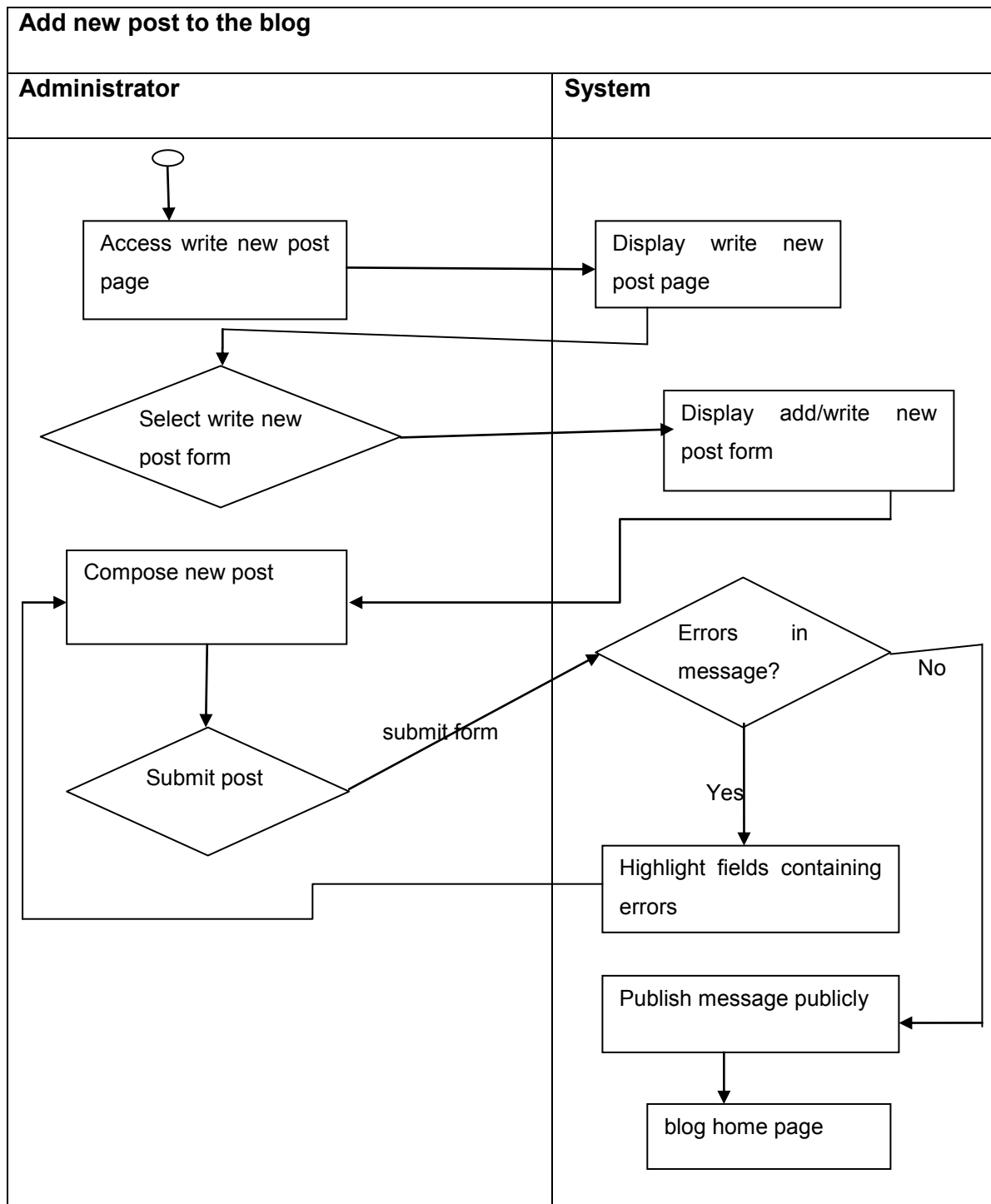


Figure 3: Activity diagram of adding new post.

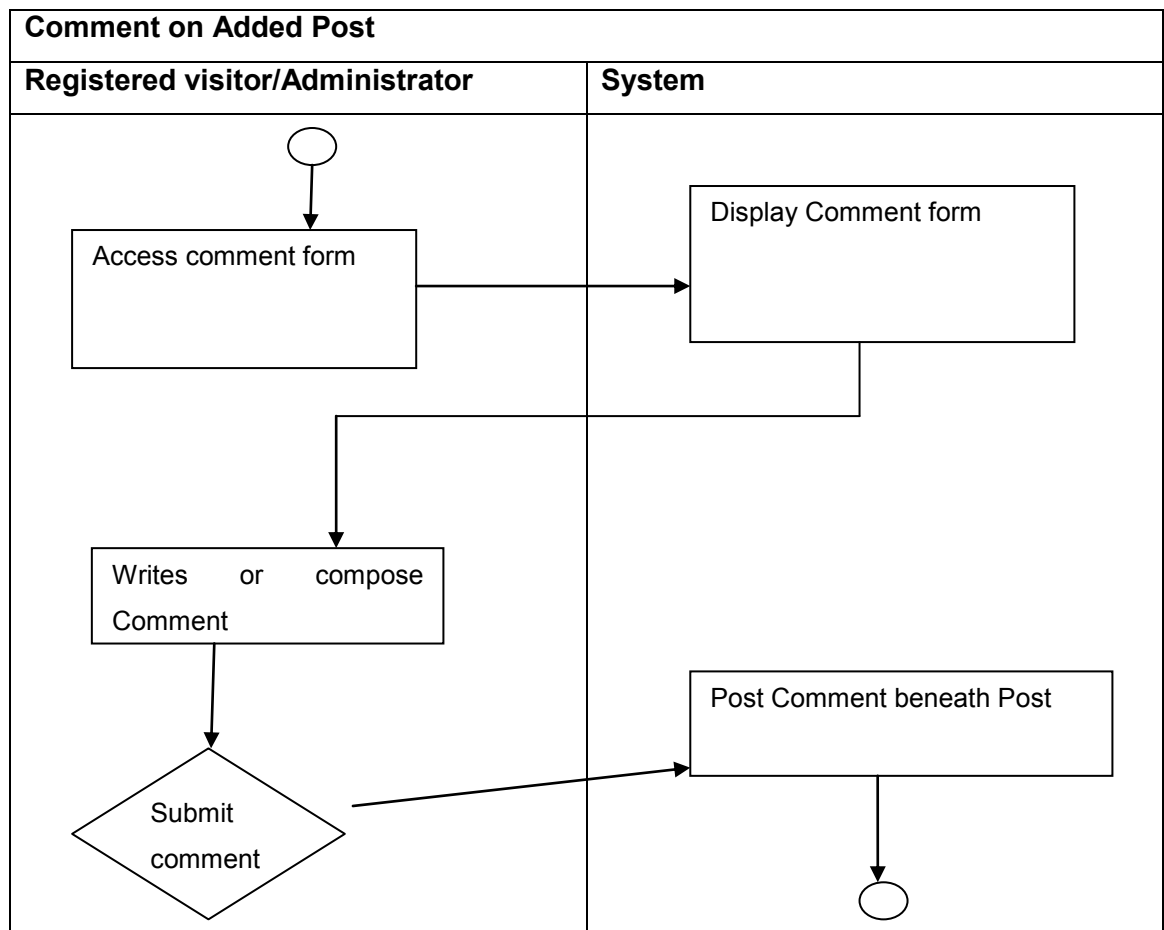


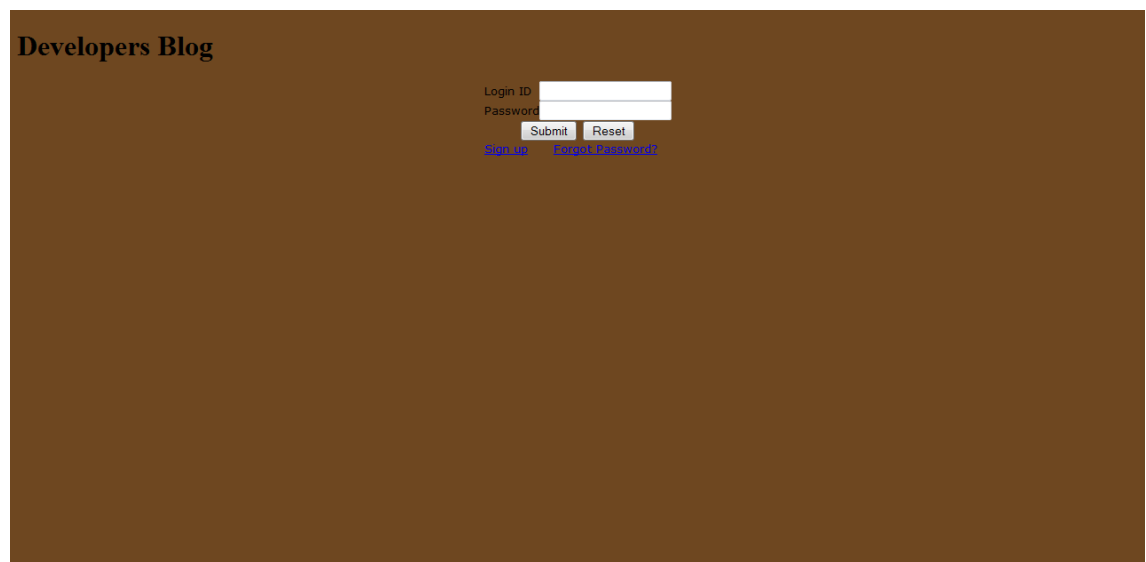
Figure 4: Activity diagram of commenting on blog post

Use Case Name	Comment on Blog Post	
Scenario:	Registered visitor or administrator wants to comment on a post entered	
Triggering Event:	Registered visitor or administrator accesses comment form on a post	
Brief description:	Registered visitor or administrator accesses comment form by clicking “add comment” button within a blog post. Registered user or administrator types his/her comment and submits the comment which will be posted beneath the blog post.	
Actors:	Registered visitor, Administrator, System	
Pre-conditions:	System must submit comment.	
Flow of Events:	Actor	System
	1. Registered visitor or administrator activates comment box by clicking “add Comment” button.	1.1 System displays comment form
	2. Registered visitor or administrator types comment into comment form.	
	3. Registered visitor or administrator submits comment	3.1 System saves and add comment beneath the blog post
Exception Conditions	<ul style="list-style-type: none"> • If registered visitor or administrator closes comment page or box, then system does not submit comment. • If unregistered user tries to comment on a post, then system will reject comment and ask user to create an account. 	

Figure 5: Use Case description and scenario of commenting on added post

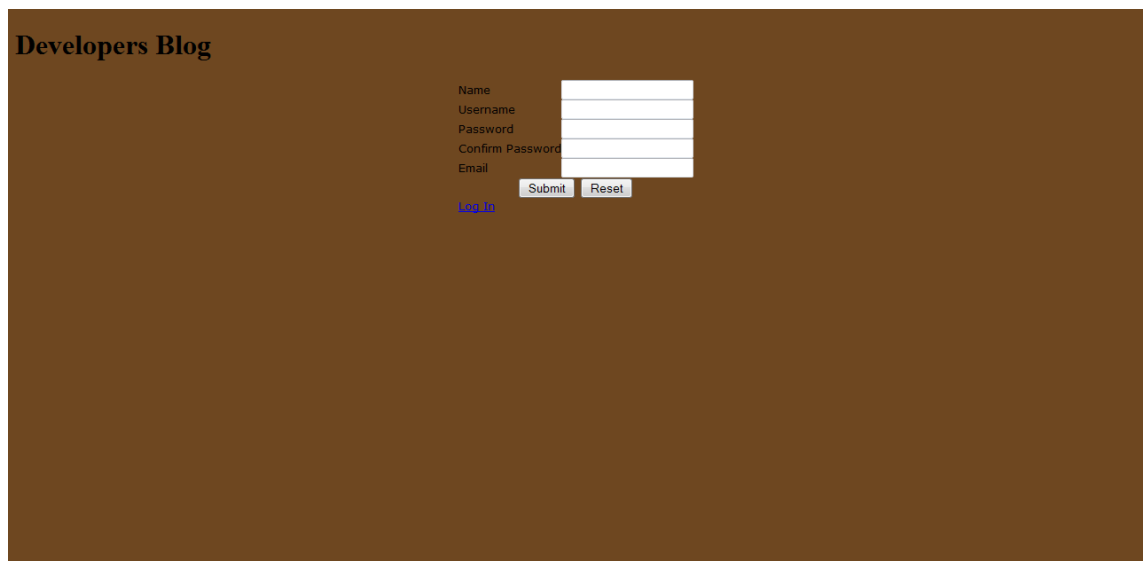
4 HOW THE SYSTEM WORKS

First of all there are other functions of the system do shown in the class diagram, Use Case diagram, and Event table (Figure 1, 2 and Table 1) respectively but will talk about it here and will not include the codes of it base on the agreement I had with this company who owns this project. Examples of such areas are login and registration code, flagging inappropriate comment and removing a user from the system.



Screenshot 1: Login Page

There are two users of the system. They are administrator and a visitor, the administrator must log into the system at all time and the visitor must also login into the system in order to access the facility and be able to comment on post. The user must input his/her Log in details in order to sign in to the system. On the other hand if a user inputs wrong or not existing sign in information, the user will be redirected to the sign up form below to sign up before he/she can long in to the system.



The screenshot shows a sign-up form on a dark brown background. The form is titled "Developers Blog" in the top left corner. It contains five input fields stacked vertically, labeled "Name", "Username", "Password", "Confirm Password", and "Email". To the right of these fields are two buttons: "Submit" and "Reset". Below the "Email" field is a link labeled "Log In".

Screenshot 2: Sign Up form

When the user is redirected here he /she will then input the name, username, password, and email and then submit it here in order to register. Again if the user is a new and have not signed up yet, he/she must register the details here in order to log in and access the facility.

When the user logs in to the system, each user has a set of functions he/she can perform. In the table below is each user and its roles briefly.

User	Function
Administrator	<ul style="list-style-type: none"> -add new post -create new account -update/edit post -delete post -comment on post -send message -remove inappropriate comment -alert/flag inappropriate comment
Visitor	<ul style="list-style-type: none"> -read posts -create new account -edit personal user information -request for password -comment on post -search for post -access archive post

Figure 6: Briefly description of roles performed by registered users in the system

Both the administrator and the visitors are supposed to register in order to login to the system (blog_register.php). During registration the administrator distinguishes itself from the visitor based on the information entered and the different user levels available. After registration the administrator is directed to administrator page. There administrator can add new post (blog_post.php), edit/update (updatepost.php) or delete (deletepost.php) post. The picture below shows how an administrator can add post. He/she selects to write post and the

window below appears. The administrator then writes the post and after that submits it into the database. Below is that form that allows you to write a post and add it to the system. The HTML and PHP code (blog_post.php) that sends the post into the database can be found in the Appendix 1.

After a post has been submitted, then the system will send back to you “A new post was successfully submitted”. If the post is not successfully submitted into the database an error will come and redirect you to the compose form. Figure 3 in page 22, shows the logic of doing that in the system.

Developers Blog

Developer's blog post area

Write a new post or go [here](#) to edit existing blog [update existing post](#)

This is a developers administrators blog page

Title

Technology on the ups!

Date

2011-05-24:03:24
yyyy-mm-dd hh:mm:ss

Message

Anti-virus software vendor, Aspersion Lab says that the number of malicious programmable targeting mobile devices has more than doubled between August 2009 and December 2010. Records of Aspersion Lab's virus database for 2010 posted on their website indicated that last year, over 65% more new threats targeting mobile devices were detected than in the previous year; and over 1,000 variants from 153 different families of mobile threats were also added to the database. The company said the growing popularity of the Android platform in particular, has drawn the cyber criminals attention, adding that in August 2010, the first malicious programmed targeting Android was detected, and since then, that number has reached 15 programmed from a total of 7 families. "The first threats targeting Apple's iPhone OS also appeared during this last reporting period, but infected only devices that had been jail-broken in order to install third-party games and other software not manufactured by Apple," Aspersion said. It noted that most mobile threats continue to target the Java 2 Micro Edition (J2ME) platform, which is supported by a huge number of mobile devices, saying, that means it is not only smartphones that are at risk of infection, but basic mobile phones as well. The records showed that the second most-targeted platform is Symbian, with Python in third place. Senior Mal-ware Analyst at Aspersion Lab, Denis Millennium was quoted as saying "the use of EMS Trojans is still the easiest and most effective means by which malicious users can earn money.

Anti-virus software vendor, Aspersion Lab says that the number of malicious programmed targeting mobile devices has more than doubled between August 2009 and December 2010. Records of Aspersion Lab's virus database for 2010 posted on their website indicated that last year, over 65% more new threats targeting mobile devices were detected than in the previous year; and over 1,000 variants from 153 different families of mobile threats were also added to the database. The company said the growing popularity of the Android platform in particular, has drawn the cyber criminals attention, adding that in August 2010, the first malicious programmed targeting Android was detected, and since then, that number has reached 15 programmed from a total of 7 families. "The first threats targeting Apple's iPhone OS also appeared during this last reporting period, but infected only devices that had been jail-broken in order to install third-party games and other software not manufactured by Apple," Kaspersky said. It noted that most mobile threats continue to

A blog administrator's corner

Screenshot 3: Writing post in the system



Screenshot 4: After submitting post in the system

Another action that can be taken by the administrator is update/edit (update_post.php). Normally the form for updating the blog post should be the same like the one for writing and adding new post. It therefore uses the same PHP file for writing the new post and submits it. However the purpose of this area is to know which post to update as we do not need to update all the entries at the same time. Every post can be uniquely identified by its **ID**, so it uses query string in the PHP file to reference that.

A query String is the list of a **name=value** pairs in a URL that follow the PHP file name. You can always see them when using the search engines such as Google, Yahoo! and Bing. The elements of a query string are possible in PHP through a super global variable. (Andy Budd et al., 2006, 285.)

The picture below shows an administrator updating post in the system.

Developers Blog

Update a post
or go here to write a new post [write a new post](#)

This is a developers administrators blog page

Title:

Date: yyyy-mm-dd hh:mm:ss

Message:

A blog administrator's corner

Search

select existing posts below to edit

Developers in the verge of merging

Screenshot 5: Updating post in the system

In this screenshot, the administrator has selected “Developers in the verge of merging” from the lists of posts in the system and trying to update it. The administrator inputs the date of the update and write the new message that should be added or remove from the old message. After is submitted into the system by the administrator. If it is successfully done, then the system will send a message to confirm it. It follows the same sequence of adding new post. The print out (Screenshot6) below shows after successfully edited posts in the system by the administrator.



Screenshot 6: After submitting update post in the system

After submitted an edited post the system sends back to the screen on the print out below “Your posted was edited successfully”. “Return to blog home page”.

Deleting post already entered in the system, is one of the functions the administrator can perform. This function can only be performed by the administrator. This is a function called delete function built into the index file (delete.php) in appendix 1 which allows the administrator to do this.

If the administrator wants to delete a post already entered in the system; it is done by selecting which post to delete. The delete button next to the link for each post added will appear and the administrator then selects the delete button and click on it. Then the post would be deleted from the system.

The administrator is able to update and delete post because the system list all post in sequential order in the page (index.php). Because of the way the system list posts, an author or administrator can easily select which entry to edit or

delete without any difficulty. The system sorts the posts into dated order with the recent post being appeared first before any other post. The (index.php) which do that could be found in Appendix 1.

Another important part of this project is ability of the visitor or administrator to comment on a post. A visitor or administrator needs to log in, in order to comment on the post.

A unique feature of blogs is the ability to let registered users comment on what they just read. If there is anything in blogging that attract people to visit your site all the time, then is the system ability to allow the registered users to comment on the post. (Wetzel, 2010.) Nowadays in the developing world, a web based system that fails to make its visitors part of the system is bounded to fail.

Figure 4 and Figure 5 in pages 23, 24, show the activity diagram, and use case text of commenting on a post. Let us take a registered user (administrator or visitor) who wants to comment on a post. The user must select the post to comment on it and the comment form or comment box will appear (blog_post_comment.php) in appendix 1 and the user can now input his/her thought and submit it. (Screenshot 7) below shows a registered user commenting on a post and the outcome.

Developers Blog

Techbology on the ups!

Anti-virus software vendor, Kaspersky Lab says that the number of malicious programmes targeting mobile devices has more than doubled between August 2009 and December 2010. Records of Kaspersky Lab's virus database for 2010 posted on their website indicated that last year, over 65% more new threats targeting mobile devices were detected than in the previous year; and over 1,000 variants from 153 different families of mobile threats were also added to the database. The company said the growing popularity of the Android platform in particular, has drawn the cybercriminals attention, adding that in August 2010, the first malicious programme targeting Android was detected, and since then, that number has reached 15 programmes from a total of 7 families. "The first threats targeting Apple's iPhone OS also appeared during this last reporting period, but infected only devices that had been jail-broken in order to install third-party games and other software not manufactured by Apple," Kaspersky said. It noted that most mobile threats continue to target the Java 2 Micro Edition (J2ME) platform, which is supported by a huge number of mobile devices, saying, that means it is not only smartphones that are at risk of infection, but basic mobile phones as well. The records showed that the second most-targeted platform is Symbian, with Python in third place. Senior Malware Analyst at Kaspersky Lab, Denis Maslennikov was quoted as saying "the use of SMS Trojans is still the easiest and most effective means by which malicious users can earn money."

[Click here to Post comment](#)

This is a developers blog

[See the Archive](#)

[RECENT POSTS](#)

[The World Today](#)

[Developers in the verge of merging](#)

[Facebook vrs Twitter](#)

[Techbology on the ups!](#)

[Open source development](#)

[Developers to launch new software soon](#)

[New office open soon](#)

Title

Great post

Name

Peter mensah

Email

Pakspub@yahoo.com

Location

Iceland

Message

you doing a good work-lovely
and keep it up and will be
great to follow your posts

A blog by Prince Akyereko

Screenshot 7: A visitor (Registered) writing comment on a post

From the (Screenshot 7) a visitor called "Peter Mensah" is commenting on the post. He inputs his name as "Peter Mensah", title as "Great Post", location as "Iceland" and adds his email address as "pakspub@yahoo.com. After submitting this input, it appears under the other comments already entered in the system (Screenshot 8) below shows the outcome after Peter Mensah has submitted his comment into the system.

Developers Blog

Techbology on the ups!

Anti-virus software vendor, Kaspersky Lab says that the number of malicious programmes targeting mobile devices has more than doubled between August 2009 and December 2010. Records of Kaspersky Lab's virus database for 2010 posted on their website indicated that last year, over 65% more new threats targeting mobile devices were detected than in the previous year; and over 1,000 variants from 153 different families of mobile threats were also added to the database. The company said the growing popularity of the Android platform in particular, has drawn the cybercriminals attention, adding that in August 2010, the first malicious programme targeting Android was detected, and since then, that number has reached 15 programmes from a total of 7 families. "The first threats targeting Apple's iPhone OS also appeared during this last reporting period, but infected only devices that had been jail-broken in order to install third-party games and other software not manufactured by Apple," Kaspersky said. It noted that most mobile threats continue to target the Java 2 Micro Edition (J2ME) platform, which is supported by a huge number of mobile devices, saying, that means it is not only smartphones that are at risk of infection, but basic mobile phones as well. The records showed that the second most-targeted platform is Symbian, with Python in third place. Senior Malware Analyst at Kaspersky Lab, Denis Maslennikov was quoted as saying "the use of SMS Trojans is still the easiest and most effective means by which malicious users can earn money."

4 Comments »

- Computer Virus**

posted by: Andy Kojo

April, 2, 2011 @ [2:38 am](#)

Location: Berekum

this is it.

Here is it;s

Kofi goes to school:
- Here is my thought**

Posted by: Sefa Kobi

April 5, 2011 @ [21:04 pm](#)

Location: Helsinki

nice ad lovely post...i like all your posts
- Good Work**

Posted by: Akyereko

April 8, 2011 @ [12:44 am](#)

Location: London

Viruses are very dangerous and needs attention. great post
- Great Post**

Posted by: Peter Mensah

April 10, 2011 @ [14:41 pm](#)

Location: Iceland

you doing a good work – lovely

and keep it up and will be great to follow your posts

[Click here to Post comment](#)

This is a developers blog

[See the Archive](#)

[RECENT POSTS](#)

[The World Today](#)

[Developers in the verge of merging](#)

[Facebook vrs Twitter](#)

[Techbology on the ups!](#)

[Open source development](#)

[Developers to launch new software soon](#)

[New office open soon](#)

A blog by Prince Akyereko

Screenshot 8: After visitor (Registered) called "Peter Mensah" has submitted his comment on a post

After submitting a successful comment using the form in (Screenshot 7) above, the registered user gets an email notification that your comment been inserted into the database. It uses the PHP mail function to do that job. The email that will be sent is made up of four parts: Address, Subject, Body and Headers. The function as specified in the code works like mail (to address, subject, body, headers). The address is where the email will be sent to which should be the registered user address, then the subject will be like the heading of the message, also the body will be the main message being sent and finally the header function creates an http header which basically is a guide to the web server to act. It also prevents readers from sending their comments twice.



Screenshot 9: Searching for post result in the system by a visitor called "Kojo"

Another thing a visitor or registered user can do with the system also is searching for post. Let say there is a visitor call 'KOJO' and wants to search for post. "KOJO" types in his message and the search result will be what we see in Screenshot 9 above.

Another function the administrator can perform in this system is flagging

inappropriate comment and finally banning the user from the system. The logic of flagging a user and banning him/her from the system is represented in (Figure 9 in Appendix 2) activity diagram of flagging and banning user. The first inappropriate comment is flagged and administrator will send a warning to the user. If the user posts inappropriate comment for the second time, the user is removed from the system and cannot comment again in any future posts.

4.1 Creating Really Simple Syndication (RSS) Feed for the Blog

It is important that the blog has RSS feed in order to tell the outside world anytime the blog has been updated so that the readers do not have to keep on coming back to the website to find out if it is updated or not. It is basically an XML file (index.xml) in Appendix 1. It can be read online through other websites. Again, it can also be read using news reader software. There are many different news readers available for all operating systems in recent times.

For sever constraints and other security reasons I would like to create a static file that is updated only when I make a change to my blog. Traditionally static files do not need a database connection and my site will be more easily to handle a lot of inquisitive RSS software.

To create this RSS feed I need to create a new document and name it index.xml and save it to my root directory and need to edit my operating system to give it a read-write privileges if it is not a Windows. After writing this code to write the RSSfile, it is needed to tell the world that you have an RSS feed (index.xml).

5 TESTING THE APPLICATION

The testing of the blog was tested by fellow colleagues and friends both in school and at the company. They were to start using the application and give the outcome. They first did the layout testing and function testing. The results are those shown based on their feedback.

5.1 Layout Testing

Table 2: Layout testing

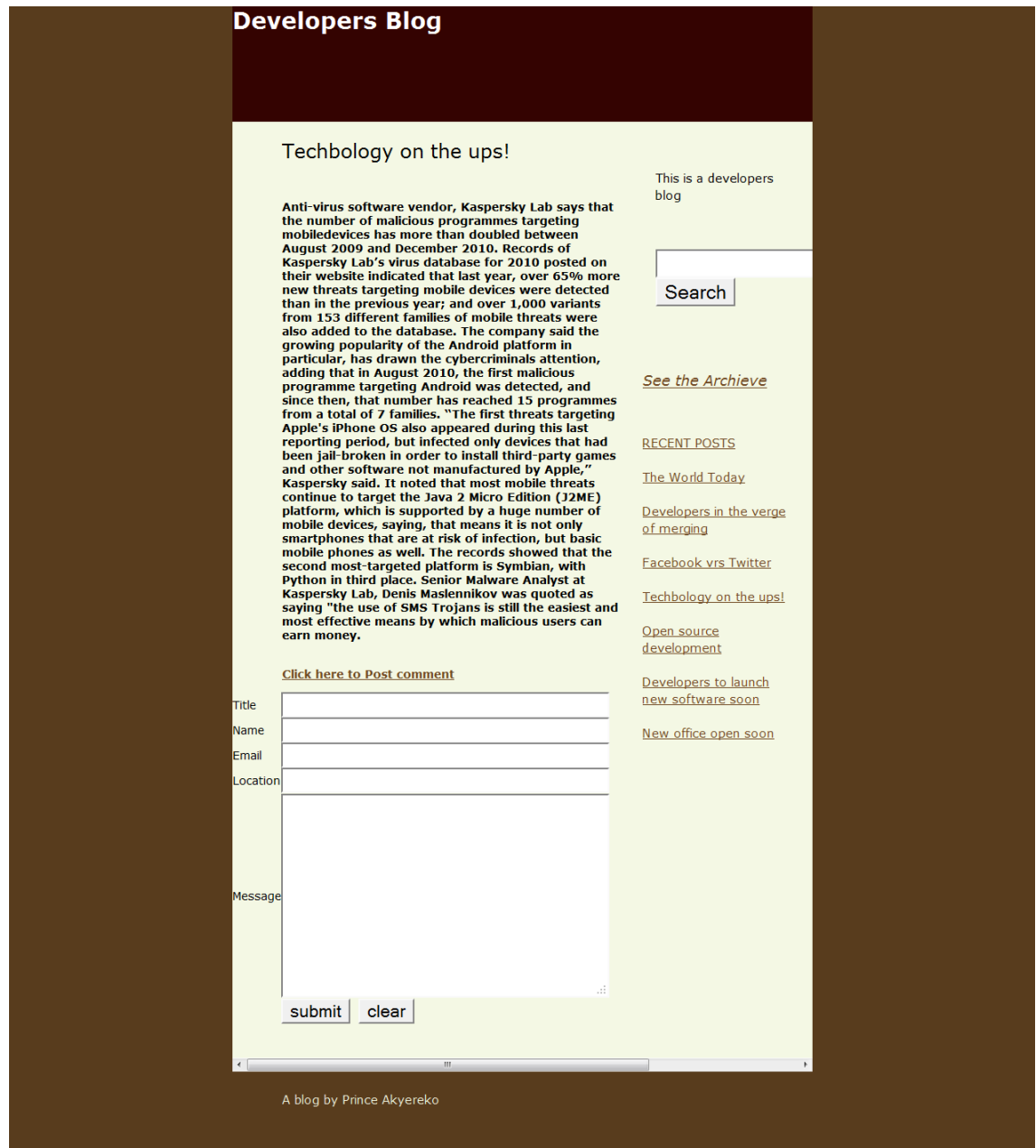
Browser → Test Object ↓	Internet Explorer 8.0	Firefox 3.6	Google Chrome	Safari 5	Opera 10.63
Font Size	okay	okay	too little than expected	too big	okay
font visibility	okay	okay	okay	okay	okay
picture visibility	okay	okay	okay	okay	okay
picture overlaps	okay	okay	okay	okay	okay
picture loading	okay	okay	okay	okay	okay
colours	okay	okay	okay	okay	okay
windows and pages resizing	okay	okay	okay	okay	okay



Screenshot 10: Layout testing in Internet Explorer



Screenshot 11: Layout testing in Google Chrome



Screenshot 12: Layout testing in Firefox



Screenshot 13: Layout testing in Safari



Screenshot 14: Layout testing in Opera

Google Chrome shows the text too little to see and this could be done by changing Chrome font-size setting to adhere to font size use in this blog. The Safari five also shows the text too big in the browser and this could be solved by

changing the font-size setting also here. The rest of the browsers tested almost all the layout requirements okay and there are no problem using them in those browsers.

5.2 Functional Testing

We need to do this testing because is an important and integral part of development. This testing helps us to know that the blog pages built in features work as expected. Customers and visitors alike become dissatisfy when the blog gives them error and may influence them how often to visit our website.

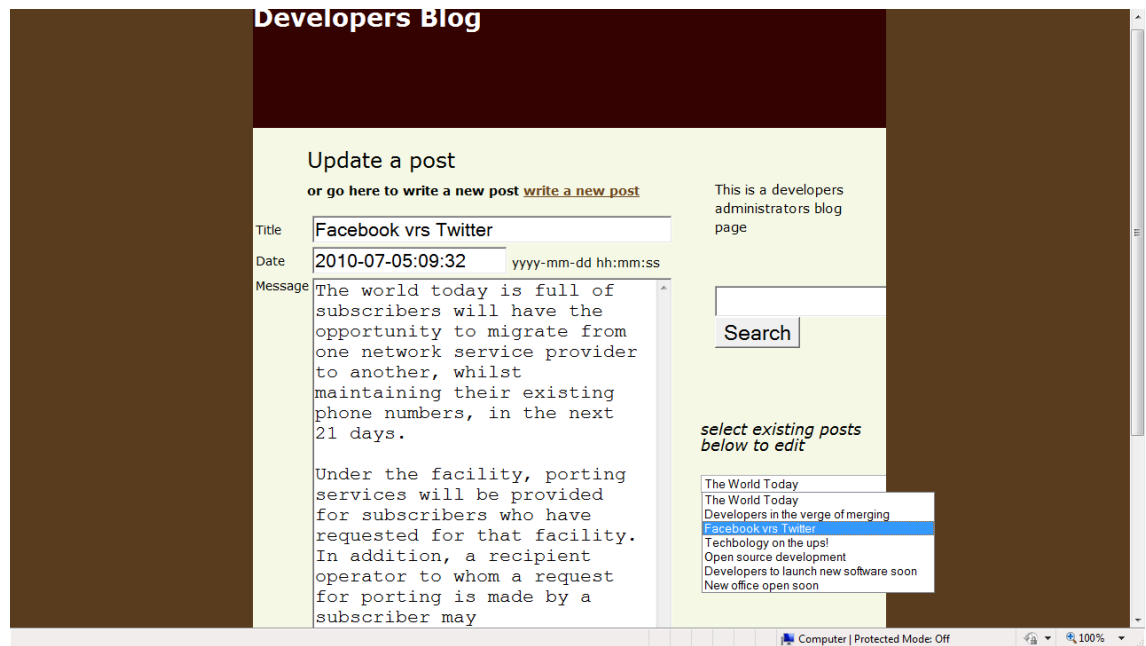
Among of those needed to be tested in this functionality testing is:

Compose post form:
 Links:
 Post selection:
 Adding posts:
 Updating post:
 Uploading pictures in the post:

The functionality testing results are shown in Table3.

Table 3: Functional testing

Browser → Test Object ↘	Internet Explorer 8.0	Firefox 3.6	Google Chrome	Safari 5	Opera 10.63
compose post form and adding post	okay	okay	okay	okay	okay
Links	okay	okay	okay	okay	okay
post selection	okay	okay	okay	okay	okay
updating posts	okay	okay	okay	okay	okay
Comment form	okay	okay	okay	okay	okay
Search post button	okay	okay	okay	okay	okay



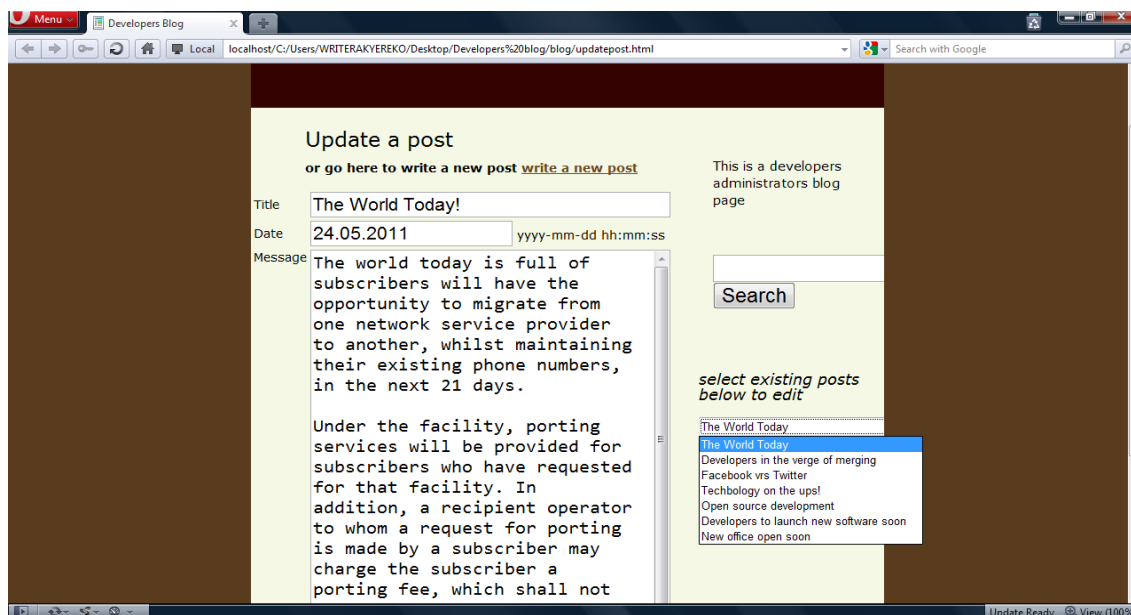
Screenshot 15: Update post in Internet Explorer



Screenshot 16: Compose form in Internet Explorer



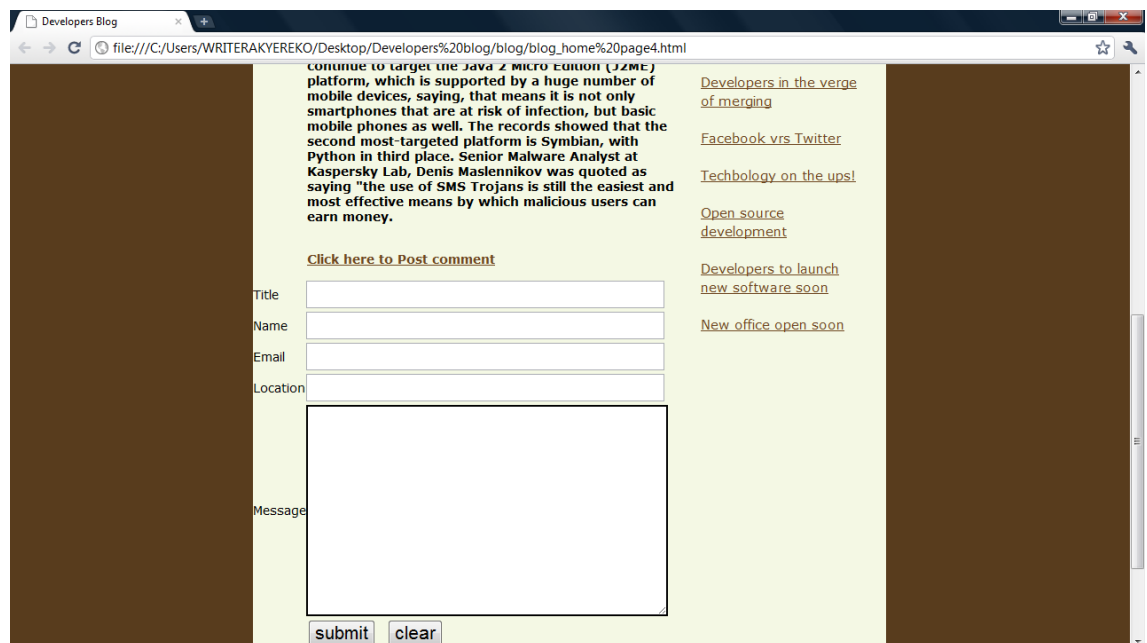
Screenshot 17: Compose post form in Safari



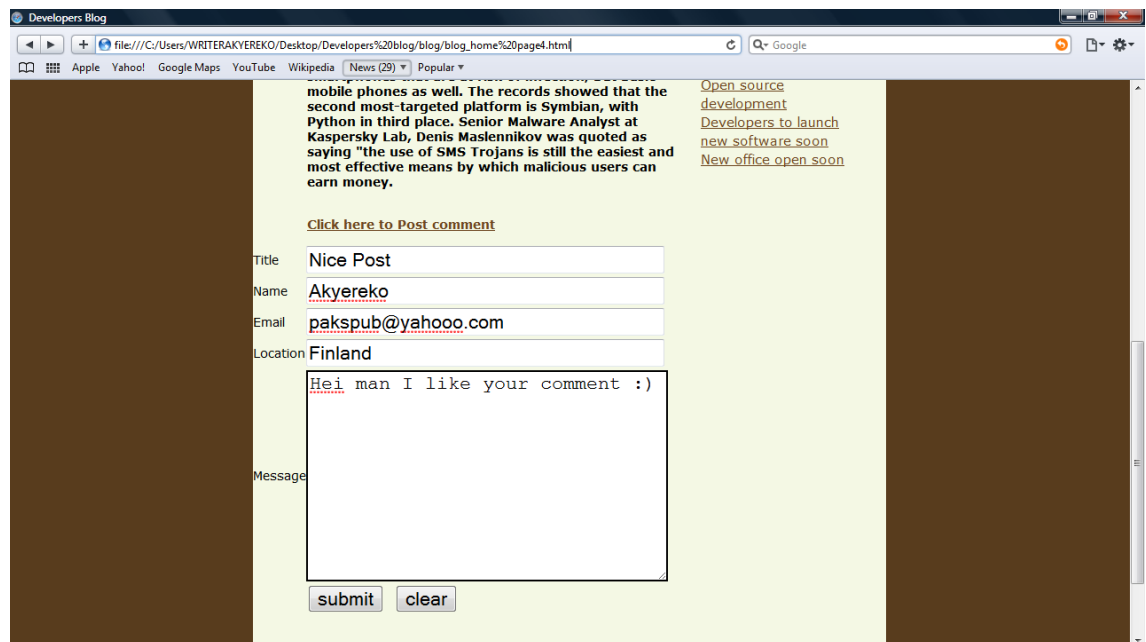
Screenshot 18: Updating post in Opera



Screenshot 19: Testing the search button and results in Opera



Screenshot 20: Post selection and comment form on Google Chrome



Screenshot 21: Post comment form and post selection in Safari

6 SUMMARY

It was clear from the project that blogs are becoming powerful way for individuals and organisations to express their views and share their experience with others. It is also an easy way for one to share their views on the internet as compare to going to other medium like newspapers and magazines. It is also cost effective way of communication as compare to newspapers and magazines.

The requirements were gathered through reading documentations, manual and interaction with other developers. The system has administration page, database and the weblog; where the post appears. The administrator uses the administration page to send a post into the weblog; lists the posts in sequential order with the recent ones appearing first, update, comment and delete post when necessary.

The blog homepage or weblog is where the post appear with the seven most recent ones appearing first and each post selected appear on a separate page in order to make the reading of the post very easily. This is where the visitor visits to access the materials in this blog system. The visitors are also able to read post, search and view achieve posts and comment on entries of their choice.

The testing area shows the outcome of the test done on the system to see how the various functions in the works. It was done in the browser to see how it would look like when in the browser depending on the type of browser. There are source materials and appendix areas where you see the referencing materials and tables, figures in the document respectively.

The availability of other blog engines available such as Text pattern, Word press, Expression Engine, Movable Type and others perhaps there was no need to build new blog from the scratch but it was done so because of the requirements of the project. However, it was good way of learning as building from scratch is more intensive and involves coding by the developer. All the

decisions of the application and building the database were done by the developer. So it was better way of learning than using pre-made template to do the blog.

SOURCE MATERIAL

Acroterion 2001-2007 consulted 18.09.2010 <http://blog.search3w.com/dynamicstostatic>.

Budd Andy, Collison Simon, Davis J.Chris, Heilemann Michael, Oxtan John, Powers David, Rutter Richard, Sherry Phil. 2006, Blog Design Solutions, USA, Apress Company.

Chen, J. (2010, April 19). This is apple's next iphone. Consulted 05.05.2011 <http://gizmodo.com/5520164/this-is-apples-next-iphone>

Developers Helsinki Oy 2009 consulted 01.09.2010 www.developers.fi.

The PHP Group 2001-2010 consulted 04.01.2010, 03.02.2010, 17.12.2010, 10.10.2010 www.php.net

Mediawiki, Creative Commons Attribution-ShareAlike 2002 consulted 15.08.2010 <http://en.wiktionary.org/wiki/blog>

W3school Group 1999-2010 consulted 02.12.2009, 19.09.2010 www.w3school.com

Miller, E., & Pole, A. (2010). Diagnosis Blog: Checking Up on Health Blogs in the Blogosphere. American

Wikimedia Foundation, Florida, USA consulted 05.09.2010 <http://en.wikipedia.org/wiki/Blog>

Wetzel, T (2010, October). To Blog or Not to Blog. Rough Notes, 153(10), 62, 64. Consulted. May 24, 2010, from ABI/INFORM Global. (Document ID: 2164426091).

APPENDIX 1.1 Application Codes

Table 4: Compose Post Form (HTML)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html Lang="English" xml:lang="English" xmlns="http://www.w3.org/1999/xhtml">
<html>
<head>
<meta http-equiv="content-type"
content="text/html; charset=iso-8859-1"/>
<title>Developers Blog</title>
<link rel="stylesheet" type="text/css" href="css_blog.css"/>
</head>
<body>
<h1>Developer's blog area</h1>

  <div id="blog_content"> <form method="post" action=" blog_post.php">
<h2>Write a new post</h2>
<table class="form_container">
  <tr>
    <td class="firstname">Title</td>
    <td class="firstinput"><input type="text" name="headline"
maxlength="140"
class="input_wide" value=""></td>
  </tr>
  <tr>
    <td class="Date">Date</td>
    <td class="firstinput"><input type="text" name="postdate" size="40"/> yyyy-mm-dd hh:mm:ss
  </tr>
  <tr>
    <td class="firstname" valign="top">Message</td>
    <td class="firstinput"><textarea name="message" cols="100" rows="20"
id="message"></textarea></td>
  </tr>
  <tr>
    <td class="firstname">&nbsp;</td>
    <td class="firstinput"><input type="submit" value="Submit" class="input_submit">&nbsp;<
<input type="reset" name="Reset" value="Clear">
  </td>
  </tr>
</table>
</body>
</html>
```

APPENDIX 1.2 Application Codes

Table 5: Sending Post into the Database (blog_post.php).

```
<?php
if (isset($_POST['message']) && isset($_POST['title'])) {
if (!isset($_GET['id']) || !is_numeric($_GET['id'])) {
db_insert("insert into blog_post set
created_at='". date ("Y-m-d H:i:s", time()) ."',
title="'. db_escape ($_POST['title'])."',
message="'. db_escape($_POST['message']) .'"");
echo "<div class=\"success\">A new post was successfully entered <br> <a
href=\"\".cfg_web_path.\"blogs\">Return to blog home page</a></div>\n";
} else {
db_insert("update blogs set
title="'. db_escape($_POST['title'])."',
message="'. db_escape($_POST['message']) .'"
Where id=". $_GET['id'];
echo "<div class=\"success\">Your post was edited successfully <br> <a
href=\"\".cfg_web_path.\"blog_posts\">Return to blog home page</a></div>\n";
}
require(cfg_path_modules . "footer.php");
}
if (isset($_GET['id']) && is_numeric($_GET['id'])) {
$sql = db_select("select title, message from blog_post where id=". $_GET['id']);
$row = mysql_fetch_assoc($sql);
$link = "? id=". $_GET['id']; $title = "Edit post";} else {
$row['title'] = "";
$row['message'] = "";
$link = "";
$title = "Write a new post";}??
<form method="post" action="<?=cfg_web_path?>blogs/write<?=$link?>">
<h2><?=$title?></h2>
<table class="form_container">
<tr>
<td class="firstname">Title</td>
<td class="firstinput"><input type="text" name="headline" maxlength="140" class="input_wide"
value="<?=stripslashes($row['headline'])?>"></td></tr><tr>
<td class="firstname" valign="top">Message</td>
<td class="firstinput"><textarea name="message" cols="100" rows="20"
class="textarea_big"><?=stripslashes($row['message'])?></textarea></td></tr><tr>
<td class="firstname">&nbsp;</td>
<td class="firstinput"><input type="submit" value="Submit" class="input_submit">&nbsp;<
<input type="reset" name="Reset" value="Clear">
</td>
</tr>
</table>
</form>
```

APPENDIX 1.3 Application Codes

Table 6: PHP Code that lists all posts on the Page. (listing_post.php).

```
<?php
include("../db_connect.php");
$sql= "select blog_post_id, title, message, created_at(postdate, '%e %b %Y at %H:%i) AS
daytime
From blog_post
Order by postdate DESC";
$result=mysql_query($sql);
$blog_post= mysql_fetch_array($result);?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html lang="english" xml:lang="english" xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type"content="text/html; charset=iso-8859-1"/>
<title> Posts posted already-Developers blog</title> </head>
<body>
<h1> Posts posted already</h1>
<?php
if (isset($message)){
echo "<p class='message'>".$message$."</p>";}
if($blogpost){echo "<ol>\n";
do{$blog_post_id=$blogpost["blog_post_id"];
$title=$blogpost["$title"];
$daytime= $blogpost["daytime"];
echo "<li value='$blog_post_id'>";
echo "<a href='blog_post.php?blog_post_id=$blog_post_id'>$title</a>
posted $daytime";
echo "</li>\n";
}
while ($blogpost = mysql_fetch_array($result));
echo"</ol>";
}
else {
echo "<p> Could not find any entry entered on the website.</p>";
}
?>
</body>
</html>
```

APPENDIX 1.4 Application Codes

Table 7: Commenting on post code

comment_post.php
<pre> <form action = "<? =\$_SERVER ["PHP_SELF"]?>" method="POST"> <input type="hidden" name="blog_post_id" value=" <=? blog_post_id?>" /> <input type="hidden" name="title" value="<? =\$title?>" /> <h2>write comment</h2> <p> Name: <input name="name" type="text" /></p> <p>Email: <input name="email" type="text" /></p> <p>website: <input name="website" type="text" /></p> <p>Comment: <textarea name="comment" cols="30" rows="15"> </textarea></p> <input type="submit" value="Submit" /> </p> <p>&nbsp; </p></form> comment.php if (isset(\$message)) {echo "<p class='message'>".\$_POST ["message"]. "</p> "; } ?> if(isset(\$_POST["add comment"]) != "") { \$title = addslashes(trim(strip_tags(\$_POST["title"]))); \$name = addslashes(trim(strip_tags(\$_POST["name"]))); \$email = addslashes(trim(strip_tags(\$_POST["email"]))); \$location = addslashes(trim(strip_tags(\$_POST["location"]))); \$website = addslashes(trim(strip_tags(\$_POST["website"]))); \$comment = addslashes(trim(strip_tags(\$_POST["comment"]))); \$sql="insert into comments (blog_id, name, email, location, website, comments) values('blog_post_id', '\$name', '\$email', '\$location', '\$website', 'comments'); \$results =mysql_query(\$sql); if(!\$results) { \$message="comment successfully added.";} else{ \$message= "Comment can't be added."; \$comment_id=mysql_insert_id(); \$emailheading = "comment is successfully posted to: ".\$title; \$emailbody = "Comment on '". \$title. " ' . "\n" "http://www.developers.fi/blogpost.php?blog_id= " . \$blog_post_id ."#c" .\$comment_id. "\n\n" .\$comment. "\n\n" .\$name. " (".\$website.")\n\n"; \$emailbody =stripslashes(\$emailbody); \$emailheader= "From: ".\$name. "<". \$email.">\n". "Reply-To: mail(prince@developers.fi", \$emailheading,\$emailbody,\$emailheader; header("Location:blog_post.php?blog_post_id=blog_post_id&message=\$message"); } </pre>

APPENDIX 1.5 Application Codes

Table 8: Deleting Post

delete.php
<pre> echo "<li value ='\$blog_post_id'>"; echo "< a href=' blog_post.php?blog_post_id=\$blog_post_id'>\$title"; echo " posted \$datetime"; echo " [< a href=' ".\$_SERVER["PHP_SELF"]."?delete=\$blog_post_id'"; echo " onclick='return confirm(\"confirm delete?\")'>delete]"; echo "\n"; and below is the delete script which should be inserted just after the db_connection code. \$delete =(isset(\$_GET["delete"]))? \$_GET["delete"]:""; if(preg_match("/^[0-9]+\$/", \$blog_post_id)) { \$sql= "DELETE FROM blogs WHERE blog_post_id=\$delete LIMIT 1 "; \$result=mysql_query(\$sql); if(! \$result) { \$message= "blog post deleted successfully."; } else{ \$message= "Could not delete post \$delete, try again. ".mysql_error(); </pre>

APPENDIX 1.6 Application Codes

Table 9: HTML code and PHP code that creates the Blog Home page (index.php)

```
<?php
include("functions.php");
// Open connection to database
include("db_connect.php");
// will select 7 most recent posts
$sql = "SELECT blog_post_id, title,message, date_time_set(postdate,'Y-m-d\TH:i:s\Z');
AS datetime FROM blog_post ORDER BY postdate DESC LIMIT 7";

$result = mysql_query($sql);
$blogpost = mysql_fetch_array($result);
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html
xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
<link          rel="stylesheet"          type="text/css"          href="css_blog"/>

<title>Developer's Blog</title>

</head>
<body>

<?php include("header.php"); ?>

<!-- this is where the posts are displayed -->

<div id="blogcontent">

<div id="blog_post">

<?php
if($blogpost) {
do {
    $blog_post_id = $post["blog_post_id"];
    $title =
$post["title"];
    $post = format($blogpost["message"]);
    $datetime = $post["datetime"];

echo "<h2 id='message'>$blog_post_id'>
<a href='blog_post.php?blog_post_id=$blog_post_id' rel='bookmark'>$title</a></h2>\n";

echo "<h4>Posted on $datetime</h4>\n";
```

```

echo "<div class='message'>$message</div>";
}
while ($blogposts = mysql_fetch_array($result));
}
else {
    echo "<p> Not yet posted onto the blog area.</p>";
}
?>
</div>

<div id="sidebar">
<div id="about">
<h3>About this</h3>
<p>This is a Developer's blog.</p>
</div>
<?php include("search.php"); ?>

<div id="current">
<h3>current posts</h3>
<?php
mysql_data_seek($result, 0);
$blogpost = mysql_fetch_array($result);
if($blogpost) {
    echo "<ul>\n";do {
        $blog_post_id = $blogpost["blog_post_id"];
        $title = $blogpost["title"];
        echo
        "<li> <a href='blog_post.php?blog_post_id=$blog_post_id' rel='bookmark'>$title</a></li>\n";
    } while ($blogpost = mysql_fetch_array($result));

echo "</ul>";}
?></div>
</div><!-- sidebar div ends here -->
</div><!-- blogcontent div ends here -->
<?php include("footer.php"); ?>
</body>
</html>

```


APPENDIX 1.7 Application Codes

Table 10: XML FEED FILE (**blogfeedindex.xml**)

```
function developersfeed()
//set the file to write
{$filename = $_SERVER["DOCUMENT_ROOT"] . "/index.xml";
//open file
$fh=@fopen($filename, "w");
if($fh){$rssfile = "<rss version=\"2.0\">
<channel>
<title>Developer's Blog</title>
<link>http://www.developers.fi/blog</link>
<description>A blog by Developers Helsinki</description>
<language>en-gb</language>";
//this pull blogs from the database
$sql="SELECT blog_post_id,title,message,
DATE_FORMAT(postdate,'%a,%d,%b %Y %T GMT')as created_at
FROM blog_post ORDER by postdate DESC LIMIT 6";
$result = mysql_query($sql);
if($blogpost = mysql_fetch_array($result)){do {
$blog_post_id = $blogpost["blog_post_id"];
$created_at = $blogpost[created_at];
$message = format($blogpost["message"]);
$title=$blogpost["title"];
$title=strip_tags($title);
$title=htmlentities($title);
$rssfile .= "    <item>\n";
$rssfile .= "    <created_at>$ created_at</created_at>\n";
$rssfile .= "    <title>$title</title>\n";
$rssfile .= "    <link>http://www.developers.fi/blog_post.php?blog_post_id=$blog_post_id</link>\n";
$rssfile .= "    <description><![CDATA[$message]]></description>\n";
$rssfile .= "    </item>\n";
}while ($blogpost = mysql_fetch_array($result));} $rssfile .= " </channel></rss>";
//Write to file
$fw=@fwrite($fh, $rssfile);
if (!$fw){
$echomessage = " Could not write to the file $filename";}else{$echomessage ="RSS file updated.";}
//close file
fclose($fh);
}else{
$echomessage="Could not open file $filename");
}
return $echo message;
}
<lilinkrel="alternate"          type="application/rss+xml"          title="RSS          2.0          feed"
href="http://www.developers.fi/blog/index.xml into just after <title> of my blog homepage
<span class="style2"></span>.
```

APPENDIX 1.8 Application Codes

Table 11: CSS File for the Application. (css_blog.css)

```
body {
    margin: 0;
    padding: 0;
}
#blogcontent, #header, #footer {
    width: 750px;
    margin: 0 auto;
}
#header {
    height: 152px;
}
#footer {
    padding: 15px 0 3 0;
}
#blogcontent {
    height: 1%;
    overflow: auto;
    padding-bottom: 3;
}
#blogposts {
    width: 500px;
    float: left;
}
#sidebar {
    margin-left: 550px;
    padding-right: 30px;
}
h1 a {
    display: block;
    width: 390px;
    height: 150px;
    text-indent: -1000em;
    margin: 0;
}
/* font colours */
body {
    font-family: Arial, Verdana, sans-serif;
}
;
font-size: 100%;
}
html>body {
    font-size: 14px;
}
h1, h2, h3 {
    margin: 0;
}
h2 {
    font-size: 1.7;
    font-weight: normal;
}
```

```

        line-height:1;
        padding-top: 1;
    }
    h2 a {
        text-decoration:none;
    }
    #blogposts h3 {
        font-weight:normal;
        line-height:1;
        font-size: 1.5;
        margin-top: 1;
    }
    h4 {
        font-weight:normal;
        font-style:italic;
        line-height:1.5;
        font-size: 1.3;
    }
    .post, dl, #posts UL, #comments P, #results P {
        font-size: 1.3;
        line-height:1.5;
    }
    #comments dd p {
        font-size: 1;
        padding-left: 0;
    }
    dd {
        margin-left: 0;
    }
    #blogposts h2, #blogposts h3, #blogposts h4, #blogposts h4, .post, DL, #blogposts UL,
    #comments P, #results P {
        padding-left: 61px;
    }
    .post p {
        margin-top:0;
    }
    html>body .post p {
        margin-bottom: 0;
    }
    .post P+P {
        text-indent: 1.5em;
    }
    .post {
        padding-bottom:1em;
    }
    #sidebar {
        font-size: 1.1em;
        line-height: 1.3636em;
    }
    #sidebar p, #sidebar UL {
        margin-left: 16px;
        padding-left: 0;
    }
    #sidebar li {
        list-style: disc;
        margin-left: 16px;
        margin-bottom: 0.70em;
    }

```

```

#sidebar h3 {
    margin: 1.46em 0 0.682em 15px;
    text-indent: -980em;
    overflow: hidden;
}
#sidebar #archive {
    text-indent: 0;
}
#archive a {
    display: block;
    text-indent: -980em;
}
#footer {
    text-indent: 62px;
    font-size: 1.1em;
}
/* colours */
body {
    background: #A0522D repeat fixed center;
    color: #CD5C5C;
}
A {
    color: #ADFF2F;
}
H2 A {
    color: #000000;
}
.post {
    background: no-repeat left 3px;
}
#header {
    background: #800000 no-repeat;
    color: #fff;
}
#footer {
    background: no-repeat;
}
#blogcontent {
    background: #FFFFFF repeat-y fixed center 111px;
    color: #000000;
}
#about h3 {
    background: no-repeat;
    width: 141px;
    height: 19px;
}
#search {
    background: no-repeat;
    width: 57px;
    height: 16px;
}
#current h3 {
    background: no-repeat;
    width: 135px;
    height: 20px;
}
#archive {
    background: no-repeat ;

```

```
        width: 128px;
        height: 15px;
    }
    #addcomment h3 {
        background: no-repeat ;
        width: 120px;
        height: 15px;
    }
    /* Forms */
    input, textarea {
        border-width: 1px;
    }
    input[type=text], textarea {
        width: 160px;
    }
    input, textarea {
        font-size:1;
    }
}
```

APPENDIX 1.9 Application Codes

Table 12: Blog Home Pages (HTML)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html lang="english" xml:lang="english" xmlns="http://www.w3.org/1999/xhtml">
<html><head><meta http-equiv="content-type"content="text/html; charset=iso-8859-1"/>
<title>Developers Blog</title><link rel="stylesheet" type="text/css" href="css_blog.css"/>
</head>
<body>
<div id="header"><h1>Developers Blog</h1>
</div><div id="blogcontent"><div id="blogposts"></div>
<div id="sidebar">
<div id="about">
<h2>About Developers</h2>
<p>This is a developers blog</p>
</div>
<form action="search.php" method="get">
<h3 id="search">Search</h3><p>
<input type="text" name="button" value="">
<input type="submit" value="Search"/>
</p></form>
<h3 id="archievet"><a href=archievetblog.php">See the Archievet</a></h3>
<div id="archievet"> <h2>Archievet</h2>
</div>
</div>
</div>
```

APPENDIX 1.10 Application Codes

Table 13: Headers and Footers. It has both header.php and footer.php

```
header.php
```

```
<div id="header">  
<h1><a href="index.php">Developers Blog</a></h1>  
</div>
```

```
footer.php
```

```
<div id="footer">  
<p>A blog by Developers Helsinki oy.</p>  
</div>
```

APPENDIX 2 Other UML Diagrams of the System

Use Case Name:	New Account Creation	
Scenario:	Not Registered User Wants to Create Account	
Triggering Event:	Unregistered User selects blog website Account Creation Form	
Brief Description:	Not registered user selects registration form on blog website. Not registered user inputs his/her information require by the form. Once user submits form, system will check for errors in form. If successful, system will send out a confirmation email, which the user must respond to. If unsuccessful, system will require user to input his/her account information again. Once user responds to confirmation email, the user becomes a registered user who is allowed to comment on post, search for post and enjoy most aspect of the system	
Pre-conditions:	Not registered user must exist., Account creation system must exist	
Post-Conditions:	1. Account must be created. 2. User must be able to log into blog website. 3. User must have account privileges that are assigned for registered users	
Flow of Events:	Actor	System
	1. Not registered user accesses account creation form	1.1 System displays form.
	2. Not registered user fills out account information	
	3. Not registered user submits completed form	3.1 System checks for errors and required fields.
	4. Repeat steps 2 and 3 only if system finds errors	4.1 System stores information. 4.2 System sends confirmation email. 4.3 System creates an inactive account until user responds to confirmation email.
	5. User responds to confirmation email by clicking link provided in email.	5.1 System confirms account by allowing full access to registered user's account. 5.2 System displays successful confirmation page
Exception Conditions:	3.1 If unregistered user inputs an invalid username, email, or password, then unregistered user must choose a new one. 3.2 If unregistered user closes form window, then system does not register or store any information input by the user. 5.1 If unregistered user does not receive confirmation email, then user can request confirmation email to be resent by attempting to log in	

Figure 7: Account Creation Use Case and Description

APPENDIX 2 Other UML Diagrams of the System

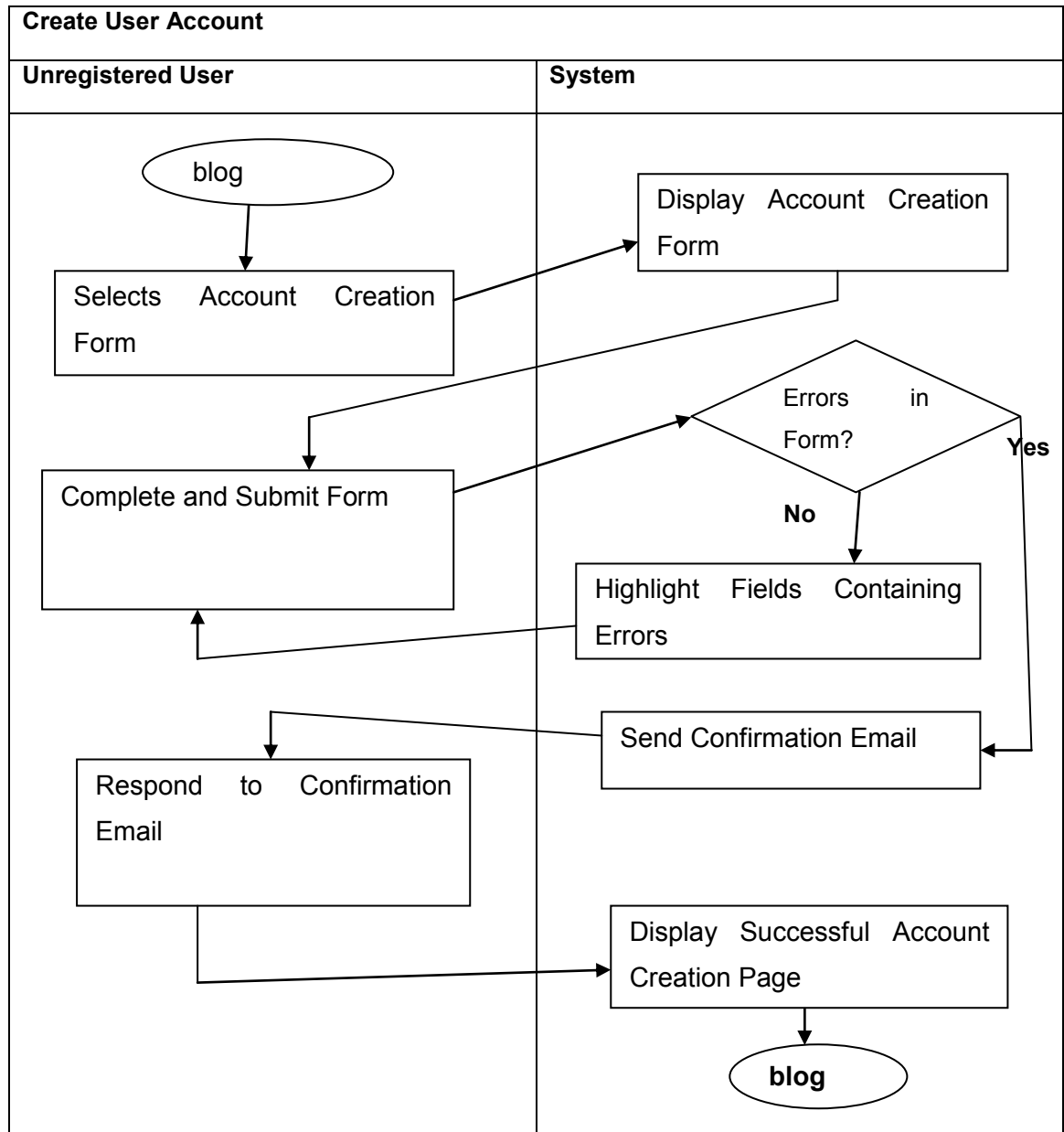


Figure 8: Activity Diagram of Creating User Account

APPENDIX 2 Other UML Diagrams of the System

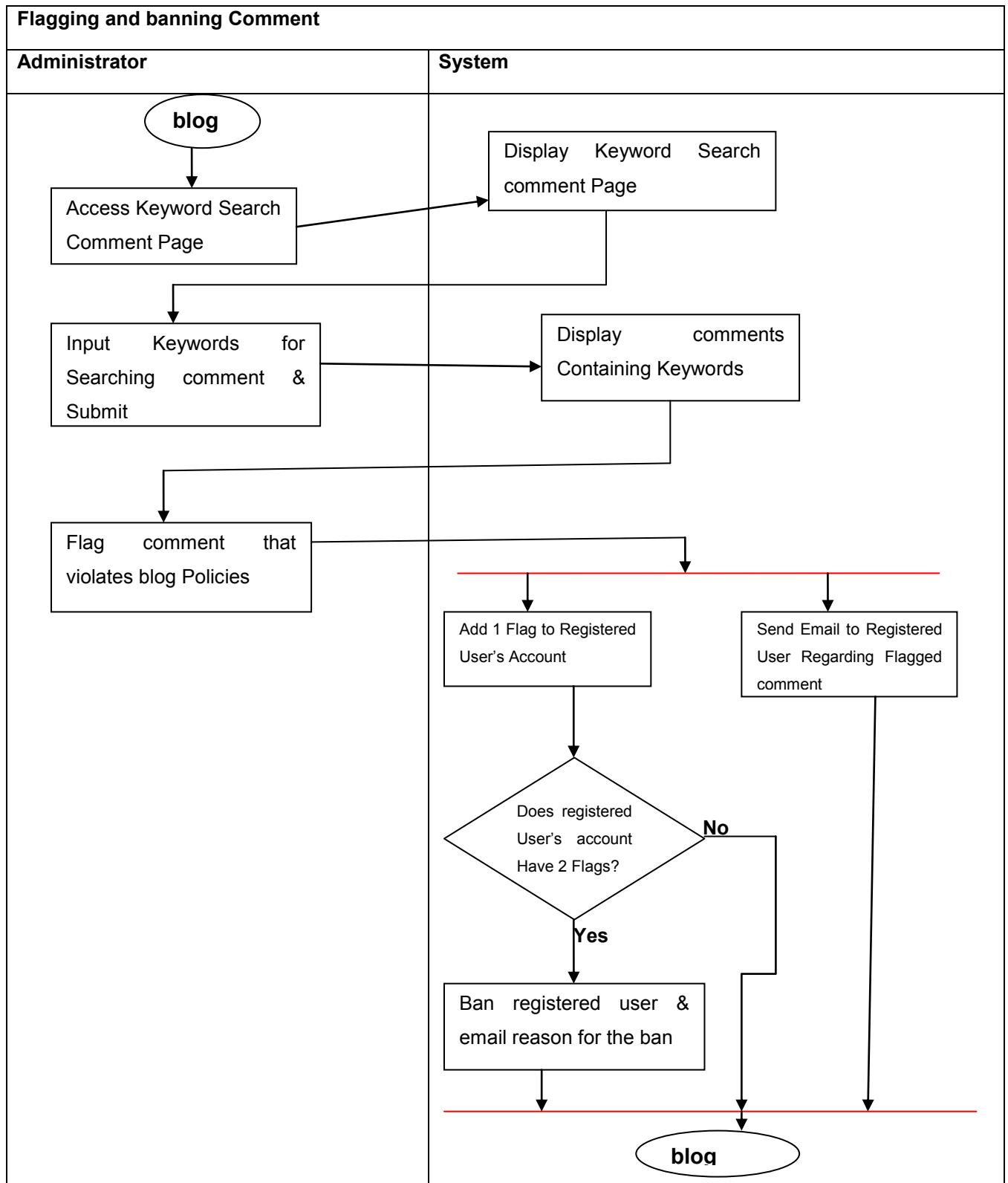


Figure 9: Flagging and Banning Inappropriate Comment