OSEI TUTU II INSTITUTE FOR ADVANCED
ICT STUDIES, GHANA

Affiliated To

KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY,
KUMASI

SCHOOL OF GRADUATE STUDIES

"IMPLEMENTING A WEB APPLICATION FOR OSEI TUTU II INSTITUTE FOR ADVANCED ICT STUDIES"

A DISSERTATION PRESENTED TO THE
INSTITUTE IN PARTIAL FULFILMENT
FOR THE AWARD OF THE DEGREE OF
MASTER OF SCIENCE IN ICT STUDIES
(MANAGEMENT OPTION)

BY

NANA BAAH GYAN

OCTOBER 2009

I, Nana Baah Gyan, hereby declare that this submission is my own work towards the MSc and that, to the best of my knowledge, it contains no material previously published by another person nor material which has been accepted for the award of any other degree of the University, except where due acknowledgement has been made in the text.

NANA BAAH GYAN		21/10/09
(Student)	Signature	Date

Certified by:

(Supervisor)	Signature	Date

Certified by:

(Academic Officer)	Signature	Date

ABSTRACT

In earlier types of client-server computing, each application had its own client program which served as its user interface and had to be separately installed on each user's personal computer. An upgrade to the server part of the application would typically require an upgrade to the clients installed on each user workstation. In contrast, web applications use web documents written in a standard format such as (X)HTML, which are supported by a variety of web browsers. Generally, each individual web page is delivered to the client as a static document, but sequence of pages can provide an interactive experience. During a session, the web browser interprets and displays the pages and acts as the universal client for any web application. The ubiquity of web browsers and the convenience of using a web browser as a client, sometimes called a thin client, without the need to install software, make web applications desirable.

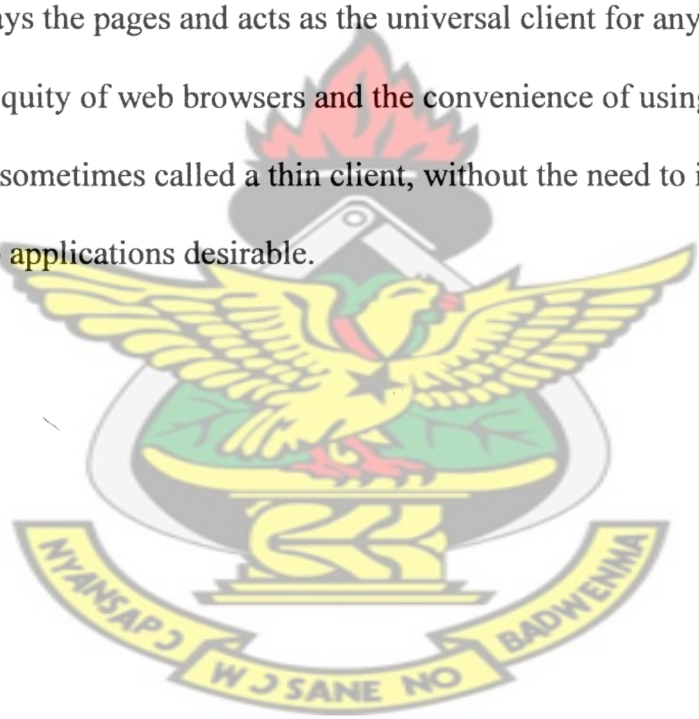


TABLE OF CONTENTS

Chapter Three

MOTIVATION FOR WEB APPLICATION FOR THE INSTITUTE

Chapter Four

DESIGN OF WEB APPLICATION SYSTEM

Chapter Five

IMPLEMENTATION OF THE WEB APPLICATION SYSTEM

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AJAX	–	Asynchronous JavaScript and XML
ASP	–	Active Server Pages
CSF	–	Critical Success Factor
CSS	–	Cascading Style Sheet
CSRF	–	Cross-site Request Forgery
DOM	–	Document Object Model
HTML	–	Hypertext Mark-up Language
HTTP	–	Hypertext Transfer Protocol
JSON	–	JavaScript Object Notation
JSP	–	Java Server Pages
OWASP	–	Open Web Application Security Project
PEAR	–	PHP Extension and Application Repository
PHP	–	PHP Hypertext Pre-processor
RAD	–	Rapid Application Development
RDBMS	–	Relational Database Management System
RIA	–	Rich Internet Application
SEO	–	Search Engine Optimization
UML	–	Unified Modelling Language
XHTML	–	Extended Hypertext Mark-up Language
XML	–	Extended Mark-up Language
XSS	–	Cross-site Scripting
ZF	–	Zend Framework
W3C	–	World Wide Web Consortium

ACKNOWLEDGEMENTS

My special thanks to Professor Vasil Fournadjiev, my able supervisor who guided me through every single page of this document.

I also say thank you to all the staff of the Osei Tutu II Institute for Advanced ICT studies especially the Rector, Professor Marten Looijen for all his help, the Vice Rector, Mrs Sophia Quarshie Sam and the Administrator, Ms Harmien Dam for their encouragement, support and advice.

To my parents for the love and care through all the difficult moments.

And finally to God Almighty who is the embodiment of knowledge and wisdom without whom this project would not have started at all.

1.0 INTRODUCTION

A web page is a document or resource of information that is suitable for the World Wide Web and can be accessed through a web browser and displayed on a computer screen. This information is usually in HTML or XHTML format, and may provide navigation to other web pages via hypertext links. Web pages may be retrieved from a local computer or from a remote web server. The web server may restrict access only to a private network, e.g. a corporate intranet, or it may publish pages on the World Wide Web. Web pages are requested and served from web servers using Hypertext Transfer Protocol (HTTP).

Web pages may consist of files of static text stored within the web server's file system (static web pages), or the web server may construct the (X)HTML for each web page when it is requested by a browser (dynamic web pages). Client-side scripting can make web pages more responsive to user input once in the client browser. Client-side computer code such as JavaScript or code implementing AJAX techniques can be provided either embedded in the HTML of a web page or, like CSS, as separate, linked downloads specified in the HTML. These scripts may run on the client computer, if the user allows them to, and can provide additional functionality for the user after the page has downloaded.

Chapter one introduces the context for this research. Here the motivation for this research is mentioned and discussed. That includes the context of research, the problem definition and the methodologies employed in the research.

1.1 RESEARCH CONTEXT

Since the evolution of the web, various institutions have sought of ways of utilizing the power of the web to accomplish various tasks. The power that the internet brings to board is the ability to remove distances that hitherto were barriers to cost-effective and timely communications between companies and prospective clients. Internet connectivity and penetration has seen a tremendous increase in the last decade around the world. There has also been a gradual but steady development in the technologies behind the web. These technologies have gone a long way to increase internet security (to some extent) and enhance privacy on the internet and user experience.

The technologies behind the web are both proprietary and open source with the latter seeing massive participation in the last few years. That has resulted in an evolved internet that can assure information security and privacy to a large extent. This thesis explores the practical usability of the technologies behind the web to design and implement a Web Application System (WAS) for a tertiary education institution in Ghana.

1.2 PROBLEM DEFINITION

The Osei Tutu II Institute for Advanced ICT Studies (The Institute) is an educational institution in Ghana, West Africa. The Institute offers Master of Science degree in the field of Information Communication Technology (ICT) in Ghana. Currently, most administrative and academic tasks within the Institute are only within its Local Area Network (LAN). Some of the tasks include:

- Submitting of results by lecturers (via post)
- Publishing of results of students
- Assessing of lecturers as part of quality assurance measures

- Creating (and editing) semester timetables

These tasks are routine processes such as developing a timetable for every new academic year. Also students are usually confined with the walls of The Institute such that once a student goes outside the premises of the school, there is hardly any medium to access useful information from authorities. Information on exam results, as an example, is only accessed within the premises of The Institute. The thesis seeks to investigate current processes within the Institute that could be effectively and efficiently undertaken through the WAS.

1.3 RESEARCH QUESTIONS

In undertaking the project, the following questions are asked:

- *Is the current website serving all the needs of the institute?*

This question seeks to find out what shortfalls there are with the current website. The weaknesses are examined carefully and the solutions found are used as the bases for the new web application.

- *What technologies are needed to implement the new website?*

This question seeks to find out the state-art-of-the-art software and programming languages in website building and development. Technologies such as JavaScript, AJAX, CSS, HTML, PHP and web application development frameworks are investigated. The technologies that are explored are then used to develop the WAS.

- *Are the systems to support the development of the website currently available in the Institute?*

This question seeks to find out how the developed system would be finally implemented for use by The Institute. It will take into consideration the necessary hardware and software support that would be required for the system's operation.

1.4 RESEARCH METHODOLOGIES

Interviews were conducted with the authorities of the Institute to solicit information on what the authorities of The Institute needed the WAS to be. This was necessary due to the fact that even though The Institute had an already working web site, the site was not seen to be serving all the needs of the Institute. The interviews produced useful inputs in the WAS in the form of information to be added on the new site, functionality that would be helpful and using it also to sell The Institute to prospective students.

Literature review was also undertaken to find out about the new trends in the field of web application development. The review also bothered on what some tertiary institutions in Ghana are doing with the web. That provided an appropriate platform upon which the new website for The Institute was built.

1.4.1 DATA COLLECTION METHODS

Interviews were also conducted with key stakeholders in charge of the day-to-day activities of The Institute. The interviews were conducted with:

- The rector of the Institute
- The vice rector of the Institute
- The academic officer of the Institute
- The facilities manager of the Institute

The interviews sought to solicit the goal of the Institute's website and how far that goal has been realized. Possible inclusion of functionality in the new web application was also solicited.

1.5 PROJECT OVERVIEW

The thesis attempts to provide information on the building of a web application for The Institute using state-of-the-art technologies vis-à-vis its motivation and relevance. The thesis is grouped into six chapters in all. Chapter one is the introductory chapter describing the basis for this research as well as the methodologies employed.

Chapter two takes a look at the literature review. With that a review of the currents trends in web application development is undertaken. Also in the review is a very brief analysis of what some tertiary institutions in Ghana are using the World Wide Web for, the online services they provide as well as the kind of information on their web sites. The review also takes into consideration what the standards are in the field of web application development.

Chapter three discusses the motivation behind the entire project and its relevance to The Institute. It takes a look at the current processes within The Institute and how they can be improved with the WAS. It also discusses the critical success factors (CSF) for the system.

Chapter four looks at the design aspect of the system. It is where all the various design of interfaces and databases are discussed. The system actors are introduced and thoroughly discussed. The so-called business logic of the system is also introduced and its design is looked at in this chapter.

Chapter five is the actual implementation of the system. It discusses the various mechanisms that are employed to design the WAS. Some of the mechanisms include input validation techniques, overall implementation of design pattern among others. The final chapter, chapter six, is the conclusion of the thesis.

2.0 LITERATURE REVIEW

2.1 INTRODUCTION

This chapter is introduced by a look at the websites of some tertiary institutions in Ghana. This is done because that picture will give a general overview of the general trend of using the World Wide Web in Ghana. The chapter goes on to introduce the various technologies that are employed to develop the WAS. Current trends (standards) and generally accepted methods in the field of web application development are explored. Also mentioned are the tools used in the implementation of the WAS. The roles that each tool plays in the system are also discussed.

2.2 WEBSITES OF SOME TERTIARY INSTITUTIONS IN GHANA

Universities in Ghana are used as the benchmark because that sets a common platform for analysis. This is particularly so because of infrastructural and technological differences that might be between Ghana and even other African countries. The criteria for the selection of these tertiary institutions (both public and private) were based on some factors. These factors are not based on research (as that kind of information is particularly hard to come by in Ghana). The factors include:

- The number of years the institutions has been in Ghana
- The programmes these institutions offer
- The popularity of the Institution

The analysis of these sites takes into consideration the information and services these sites are offering (if any), the actors that interact with the services (if any), and possibly to identify the technologies that are behind the sites. The knowledge of these then gives the idea of what the Institute's new web site should be like in terms

of information and services to be provided online. The selected tertiary institutions are:

1. The University of Ghana
2. The Kwame Nkrumah University of Science and Technology
3. The University of Cape Coast
4. The Ghana Telecom University College
5. The Ashesi University College

2.2.1 THE UNIVERSITY OF GHANA (UG) WEBSITE

The University of Ghana was founded in 1948 as the premiere tertiary institution in Ghana. With a current student population of about 29,754 (representing male/female ratio of about 2:1) the University of Ghana is the oldest and largest of the six public Universities in Ghana. Breakdown in terms of programmes and students as at year 2009 are as follows:

Table 2.1: Programmes and respective population of UG

Programme	**Number of students**
Post-Graduate	1,816
Bachelors'	26,154
Sub-Degrees	1,784
International Students	1,142
Senior members engaged in research and teaching	865
Senior administrative and Professional staff	128

A summary of information and services on the website is shown in the table below:

Table 2.2: Summary of information and services on UG website

Information on website	Services offered on website
Administration related information	Mail service
Academics/Admissions related information	E-learning portal
Alumni related information	Online search tool
Directories of staff	
News and upcoming events information	
Information on research	

2.2.2 THE KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY (KNUST) WEBSITE

KNUST is a technical university in Ghana. It is the second public university to be established in the country. It was founded as a college of technology in 1952 and was affiliated to the University of London. In 1961, the college was granted full university status. The table below summarizes the information and services offered on the website:

Table 2.3: Summary of information and services on KNUST website

Information on website	Services offered on website
Administration related information	Mail service
Academics/Admissions related information	Online results service
News and upcoming events	Online registration
Alumni information	Online forum
Research information	Staff web space
	E-learning
	Online search tool

2.2.3 THE UNIVERSITY OF CAPE COAST (UCC) WEBSITE

UCC is one of the public universities in Ghana. The University was established in 1962 out of a dire need for highly qualified and skilled manpower in education. Thus, it was established to train graduate teachers for second cycle institutions, Teacher Training Colleges, and Technical Institutions. From an initial intake of 155

students in 1962, the University now boasts of a student population (as at 2009) of about

- 17,000 regular students,
- 4,000 sandwich (summer) students and
- 20,000 distant learners.

The table below summarizes the information and services offered on the site:

Table 2.4: Summary of information and services offered by UCC website

Information on website	Services offered on website
Admission related information	Online library system being developed at the time of visit.
Academics related information	Online search tool
Alumni related information	Mail service
Faculty/Staff related information	

2.2.4 THE GHANA TELECOM UNIVERSITY COLLEGE (GTUC) WEBSITE

GTUC is a relatively young university founded in 2005 by Ghana Telecom, the national telecommunications company. It was selected for this study because they offer quite a similar curriculum to what The Institute offers. The university was established to provide first degrees (bachelor's degrees) and graduate programs, particularly in Telecommunications Engineering and Information and Communications Technologies.

It also offers certificate programs, with courses that provide credit for the bachelor's degree, and a wide assortment of professional development seminars and workshops. Since its first class of 350 students matriculated in Fall 2006, GTUC has increased its enrolment to about 1,500 students, with 2,000 expected in August 2009. The college's website also gives general information concerning administration, academics, alumni news and events and general information. The online library

system was non-existent, at the time of visit, though the site indicated its development in the future.

Table 2.5: Summary of information and services offered by GTUC website

Information on website	Services offered on website
Administration related information	Indications of plans to build an online library system.
Academics related information	
Alumni related information	
News and events	

2.2.5 ASHESI UNIVERSITY COLLEGE (AUC) WEBSITE

AUC is also a private university college that opened in March of 2002. This university was selected for the study because of its popularity in Ghana. The university features a four-year bachelors program grounded in a liberal arts core curriculum, offering degrees in Business Administration, Management Information Systems and Computer Science. AUC website also offers the following information and services on the website:

Table 2.6: Summary of information and services offered by AUC website

Information on website	Services offered on website
Academics related information	Online library service
News and events	Mail service
Admission related information	Online search
Alumni related information	
Career development related information	

2.2.6 EVALUATION/CONCLUSION

This paragraph summarizes, as a way of analyzing the findings above, the features and services the above mentioned websites of the universities offer. These features are not comprehensive and only seek to find out what the general practice is in Ghana and then goes on to compare it with what is implemented in the WAS.

Table 2.7: The various services websites of selected universities offer.

Feature\University	UG	KNUST	UCC	GTUC	AUC	INSITUTE
General Information	A	A	A	A	A	A
Web mail	A	A	A	NA	A	NA
e-Learning	A	A	NA	NA	NA	NA
Online Library	NA	NA	NA	NA	A	NA
Online Results/Timetable	NA	A	NA	NA	NA	A
Discussion Forum	NA	A	NA	NA	NA	A
Online Search	A	A	A	NA	A	A

Legend:
A – Available/Present on the website NA – Indeterminate from the website

Table 2.6 above shows that the WAS is built to offer online information about The Institute, give the ability to students to check timetable and results online, offer a discussion forum and a search tool to allow users search the information database. It does not, however, offer online library, mail and e-Learning services as some of the tertiary institutions do.

2.3 EVALUATING CURRENT TRENDS IN WEB APPLICATION DEVELOPMENT

In the early days of the web, web sites consisted of static pages. Static content prevents the application interacting with the user. As this is limiting, web server manufacturers allowed external programs to run by implementing the Common Gateway Interface (CGI) mechanism. This allowed input from the user to be sent to an external program or script, processed and then the result rendered back to the user. CGI is the granddaddy of all the various web application frameworks, scripting languages and web services in common use today. CGI is becoming rare now, but the idea of a process executing dynamic information supplied by the user or a data store, and rendering the dynamic output back is now the mainstay of web

applications. This has been made possible by server-side scripting languages that can be embedded directly into web pages to generate web content dynamically. There are both open source (include PHP and Sun's JSP) and proprietary (includes Microsoft's ASP) implementations of these.

In just over a decade the web has gone from a technology with promise to major part of the world's infrastructure. It's been a fascinating time and many useful resources have been built in the process. From the initial objective of displaying static web pages mainly for sharing information among researchers, the World Wide Web has developed to the present-day platform that enable e-commerce, e-learning, social forums among others. Some of the developments have mainly evolved from the open source communities that leverage the power that currently available programming web tools offer, to come out with solutions that are either non-existent or are half-solutions to enterprise-wide problems and needs. PHP (an open-source server-side scripting language) and Apache (an open-source web server) has been tagged as being behind over 60% of Web Servers world wide (Kabir 2002).

A popular trend in web applications development in recent times is the use of frameworks that implement various design patterns. One such design pattern is the Model View Controller (MVC) design pattern. MVC was introduced as a graphical user interface organizational principle in Smalltalk platform in the mid-70s by Trygve Reenskaug (Fowler 2009a). It can be a good idea to segregate different kinds of code in an application, based on changes that one is likely to make. For programs with user interfaces (such as this system), it is generally considered a good idea to separate the user interface related code (the view logic or business logic) from the domain-related (model logic or application logic) code. Separating the two allows the programmer to make a change in one without having to touch the other.

Separating the user interface from the domain logic also allows one implementation to be swapped with another. Different views and controllers can be substituted to provide alternate user interfaces for the same model. For example, the same model data can be displayed as bar graph, a pie chart or a spreadsheet. A precise definition of MVC is almost impossible but Hayder (Hayder 2007) attempts to define MVC as the following:

"A Model is an object, which interacts with a database. All business logics are usually written inside the model. A Controller is a piece of code, which takes user inputs and based on that initializes models and other objects, and finally invokes all of them. Finally, the View is a component, which displays the result generated by controller with the help of model."

MVC is diagrammatically shown in the figure below:

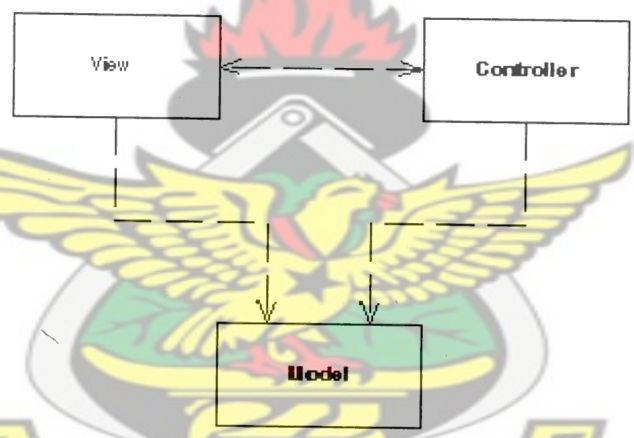


Figure 2.1: MVC splits user interface interaction into three distinct roles. Credit: Martin Fowler (April, 2009)

When using the MVC approach in web application development, the view component can then be managed by a third party program (for example Smarty Engine) for displaying purposes. The overall impact on application development is code manageability and maintainability and flexibility. In RAD (Rapid Application Development) with PHP, MVC frameworks play a vital role. These days several MVC frameworks have gained public interest and many of them are enterprise-ready (Hayder 2007). The Zend Framework is used to implement the MVC design pattern in the WAS.

JavaScript and AJAX (Asynchronous JavaScript and XML) have also enhanced user interactivity on the web. With the help of AJAX, for example, page information can be refreshed without necessarily refreshing the entire page. AJAX makes Web sites feel more interactive by updating only dynamic contents such as news feeds and stock quotes. As a result, X/HTML, CSS and scripting languages together with all the various web technologies available today could be brought together in a web application to offer robust and efficient web applications comparable to most stand-alone applications today.

2.4 STANDARDS IN WEB APPLICATION DEVELOPMENT

A technical standard is an established norm or requirement. It is usually a formal document that establishes uniform engineering or technical criteria, methods, processes and practices (Wikipedia 2009a). Standards are necessary to ensure cross-platform operations of systems and to promote systems interoperability. The World Wide Web Consortium (W3C) is the organization responsible for the development of standards for the World Wide Web and recommends some best practices for creating web pages.

Since 1994, W3C has produced more than ninety Web standards, called "W3C Recommendations." A W3C Recommendation is the equivalent of a Web standard, indicating that this W3C-developed specification is stable, contributes to web interoperability, and has been reviewed by the W3C Membership, who favour its adoption by the industry (W3C). The figure below shows the web architecture and also representing standards that are enforced by the W3C.

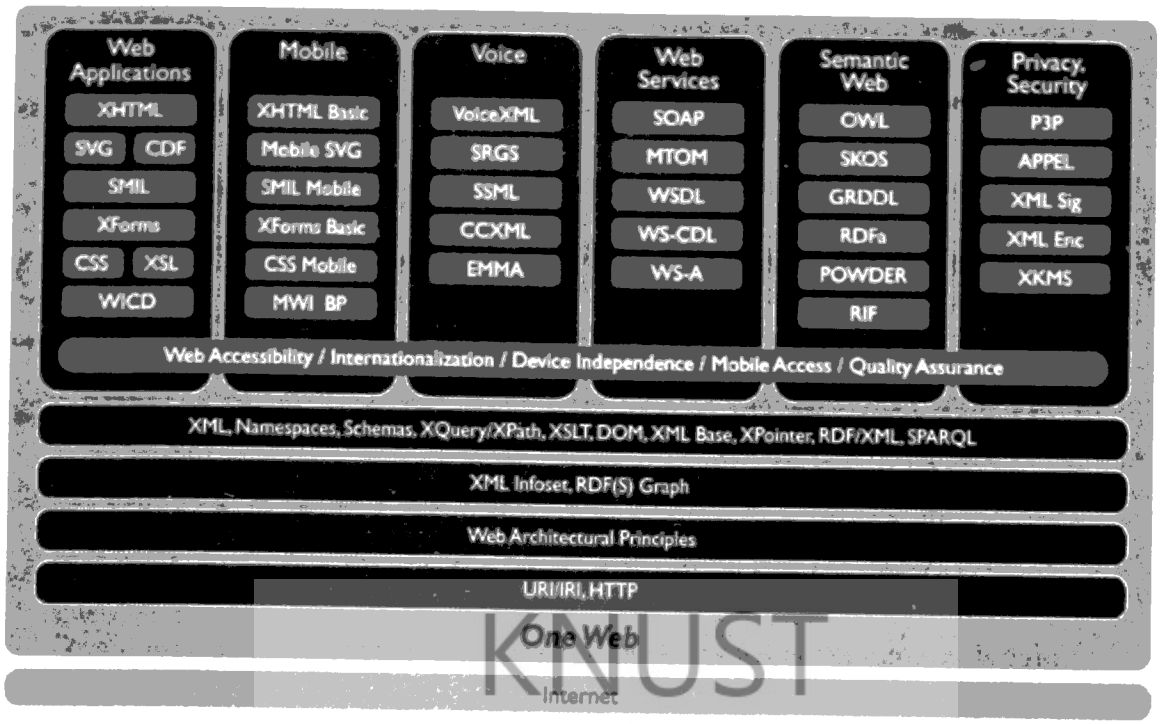


Figure 2.2: The web architecture as maintained by the W3C. Credit: W3C.org

The web architecture shows one view of web infrastructure, the focus of most work at W3C. The foundation of URIs, HTTP, XML, and RDF supports pursuits in five areas. Themes of accessibility, internationalization, device independence, mobile access and quality assurance pervade W3C technologies. The Web architecture is depicted as a series of layers, each building on the other.

2.4.1 WEB APPLICATION SECURITY

The Open Web Application Security Project (OWASP) Guide (Wiesmann, Stock et al. 2005) states that information security has relied upon the following pillars:

- Confidentiality – only allow access to data for which the user is permitted
- Integrity – ensure data is not tampered or altered by unauthorized users
- Availability – ensure systems and data are available to authorized users when they need it.

The application logic of the WAS takes into consideration the above pillars. This is necessary because the system makes use of user-submitted inputs. The security of the

system to some extent is ensured through the filtering of user inputs and correctly 'escaping' user-submitted data when it is returned to user's browsers.

Other system security threats could be cross-site scripting (XSS) and cross-site request forgery (CSRF). Zervaas (Zervaas 2008) defines XSS as when a malicious user masquerades as an authenticated user within an online application. He also defines CSRF as when a malicious user uses authorised identity information to perform tasks such as updates and delete tasks within a database. These security threats are mainly achieved through the use of JavaScript and session cookies.

A security measure that is implemented in the WAS is application logging. This is a log file that will be maintained automatically within the application to record significant events. The process makes use of the Zend_Log component of the Zend Framework. For example a log entry will be made when someone tries to use the system using incorrect identity credentials. The reason for keeping this file is that various user login activities can be monitored and helps the easy detection of malicious users.

Yet another possible security threat is what is called dictionary attacks. This is where a malicious user tries to access the system using brute force trial and error method to guess user names and passwords. The system deals with this type of threat by implementing an approach by Alshanetsky (Alshanetsky 2007).

2.4.2 APPLICATION MAINTAINABILITY AND EXTENSIBILITY

In software engineering, maintainability refers to the ease with which software can be modified to:

- Correct defects
- Meet new requirements
- Make future maintenance easier, or

- Cope with a changed environment;

These activities are known as software maintenance (Wikipedia 2009b). Looijen also explains that maintainability is the extent to which the information system components can be adapted as a result of maintenance (Looijen 1998).

Also in software engineering, extensibility is a system design principle where the implementation takes into consideration future growth. It is a systemic measure of the ability to extend a system and the level of effort required to implement the extension. Extensions can be through the addition of new functionality or through modification of existing functionality. The central theme is to provide for change while minimizing impact to existing system functions (Wikipedia 2009c).

In the quest to make the system maintainable and extensible, the following are considered:

- Using a template engine to separate application logic from display logic. As earlier explained in the chapter, the MVC pattern of web application development enables the separation of application logic from display logic. The smarty templating engine is employed to handle the view part of the MVC design pattern of the WAS.
- Using a Framework for building the system. The Zend Framework itself was designed with extensibility in mind (Zend 2008). For example using database abstraction to handle database server interaction (by way of its Zend_Db classes) enables the possibility of interacting with different database systems without changing much code. Other components as well can be extended through interfaces to meet the needs of programmers as and when desired.

- Making heavy use of Object-Oriented Programming (OOP) features in PHP scripting language to organize code. OOP helps build a culture of creating clean codes that can be modified by different programmers.

2.4.3 OTHER RELATED FACTORS CONSIDERED

2.4.3.1 SEARCH ENGINE OPTIMIZATION

Search Engine Optimization (SEO) is the process of improving the volume or quality of traffic to a web site from search engines via "natural" ("organic" or "algorithmic") search results (Wikipedia 2009d). As an Internet Marketing strategy, SEO considers how search engines work and what people search for. Typically the earlier a site appears in a search results list, the more visitors it will receive from the search engine.

The application has been developed and optimized for high search engine rankings. What that means specifically is that:

- The use of user-friendly URLs. For example, if there is a document called "About Us," in the website, a URL such as http://www.example.com/about-us would be user friendly, while a URL such as http://www.example.com/documents.php?id=1234 would not be so friendly.
- Correctly using HTML markup (such as headings, paragraphs, and tables).
- Correctly using HTTP status codes and content types (where relevant).

These features have been employed as a way of marketing The Institute through the web. It is envisaged that a wider audience would be reached through the internet as a way of advertising The Institute.

2.4.3.2 PHPDoc-STYLE COMMENTING

PHPDoc is an adaptation of Javadoc for the PHP programming language. It is a formal standard for commenting PHP code. It offers advantages to generic or random commenting styles such as allowing Integrated Development Environments (IDE) such as NetBeans to interpret variable types and other ambiguities in loosely typed language such as PHP which is used in building the WAS.

2.4.3.3 CROSS WEB BROWSER CONSIDERATIONS

Different implementations of web browsers can sometimes prove difficult in the build of web pages. Because there are many browsers and each one has its own implementation of the W3C standards of web page rendering, efforts are made to ensure that the WAS is cross-browser operable. This is made possible through the right use of CSS standards for displaying web pages.

2.5 TECHNOLOGIES/TOOLS EMPLOYED

These subsections below discuss the various tools or technologies employed in the development of the system. These technologies are of open source origin and have been widely used in the implementation of various projects including enterprise wide solutions. Some of the tools mentioned below do not form part of the final system that deployed on a production server since they were only used to help manage and develop the WAS. The versions of the tools used are mentioned accordingly where appropriate.

2.5.1 HTML/XHTML

HTML (Hypertext Markup Language) is the predominant markup language for web pages. It provides a means to describe the structure of text-based information in a document – by denoting certain text as links, headings, paragraphs, lists, etc. – and

to supplement that text with interactive forms, embedded images, and other objects (Wikipedia 2009e). HTML is written in the form of "tags" that are surrounded by angle brackets. HTML can also describe, to some degree, the appearance and semantics of a document, and can include embedded scripting language code (such as JavaScript) that can affect the behavior of Web browsers and other HTML processors. The Extensible Hypertext Markup Language, or XHTML, is a markup language that has the same depth of expression as HTML, but also conforms to the Extended Markup Language (XML syntax). HTML is the antecedent technology to XHTML. The most important change is the requirement that the document must be well-formed and that all elements must be explicitly closed as required in XML.

2.5.2 CSS

Cascading Style Sheets (CSS) is a style sheet language used to describe the presentation (that is, the look and formatting) of a document written in a markup language (Wikipedia 2009f). Its most common application is to style web pages written in HTML and XHTML, but the language can be applied to any kind of XML document. CSS is designed primarily to aid the separation of document content (written in HTML or a similar markup language) from document presentation, including elements such as the colours, fonts, and layout. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content (such as by allowing for tableless web design).

2.5.3 JAVASCRIPT/AJAX/JSON

JavaScript is a scripting language used to enable programmatic access to objects within other applications. It is primarily used in the form of client-side JavaScript for the development of dynamic websites. The primary use of JavaScript is to write functions that are embedded in or included from HTML pages and interact with the Document Object Model (DOM) of the page.

Asynchronous JavaScript and XML (AJAX), is a group of interrelated web development techniques used to create interactive web applications or Rich Internet Applications (RIAs) (Wikipedia 2009g). With AJAX, web applications can retrieve data from the server asynchronously in the background without interfering with the display and behaviour of the existing page. JavaScript Object Notation (JSON) is a data-exchange format in AJAX-enabled web application. It is essentially a JavaScript code and is used to serialize JavaScript arrays or objects in a simple format that can be exchanged between a client and a server.

2.5.4 PROTOTYPE, SCRIPTACULOUS, jQUERY LIBRARIES

Prototype is a JavaScript framework which provides an AJAX framework and other utilities (Wikipedia 2009h). The features of the framework range from programming shortcuts to major functions with XMLHttpRequest. XMLHttpRequest is JavaScript API that allows a background HTTP request to occur while a user is viewing a web page. It typically means that a page can be updated based on a response from the server without the user navigating to another page on the website. Prototype simplifies JavaScript development by providing the means to write a single code for different platforms (browsers). Prototype also provides library functions to support classes and class-based objects, something the JavaScript language does not have.

Scriptaculous, on the other hand, is a JavaScript library built on the Prototype JavaScript framework, providing dynamic visual effects and user interface elements via the Document Object Model (DOM). Prototype (version 1.6.0.1), Scriptaculous (version 1.8.1) are used together for client-side validation and other interactive techniques in the WAS. jQuery is also a lightweight JavaScript library that emphasizes interaction between JavaScript and HTML. Some of its features include DOM element select, DOM traversal and modification and AJAX functionality. jQuery (version 1.2.6) is used for enhancing slideshow display for galleries of the WAS.

2.5.5 THE ZEND FRAMEWORK

Zend Framework (ZF) is an open source, object-oriented web application framework implemented in PHP (version 5 and up). ZF seeks to promote web development best practices in the PHP community. ZF includes the following features:

- All components are fully object-oriented
- Extensible MVC implementation supporting layouts and PHP-based templates by default.
- Support for multiple database systems and vendors, including MySQL, Oracle, and Microsoft SQL Server, PostGreSQL Server among others.

The application uses ZF (version 1.7.8) components extensively. The framework enables the flexible management of database connections and transactions, user authorisations and management among others. Major components that are used in the WAS are mentioned and its implementation discussed accordingly.

2.5.6 PHP ENGINE

PHP (PHP: Hypertext Pre-processor) is an open source scripting language originally designed for producing dynamic web pages. It is a widely-used general-purpose scripting language that is especially suited for web development and can be embedded into HTML codes. It generally runs on a web server, taking PHP code as its input and dynamically creating web pages as output. Effectively, this means that PHP codes are hidden from clients but are 'seen' by PHP-enabled web servers. The application uses the PHP engine (version 5.2.5) for all server scripting purposes.

2.5.7 PEAR

PEAR (PHP Extension and Application Repository) is a repository of PHP software code. It seeks to provide a structured library of code, maintain a system for distributing code and for managing code packages and promote a standard coding style. The PEAR packages that were used in the WAS include:

- Text_CAPTCHA (version 0.3.1): this package is used for generating Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) images. CAPTCHA images are used to prevent spams on websites.
- Text_Password (version 1.1.1): this package is used for generating unique passwords for user of the system. Passwords are generated automatically by the WAS and sent by email to newly registered users of the system. Text_CAPCTHA also uses this package.
- Number_Roman (version 1.0.2): this package is used for generating roman numerals on the site.
- Image_Text(version 0.6.0beta): this is used together with Text_CAPTCHA to generate a CAPTCHA image.

The packages would also be required for its final deployment on a production server.

2.5.8 THE SMARTY TEMPLATING ENGINE

Smarty is an open source web template system used in implementing the view aspect of the MVC design pattern in the WAS. Smarty is primarily promoted for separation of concerns, which is a design strategy for certain kinds of applications such as web applications. Smarty is intended to simplify compartmentalization, allowing the presentation of a web page to change separately from the back-end (Wikipedia 2009i). Under successful application of this development strategy, designers are shielded from the back-end coding, and PHP programmers are shielded from the presentation coding.

Below is a pictorial representation of the using templates in web application design:

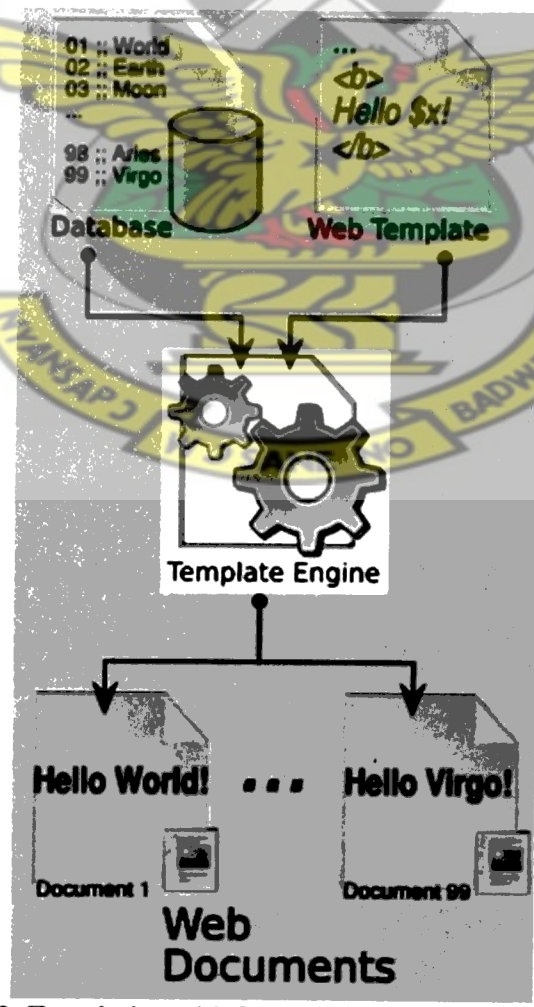


Figure 2.3: Templating with Smarty Engine. Credit: Wikipedia

In the figure above, contents from a database and presentation specifications in a web template are combined through the template engine to produce web documents. The Smarty engine (version 2.6.20) is used as the templating engine of the WAS. What that means is that anything that would be shown a user (that is, the HTML output) will be stored in a template file (with a .tpl extension). After a user request has been processed smarty will be used to output that template file.

2.5.9 MySQL RDBMS

MySQL is an open source relational database management system (RDBMS). It is popular for web applications as its popularity for use is closely tied to the popularity of PHP. Several high-traffic web sites including Flickr, Facebook and YouTube use MySQL for its data storage (Wikipedia 2009j). As the backend to the application, MySQL (version 5.0.45) is used to store all the various data that is generated from user requests. The Zend_Db_Adapter component of the Zend Framework is used to interface MySQL for its efficient data management.

2.5.10 PHPMyADMIN

PHPMyADMIN is an open source tool written in PHP intended to handle the administration of MySQL over the web. It can be used to do various tasks such as creating, modifying or deleting databases, tables, fields or rows; executing SQL statements; or managing users and permissions. This tool (version 3.1.3) is not part of the main application that is deployed to a server but is used to manage the database on the web server in the development environment.

2.5.11 BBPRESS FORUM SOFTWARE

BbPress is an open source forum software. It very flexible and allows a variety of settings that can be used to change its look and style. These include a wide variety

of plug-ins that can be added to add or change its functionality by the administrator. It is also supported by a very active online community of both users and developers. BBpress (version 0.9.0.2) is incorporated into the system to manage the forum for the Institute. It is also used to announce information to lecturers, students and staff of The Institute.

2.5.12 FCKEditor

FCKEditor is a lightweight and ready-made WYSIWYG editor that does not require installation of any programs on the client's computer. It is highly customizable with the following features

- It is easy to change the toolbar buttons available to users.
- It generates valid XHTML code (standards compliance).
- It is cross-browser compatible (except the Safari browser) and therefore replaced with a normal text input when necessary.

Version 2.6.4 of FCKeditor is used to enable the administrator to create and edit site updates on news and upcoming events.

2.6 CONCLUSION

Chapter two gave a brief introduction to the various websites that have been implemented by some tertiary institutions in Ghana. The reason was to sample out what already existing tertiary institutions in Ghana are doing through the web. The services that the websites offered were enlisted and then used to determine what the new website of The Institute could offer. The chapter also discussed the various technologies currently available in the development of web application. Also discussed were the various open source tools that were either developed or

incorporated into the system during implementation. Other aspects relating web application security and standards were also discussed.

3.0 MOTIVATION FOR WEB APPLICATION FOR THE INSTITUTE

3.1 INTRODUCTION

This chapter discusses the various processes that take place within The Institute. The processes are mainly those that relate to academic affairs and that is because they are the processes that the WAS seeks to improve. The chapter analyzes and evaluates these academic processes and then the ways that the WAS improves these processes.

3.2 ACADEMIC PROCESSES WITHIN THE INSTITUTE

Within The Institute various processes are involved to ensure a successful academic year. For those processes relating to examinations, the following are involved:

- A lecturer can teach at least a module. A module is scheduled to last for two weeks maximum. Some lecturers that teach in The Institute are from countries such as the Netherlands, South Africa, etc while others are from Ghana. There is an exam after every two successful modules.
- The lecturers of the various modules receive hand-written examination scripts via post in their countries of origin after their examination questions are administered by the academic officer.
- The scripts are marked and then sent back to The Institute also via post. Depending on factors such as when the marked scripts are sent to the post, it takes about a month for examination results to be received and then finally approved by the Academic Officer of The Institute.

- After results are approved by the Academic Officer, it is published on the students' notice board. Students who have doubts about a result are given the opportunity to appeal for a remark. Students who fail a module on the other hand are given the opportunity to rewrite the paper.

The diagrammatic representation of the processes is shown below in the figure:

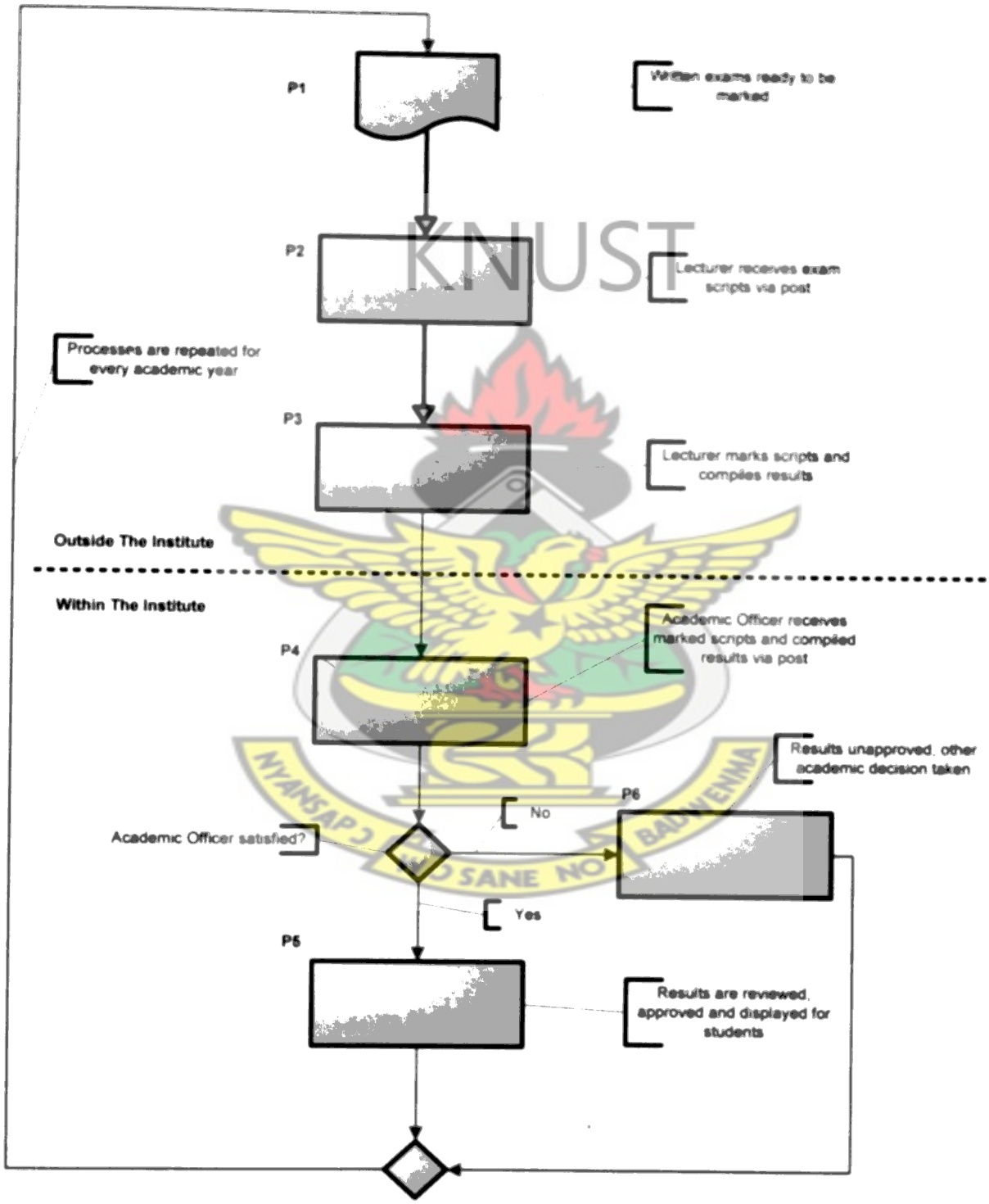


Figure 3.1: Results related processes within the Institute

The flow of processes in Figure 3.1 above describes what in involved between a lecturer and The Institute. The processes are numbered P1 to P6.

- **P1:** Process P1 represents examination scripts that have been written, submitted to the Academic Officer and then prepared and made ready to be sent via post to the lecturer responsible for the module. Other activities that go in the preparation stage involve backing up scripts (through scanning all the scripts) as a safety measure.
- **P2:** Process P2 represents all that is involved when a lecturer receives the examination scripts via post. This may involve verification and acknowledgement of receipt to the academic officer.
- **P3:** In process P3 the lecturer marks the scripts and compiles the results usually in Excel format. The marked scripts together with the results are packaged and sent back to the institute via again via post.
- **P4:** Process P4 is when the Academic officer receives the marked scripts and results from the lecturer via post. He also replies with an acknowledgement to the lecturer.
- **P5:** In process P5, the Academic Officer reviews the results, approves them and then displays them to students.
- **P6:** Process P6 represents the situation where results are not approved after review as a result of other prevailing matters. That then leads to other academic decisions to be taken

Another process that is examined is related to The Institute's quality assurance measures. As part of quality evaluation students are also allowed to assess the module at the completion of the module. The following are involved:

- The Academic Officer prepares the assessment questionnaire that the students use to assess every lecturer. This questionnaire is the same for all lecturers to ensure fairness.
- After every module, each student is given a questionnaire to fill. Although not mandatory for students, the practice is encouraged by The Institute. The completed questionnaires are submitted back to the Academic Officer.
- The Academic Officer takes the completed questionnaires and then processes/examines them for intelligence.

The diagrammatic representation of the processes is shown in Figure 6 below:

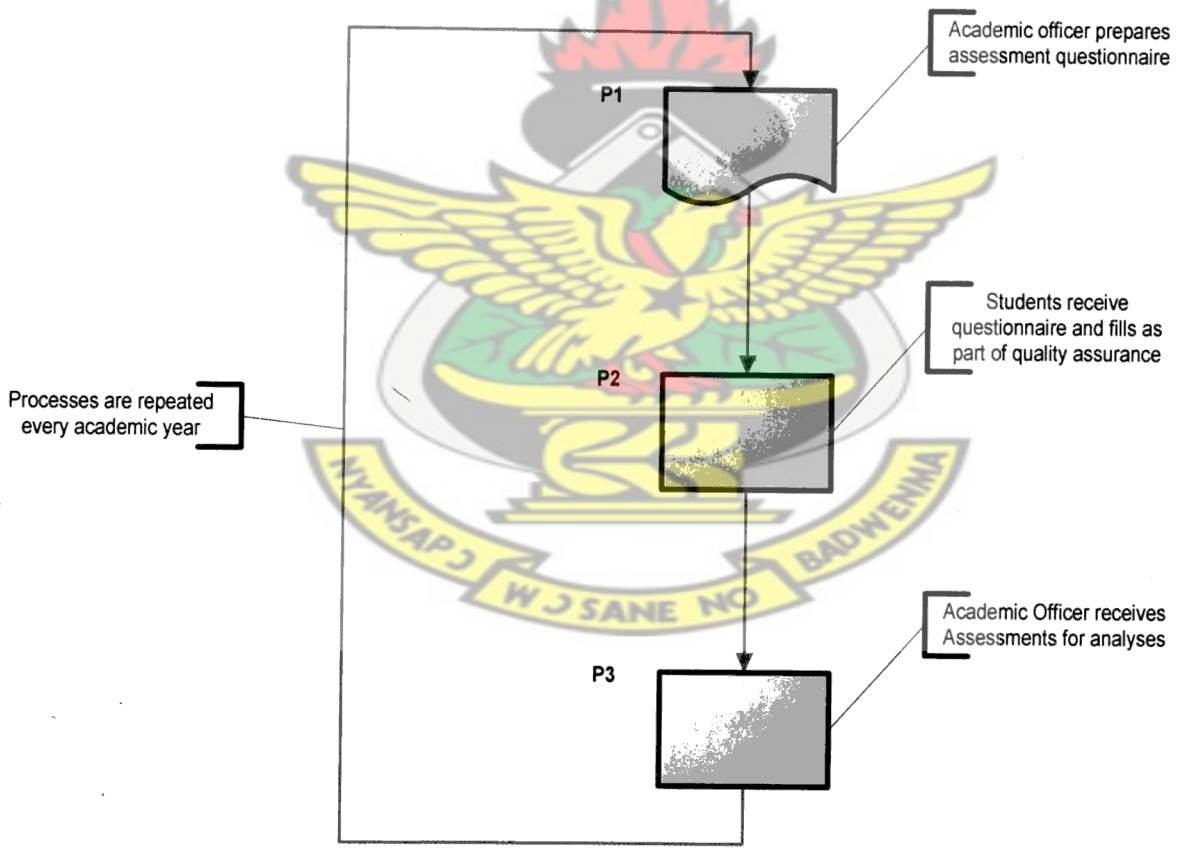


Figure 3.2: Quality Assurance related processes within The Institute

Figure 3.2 shows three processes named P1, P2 and P3. These processes are explained below:

- **P1:** Process P1 represents all the activities involved in the preparation of the questionnaire. The questionnaire takes into consideration a number of factors such as the way of lecture, the explanation of details and other questions relating to general lecturer-student relationship.
- **P2:** Process P2 represents the actual completion of questionnaires by students
- **P3:** Process P3 represents the receipt of questionnaires by the Academic Officer and any other academic decision that may be taken as a result.

3.3 ANALYSES OF PROCESSES

3.3.1 SHORTCOMINGS OF THE AS-IS PROCESS

A critical look of the current processes within The Institute for an academic year reveals a number of improvable processes. The causes of these inefficiencies are mainly due to the distance barrier that exists between the lecturers and/or students and The Institute. Other causes of the inefficiencies are as a result of manual processes which are error-prone and inefficient. The shortcomings identified are as follows:

- Students have to be physically present in the premises of The Institute to be able to assess a module.
- Examination scripts sent via post is risky and cannot be certified but can only be hoped to reach its destination at the time of being sent.
- Unmarked examination scripts and/or results pass through various stages and can be compromised. For example, when the scripts go missing.
- Compilation of results and assessment results could be very tedious if the number of students involved is high.

- General issues of concern to staff, students, and lecturers are addressed mainly requiring the physical presence of all stakeholders. For example when students have a peculiar concern that needs to be addressed, it becomes necessary that all the students meet with the authorities to resolve the issue.
- Building intelligence from assessment report with time could be virtually impossible as these completed questionnaires need to be examined individually as opposed to retrieving it from a database.

3.3.2 IMPROVING PROCESSES WITH THE WEB APPLICATION

The web application basically removes the distance barrier that presently exists between The Institute and the various stakeholders (students, staff, lecturers). The following can be achieved with the WAS:

- Lecturers can post online results to The Institute no matter where they would be in the world. All they would require is internet connection and access rights to the use the system.
- An online discussion forum would help all stakeholders to effectively communicate.
- Students would no longer be necessarily required to be within the premises of The Institute in order to access a resource such as results of examination.
- Academic results would be available to The Institute as soon as they are submitted by the Lecturer.
- The risk of losing marked/unmarked examination scripts through the post is reduced to a barest minimum.

- Students who complete a module could assess the lecturer without necessarily being in The Institute.
- Processing of required requests, such as a student's request for transcript is quickly done as all the information that would be needed would be in the database and can be easily retrieved.
- Overall students' data management is improved as it is centralised.
- The overall means for effective and efficient communication among stakeholders is improved.

3.4 CRITICAL SUCCESS FACTORS (CSFs)

The effective and efficient operation of the web application would depend on the following factors:

- Availability: According to Looijen, availability indicates the extent to which the information system is available for use when and where necessary (Looijen 1998). That means that the information system needs always be ready at any time to execute requests from users. This is worthy of mention because first of all the entire application would be hosted on the server of a third-party service provider. This is so mainly because of cost. As a result, the honouring or otherwise of Service Level Agreements (SLAs) between The Institute and service provider could influence effectiveness of the system.
- Reliability: Reliability is the probability that an information system performs specified functions accurately during a certain variable time period, whilst the operation of the information system takes place under certain environmental conditions (Looijen 1998). This CSF embodies the

continual functioning of the system, given the prevailing conditions, in performing its core functionality.

3.5 CONCLUSION

This chapter looked at the academic related information within The Institute, its deficiencies and how the WAS would help improve them. As noted earlier in the chapter, the web-based system for improving the academic processes within The Institute primarily removes the distance barrier that has always existed between all stakeholders. When that barrier has been removed, it is envisaged that all the processes will be streamlined and improved for proper and effective academic work. The discussion forum would also improve the communication among stakeholders. Students, staff and lecturers of The Institute can collaborate on online forums and discuss issues of importance to all. The system also builds up the image of The Institute in the public eye as a centre for teaching advanced information and communication technology.

4.0 DESIGN OF WEB APPLICATION SYSTEM

4.1 INTRODUCTION

This chapter discusses the design of the WAS. Actors of the system are identified and the user roles and privileges assigned to the different actors are discussed. Also explained is the system business logic which is mainly about database planning and engineering in accordance with 'business rules' of The Institute.

4.2 THE WEB APPLICATION SYSTEM

The WAS consists of a front-end and a backend. The front-end is the user interface that would be shown to users. The backend consists of the administration interfaces and data storage and management within in RDBMS. Access to resources is controlled so that only users with the right permissions are allowed. These permissions are set by the administrator and therefore required that users be authenticated and authorised in order to access a system resource.

Actors refer to the various users who will interact with the system. An actor may be a class of users, roles users can play, or other systems (Malan and Bredemeyer 2001).The users are grouped according the privileges given to them and subsequently the type of actions they are allowed to within the system. Actors for the WAS are classified as follows:

1. Unregistered users (also referred to as guests) – this group of users only has access to the front-end of the system. They are not allowed access to privileged information such as exam schedules and results.

2. Registered users of the system – this category of users are registered by the administrator of the system and so are given unique user identification tokens for the purposes of authentication and authorisation. They include:
    - students
    - lecturers

Registered users have controlled access to the backend of the system.

3. Administrator of the system

An administrator of the system is a super-user. They have overall control over the system and can even revoke the membership of other registered users.

Various access control mechanism are employed to ensure the security of the system using the Zend_ACL (Access Control List) component. Zend_ACL provides a lightweight and flexible access control list implementation for privileges management (Zend 2008). The diagrammatic representation is shown in the figure below:

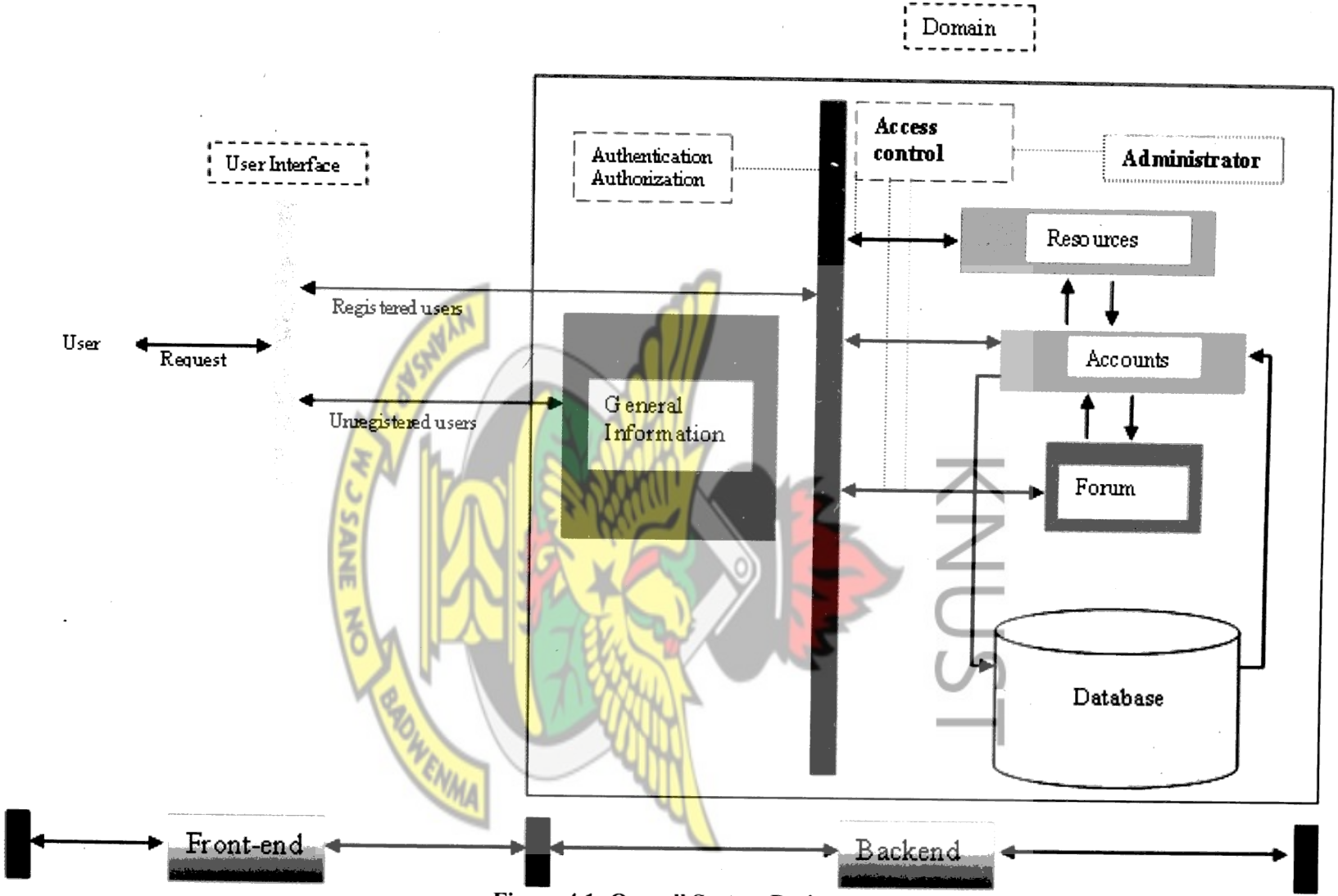


Figure 4.1: Overall System Design

4.3 FUNCTIONAL REQUIREMENTS

In software engineering, a functional requirement defines a function of a software system or its component. It can also be defined as a system requirement that describes an activity or process that the system must perform (Wikipedia 2009k).

Use cases have quickly become a widespread practice for capturing functional requirements (Malan and Bredemeyer 2001). This is especially true in the object-oriented community where they originated even though their applicability is not limited to object-oriented systems. A use case defines a goal-oriented set of interactions between external actors and the system under consideration. The functional requirements for the WAS gathers information about what the system offers for the various actors who interact with the system.

4.4 USE CASES AND UML DESIGN

An actor in Unified Modelling Language (UML) specifies a role played by a user or any other system that interacts with the subject. UML designs are used to elicit the roles that the different actors play to aid program design. The use cases are designed for the actors who will interact with the system.

4.4.1 GUESTS (UNREGISTERED USERS)

Unregistered users refer mostly to the general public. They only have access to the front-end of the system (see figure 4.1). This category of actors cannot log into the system as they are not allowed any privilege to do so. The front-end of the system offers general information about The Institute.

A use case diagram (see figure 4.2) has been used to show what a guest is allowed by the system. In the figure, a guest is only allowed information that The Institute uses for promotional purposes.

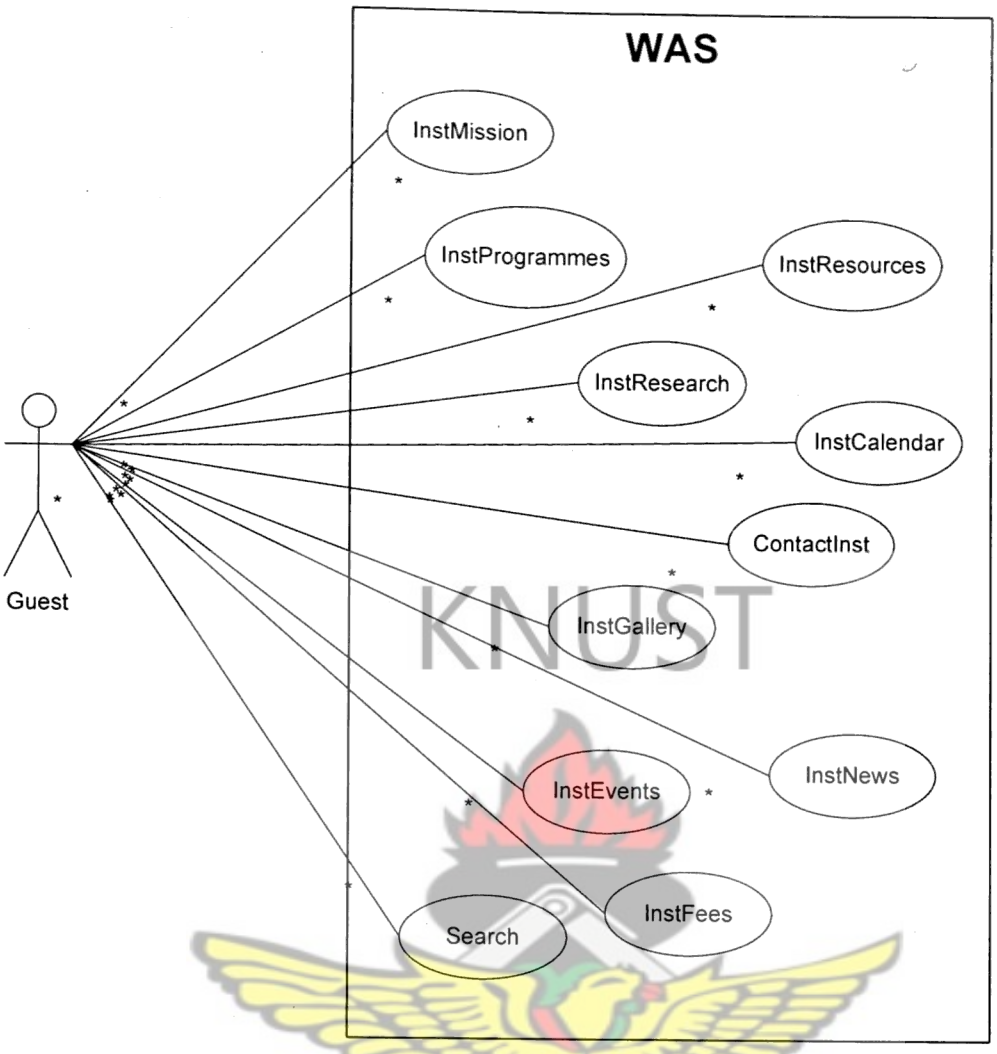


Figure 4.2: Use case diagram for the guest actor

This information bothers on

- The history of The Institute
- The vision and mission of The Institute
- The programmes offered by The Institute
- The academic and non-academic resources of The Institute
- News and events about The Institute
- A gallery for the general public
- A search tool for searching through news and events

4.4.2 STUDENTS

Students represent one group of actors that have controlled access to the system. Once a student actor provides their identity and credentials to the system, they are authenticated and granted access to the system. The use case diagram is shown in figure 4.3 below:

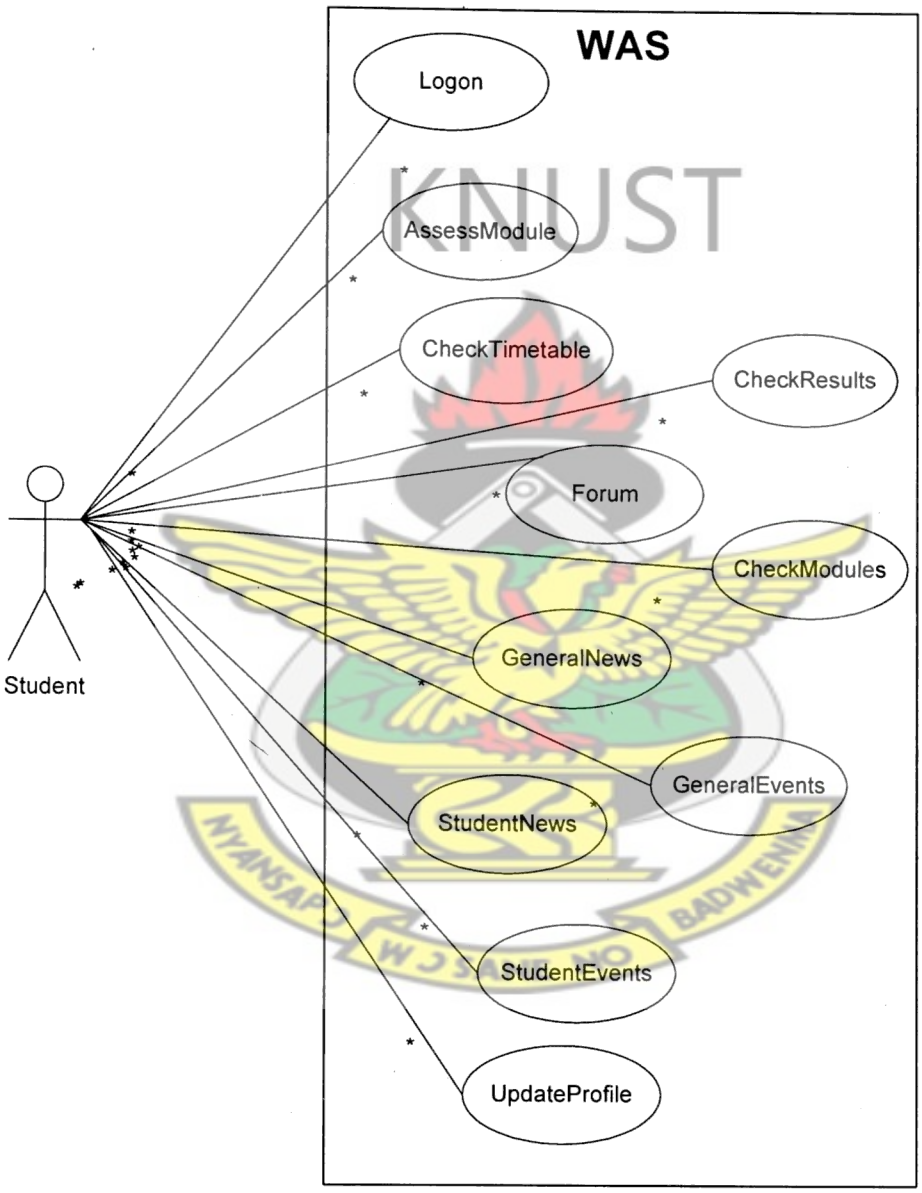


Figure 4.3: Use case diagram for the student actor

A student actor has a controlled access within the WAS. They can

- Log into the system and the discussion forum
- Have access general Institute information for guests
- Have access to information only for students (news and events).

- See and print the latest timetable for the academic year.
- See/Check their exam results.
- See the registered various academic modules.
- Assess a lecturer for a completed module of an academic year.
- Update their system account information.
- Can have access to the forum to contribute to discussions.

4.4.3 LECTURERS

Lecturers represent another group of actors with controlled access to the system. The use case diagram is shown in Figure 4.4 below:

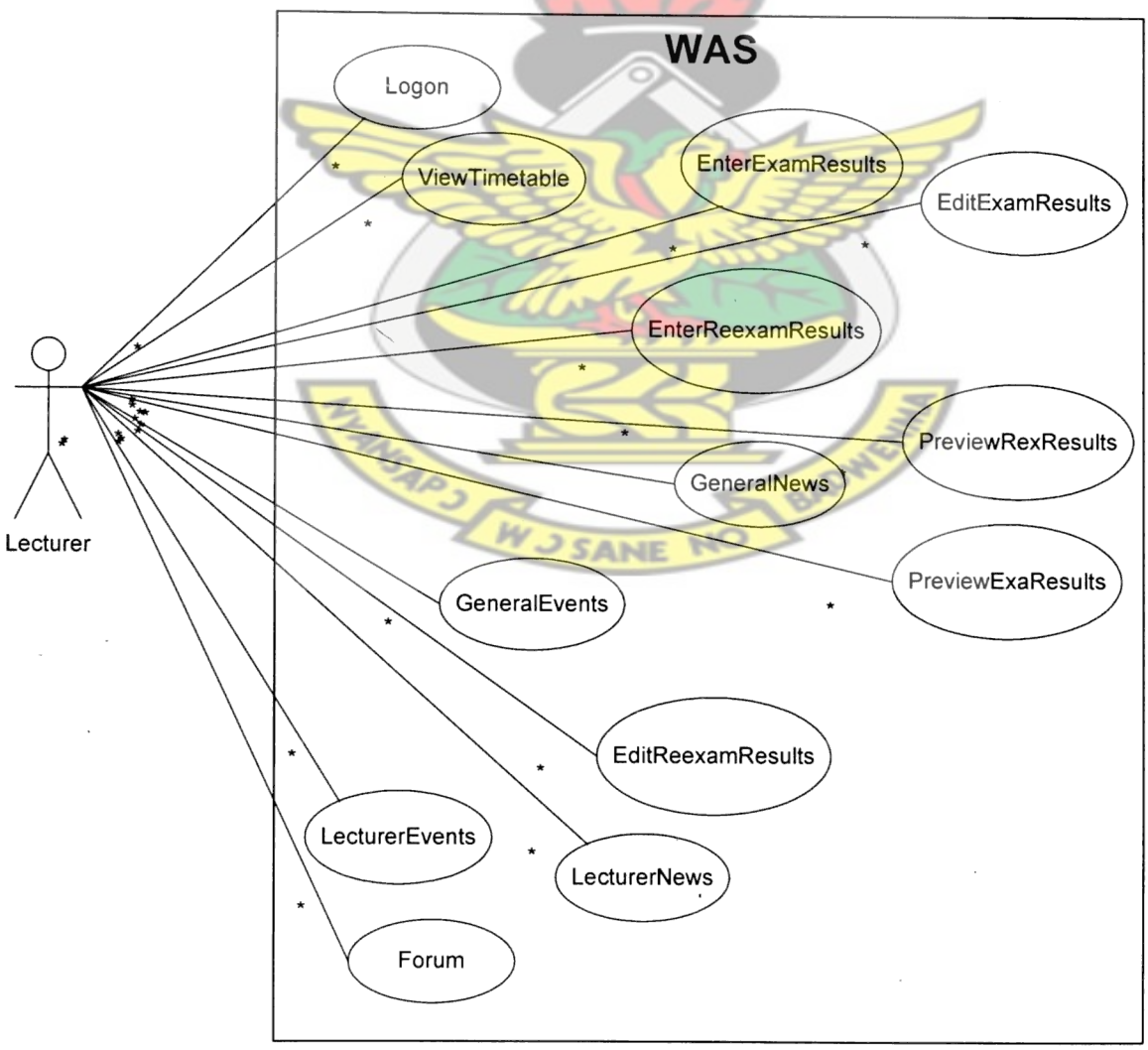


Figure 4.4: Use case diagram for the lecturer actor

A lecturer actor can

- Log into the system and the discussion forum
- Get access to general Institute information.
- Get access to only lecturer information (news and events).
- Enter results (both main examination and re-examination) for students.
- Edit results entered, if any.
- View registered modules for the academic year.
- Have a preview of the results submitted.
- Update their system account information.
- Can also have access to the forum to contribute to discussions.

4.4.4 ADMINISTRATORS

Administrator represents the group of actors with unrestricted access to the system. An administrator has complete control over the system which includes access to the database and some aspects of system design and implementation. The use case diagram for the administrator actor is shown below:

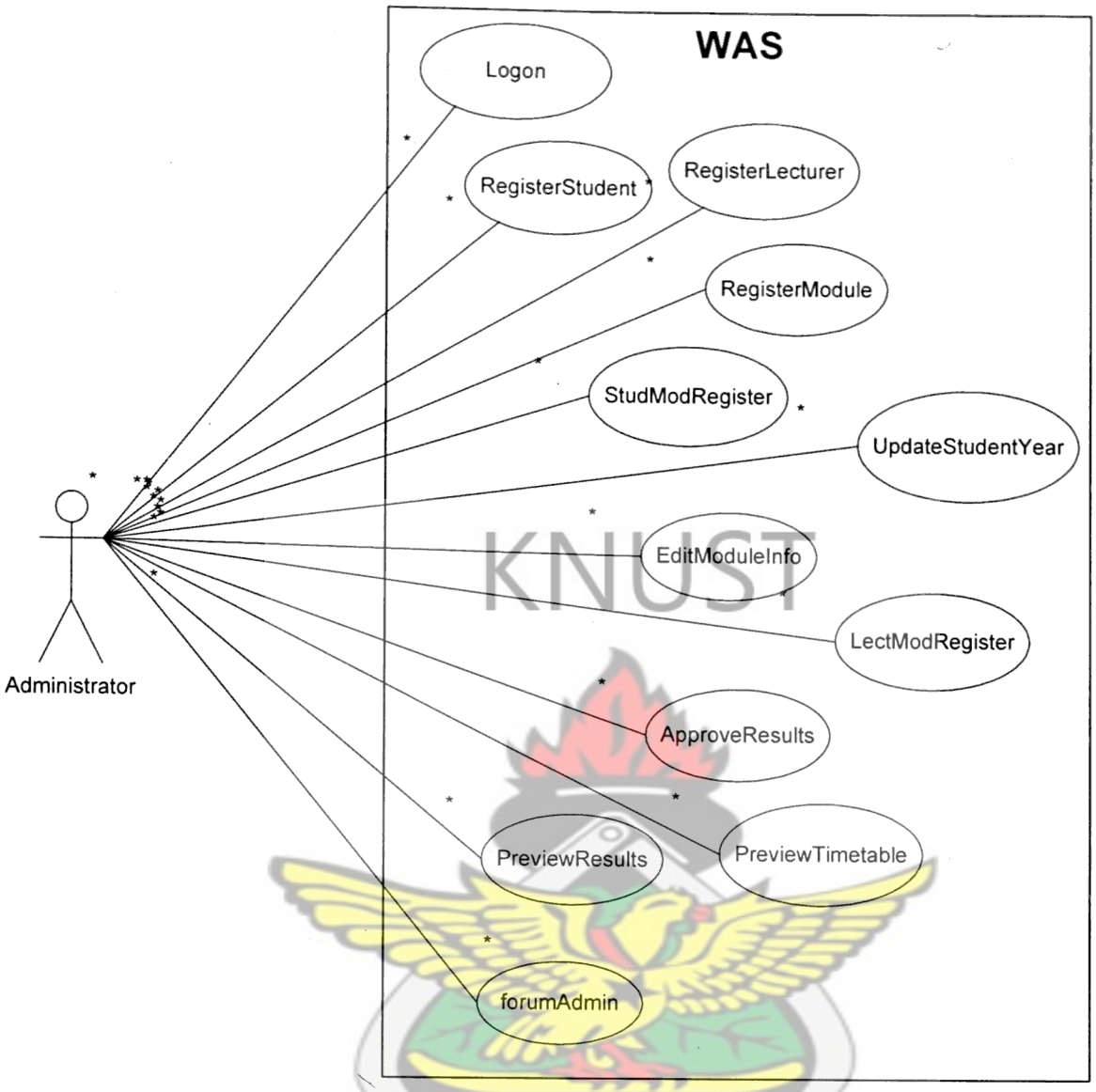


Figure 4.5: Use case diagram for an administrator actor

An administrator is allowed to do the following:

- Log into the system and the discussion forum
- Register modules to be taught for the semester.
- Create new accounts for both students and lecturers.
- Register a lecturer or a student for a module for an academic year.
- Preview/Edit the timetable for the academic year.
- Preview submitted results from lectures.
- Approve results submitted by a lecturer.
- Update a student year.

4.5 DESIGNING THE USER INTERFACES (WEB PAGES)

The user interface is an important aspect of any system. This is because how a system's user interface is designed contributes a lot to the system's ease-of-use feature. Building the user interface includes the following:

- Adding navigation and search engine optimization elements, such as the document title, page headings and breadcrumb trails.
- Creating a set of generic global styles that can easily be applied throughout all templates (such as forms and headings) using CSS.
- Allowing for viewing on devices other than a desktop computer (such as creating a print-only style sheet for "printer-friendly pages").
- Integrating the HTML and CSS into the existing Smarty templates, using Smarty templates to easily generate maintainable HTML.
- Creating an Ajax-form validator for user registration and other activities.

4.5.1 PAGE LAYOUT

The layout of every page in the WAS follows Matthew Levine's (Levine 2006) Holy Grail technique of using CSS to create a fluid three-column layout. It creates a middle column and two side columns according to developer-specified widths. The figure show a web page from the WAS with three columns namely: left column, right column and the middle column, the header and the footer.

The left and right columns are used for displaying auxiliary information such as news and events and the login form. The middle column is the content container and contains all forms and information for a user. The header is used for The Institute's logo (and can be expanded to include other information when the need comes up). The footer is used for copyright information and other notices.

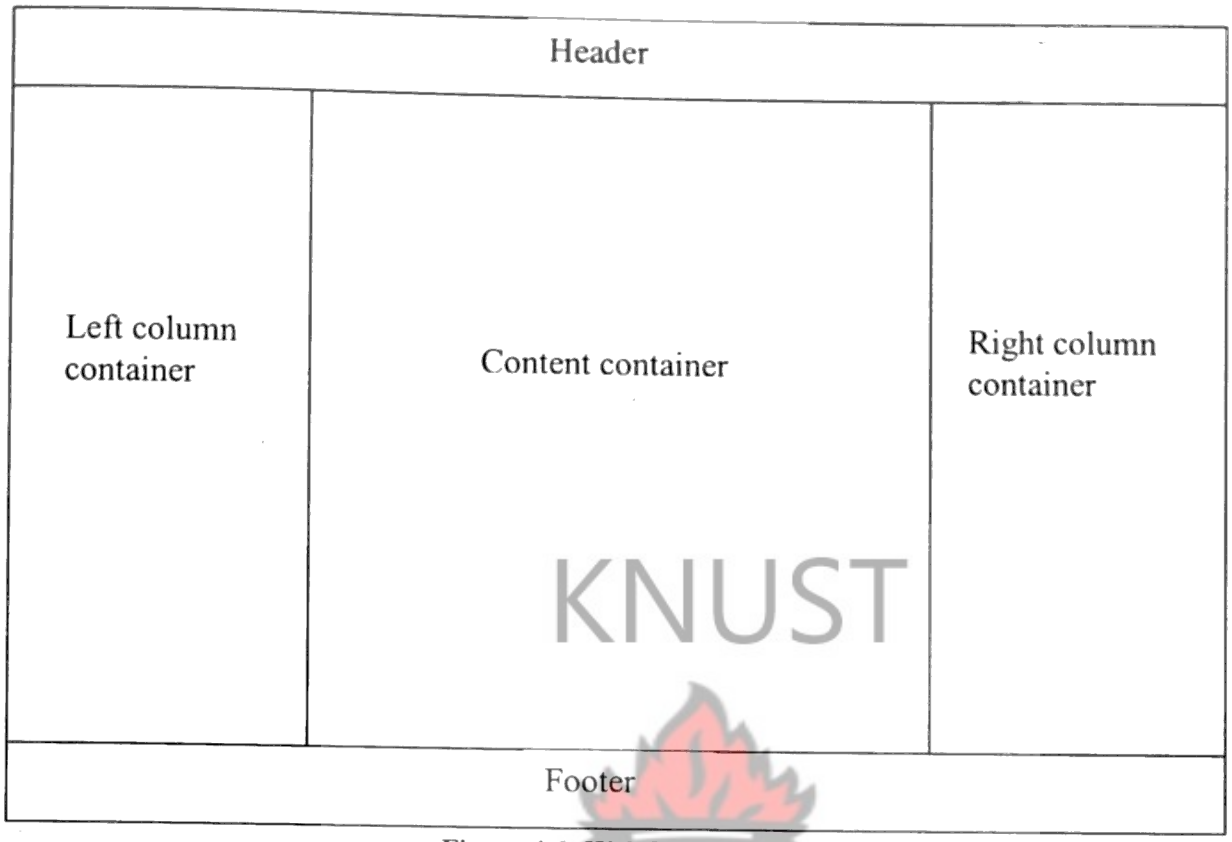


Figure 4.6: Web Page Layout

The key elements of this layout include:

- Three columns with a fluid middle column and fixed-size left and right columns
- No tables to set the columns
- A header area for a logo
- A tabbed navigation system with drop-down menus (just below the header)
- A breadcrumb trail and page title (at the top of the content container)
- A footer for copyright and other notices

4.5.2 PAGE TITLES AND BREADCRUMBS

Visually indicating to users where they are in the structure of a web site is important for the site's usability. This is done so that a user is able to identify where they are and how they got there without having to retrace their steps. To do this a title is assigned to every page in the application. A breadcrumb system is also built to aid navigation. A breadcrumb trail is a navigational tool that shows users the

hierarchy of pages from the home page to where they currently are. It essentially shows all of the parent sections the current page is in. A breadcrumb system might look like this:

Home > About US > Contact Us

In the example above, the current page would be Contact Us, while Home would be hyperlinked to the web site's home page and About Us would link to the appropriate page. In implementing the title and breadcrumb system, the following are done:

- The class (BreadCrumbs class) is used to hold each of the breadcrumb steps. The Breadcrumbs object is then assigned to the template, so it can be output in the page.
- A trail is built in each controller action with the steps that lead up to the action. The steps (and number of steps) will be different for each action, depending on its specific purpose.

4.5.3 PAGE NAVIGATION

The figure below shows the menu bar with a submenu for the home page.

Figure 4.7: Application navigation showing submenus

Page navigation is achieved with CSS. It consists of a horizontal main menu bar and vertical submenus for a menu bar where necessary. The menu bar and submenu are

all integrated into authentication and authorisation design such that they are changed to reflect the needs of the user.

4.5.4 CREATING A PRINT-ONLY SHEET

Creating a print-only sheet was implemented using the CSS print media type property. As a result, no secondary page needs to be loaded when one wants to print a user-friendly version of all the pages. The advantages are that

- The user doesn't need to navigate to another page in order to print content.
- There's not extra coding for the developer.
- The server does not have to serve an extra page, reducing server load and bandwidth.

What that really involves is hiding elements that do not need to be printed in which case means the navigation and left and right columns when a user wants to print from the browser. Typically, web browsers will strip out background colours and images when printing pages. This is dealt with by placing a printer-friendly image in the HTML and forces the browser to print the logo but is not displayed on the screen.

4.5.5 WEB FORMS DESIGN

Many forms are used in the system. All the forms meet the following requirements:

- Elements are clearly labelled
- Errors that occur on the form are highlighted to user.
- A submit button (and other buttons) are included.

Notifying errors to users on forms is done by way of a hidden div that is activated when the form contains errors. The Figure below shows a sample form from the

system showing arrangement of the form elements. The red labels below the elements indicate errors that have occurred on the form as a result of incorrect user inputs:

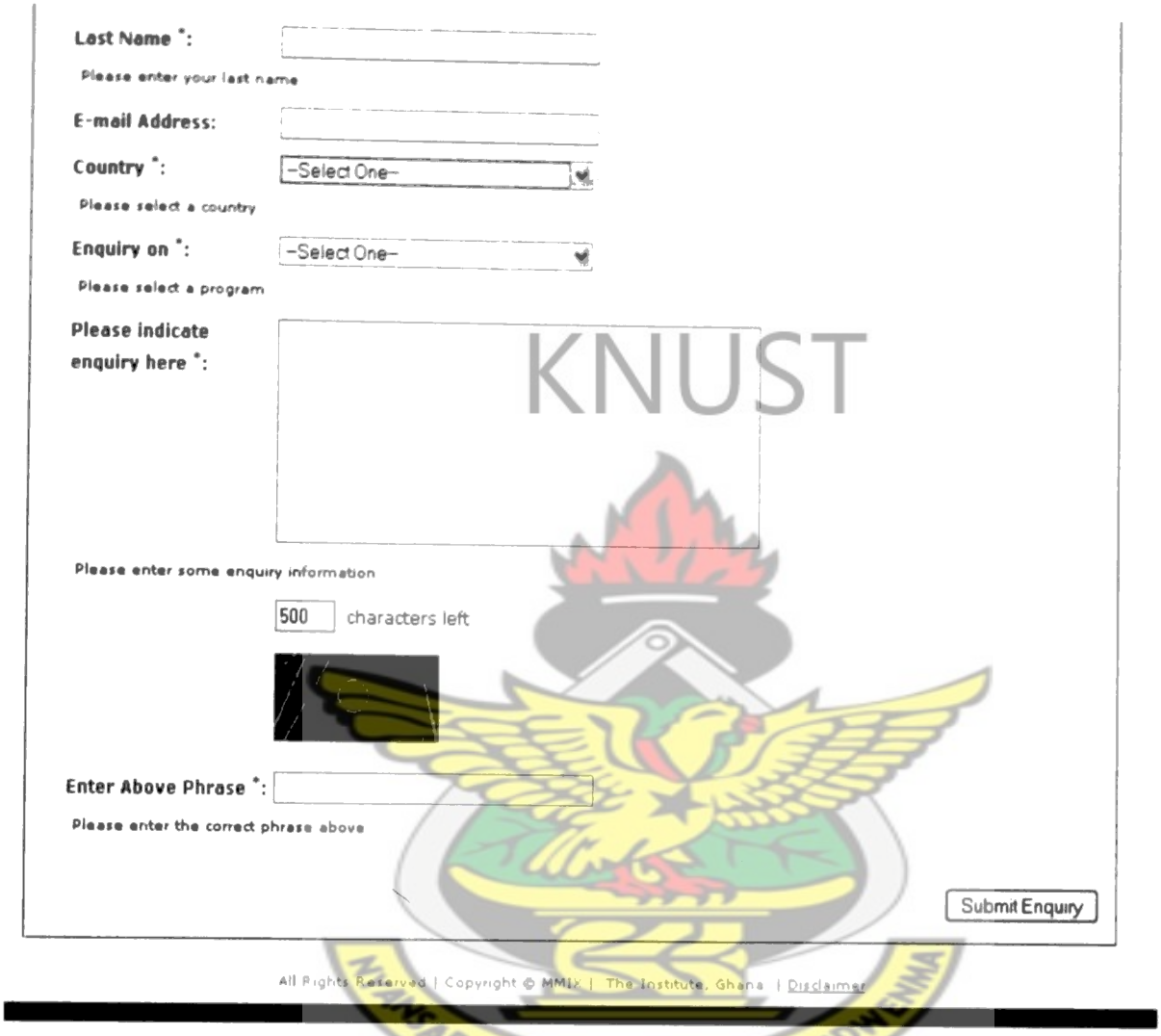


Figure 4.8: Picture of a form with errors in a browser

4.5.6 FORUM

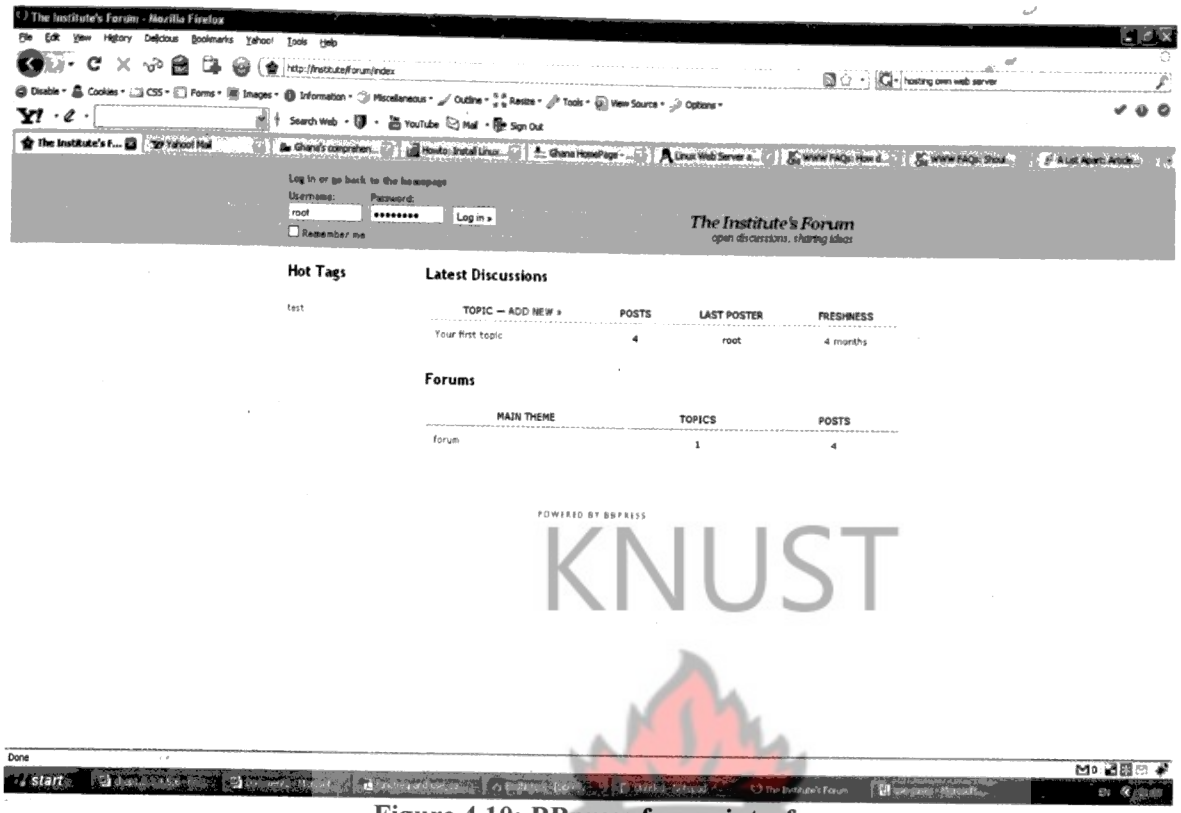


Figure 4.10: BBpress forum interface

The forum is implemented using the open source BBpress software as described in chapter two. It is incorporated in the WAS as-is with little interface customization. It is activated through an appropriate URL pointing to its location on disk within the WAS. Once installed, it allows logged in users to submit post(s) and reply to discussions. Administrators can do a lot more by way of monitoring and managing registered users, and changing various system settings including themes and managing plug-ins that enhance features of the system.

4.6 BUSINESS LOGIC AND FUNCTIONS

As discussed in chapter 2, business logic is a non-technical term generally used to describe the functional algorithms that handle information exchange between a database and a user interface. Business logic comprises:

- Business rules that express business policy (such as channels, location, logistics, prices, and products).
- Workflows that are the ordered tasks of passing documents or data from one participant (a person or a software system) to another.

The business logic behind WAS is as follows:

- Lecturers and Student actors must have accounts before they can access the system.
- Each Student actor must be registered for a module for an academic year. Otherwise a student is not allowed to assess a lecturer for a module.
- Each Lecturer actor must be registered for a module for an academic year otherwise a registered Lecturer actor will not be able to enter results for the model taught.
- The timetable is automatically generated based on the information registered for each module in the database.
- A result submitted by a lecturer must be approved by the administrator before it is made public to students.
- An approved result cannot be edited by the lecturer who submitted it.
- A lecturer who teaches more than one module cannot enter results for a second module if the results for the first recorded results have not been approved.

The above logic is built into the database design and is enforced through code and database design. Figure 4.10 shows the logical view of the database design implementing the business logic described the paragraph:

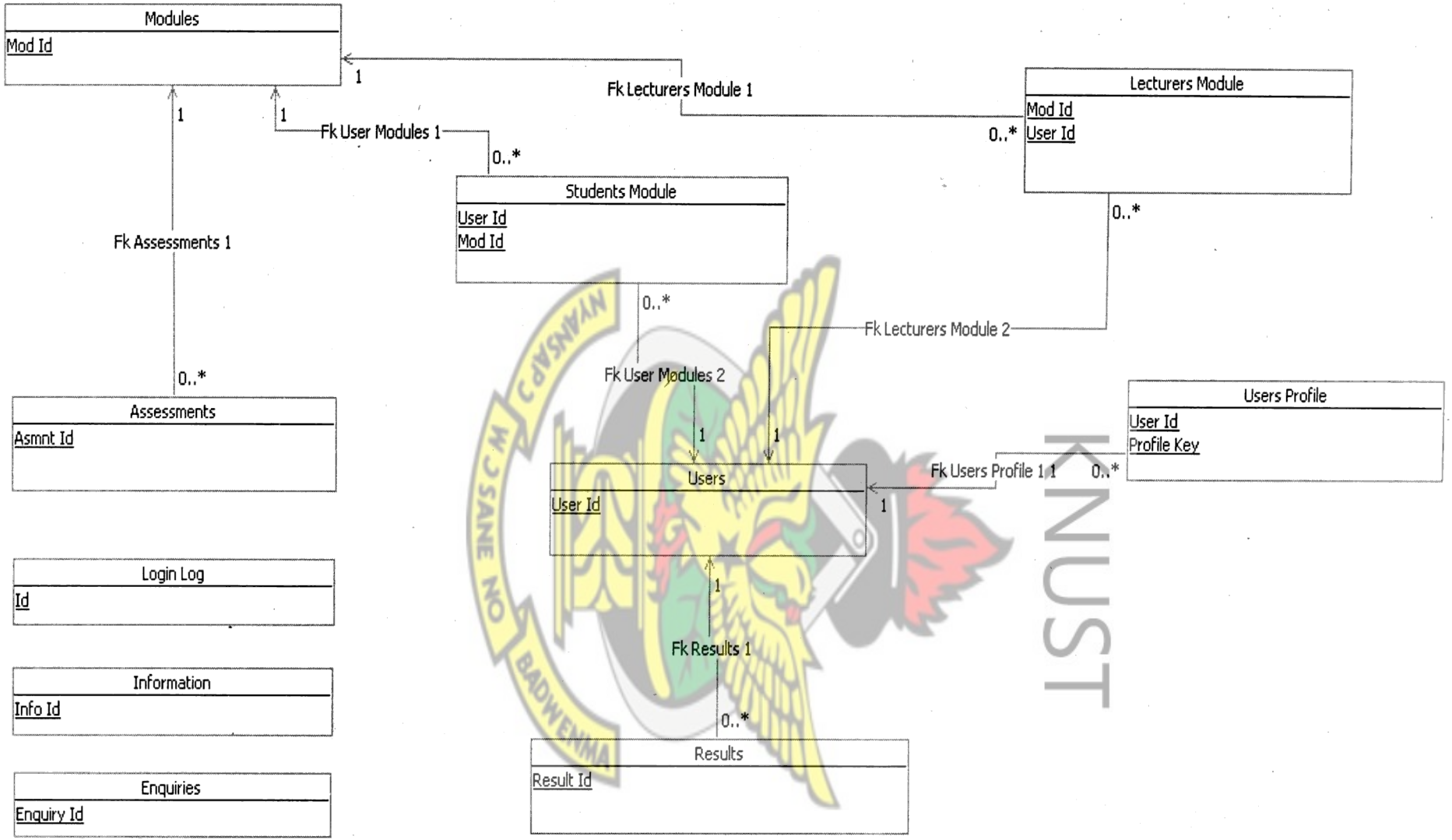


Figure 4.10: Logical view of database design

The database design above shows the following tables:

- Modules: This table stores all the information that pertain to the modules that are taught for each academic year. Information name such as the module name, the semester the module is taught, etc are stored here in the database.
- Users: This table stores the main user identity and credentials that are required for authentication and authorization. Other information such as the last user login time and the user type (which actor the user is) are also kept for aiding the monitoring user activities within the system.
- Users Profile: This table stores all other user information that are not for authorization or authentication. This design helps to separate personal user information (such the name, telephone number etc) and any other information that may be required later on from the system required user credentials and identity information.
- Results: This table keeps all the results that would be submitted by a lecturer to The Institute. Fields such as assignment marks, exam marks, the student who has the results etc are recorded here within the database.
- Assessments: This table takes the record of all the students' assessment data. All questions that are answered by the students by way of assessing a lecturer are kept in the table within the database.
- Enquiries: This table keeps all enquiries that are submitted by guests who would make enquires to The Institute online.
- Information: This table keeps all the various information (news and events) for all the actors that interact with the system. It is designed such that the different actors are shown only the information meant for them.

- Login Log: This table is implemented for system security and activity monitoring. It mainly is designed to help counteract and effectively deal with dictionary attacks from malicious users.
- Students Module: This table is only an association table.
- Lecturers Module: This table is also an association table.

4.7 THE SYSTEM ARCHITECTURE

4.7.1 PROCESS FLOW

By system architecture, all the various request processes as well as the objects, classes and interface for these processes are described. Process flow describes all the processes involved from when a user requests a page from a web browser to when and how the generated interface is shown to the user. The processes (1 to 19) are shown in Appendix 1. When a user submits a request through a browser, the browser passes the request on to the Controllers. Authentication and authorization checks are performed on the request. The subsequent results are then forwarded to Templates and Helpers which then generate the HTML and HTTP headers and all the necessary URLs needed to display the page in the browser.

All the processes are handled using the MVC design pattern described in chapter two. The View component, made up of Templates and Helpers, is responsible for the presentation logic. The Model component, also made up of Object Models, Resource Models and actual Database Resources, on the other hand is responsible for the business logic of the system (see Appendix 1).

4.7.2 PACKAGE DEPENDENCIES

Appendix 2 shows a diagram representing all the various dependencies that make up the system. It shows the various classes, objects and packages that interact

with one another. The classes DatabaseObject, Profile and FormProcessor (shaded in ash) are abstract classes. The packages Database, FormProcessor and Profile (shaded in light green) all contain various classes for managing database queries, processing web forms and table profile data respectively. All the classes in these packages extend one or more of the abstract classes. The Zend Framework package (shaded in dark green) is at the heart of the application and it is used by various classes and other packages as well. The Controller package contains classes that implement the controller aspect of the MVC design pattern. They interact with the Zend Framework through the CustomControllerAction class (shaded in yellow).

The CustomAclManager class is responsible for implementing the user authentication and authorisation. This implementation is based on the Zend Framework's Access Control List method of authorization and authentication. The is called automatically prior to user request dispatch. The Templater class (shaded in yellow) uses both templates and the Smarty Engine to render web pages. It also tied to the Zend Framework. Templates render web pages using standard HTML/CSS codes to ensure compliance, maintainability and reusability. FCKEditor class (shaded in yellow) uses the FCKEditor package to create a flexible text editor within a web page. The forum package depends on the controller to create and manage a discussion forum for the WAS. EmailLogger and Breadcrumbs classes (shaded in yellow) are used for creating objects to manage system emails and create breadcrumbs for web pages respectively (see Appendix 2).

4.8 CONCLUSION

This chapter looked at the design of the WAS. The chapter captured the functional requirements of the system. Use cases were used to identify the various actors and their roles and functions within the system. The chapter also discussed the

implementation logic aspect of the WAS. The various rules that exist with The Institute concerning module teaching and learning, assessment of lecturers were also discussed. Those rules and how they were implemented within the database design were also looked at. The system architecture looked at process flows consisting mainly of the processes involved when a user requests a page from the system and how the system responds. The process flows indicated the various roles the different aspects of the MVC design pattern played. Also discussed were the various packages and classes within the system as well as the dependencies between these classes and packages.

5.0 IMPLEMENTATION OF THE WEB APPLICATION SYSTEM

5.1 INTRODUCTION

Chapter 5 discusses the actual implementation of the WAS. Here, how files are arranged on disk is discussed. The chapter also discusses the implementation of the Model View Controller design pattern using the Zend Framework. How that design pattern is used to implement the various functional requirements gathered is also discussed.

5.2 APPLICATION FILESYSTEM LAYOUT

The structure of the file system on disk is necessitated by the use of ZF which recommends a particular file hierarchy in order to function at all. These folders could be named as desired but must be arranged to ease auto loading of required classes and libraries. The various folders and their contents are described in the following sub paragraphs.

5.2.1 WEB ROOT DIRECTORY

This directory is where the web server looks for files when a user requests a page on the site. For security reasons, most of the files used by the system exist outside of this directory (such as PHP classes and web site templates), which prevents users from directly accessing these files. In the development environment the web root is called *htdocs* even though it could be changed to suit the desired needs on a production server.

5.2.2 DATA STORAGE DIRECTORY

This directory is purposely for storing application data (that is, data in addition to that in the database). This is where log files (both from the web server, and those created by the system) and temporary data are kept.

5.2.3 PHP CLASSES DIRECTORY

This directory contains all PHP functions and libraries. It is named the *include* directory. Any third-party scripts used is also stored in this directory in addition to self-developed codes. Application controllers (scripts that define the different actions users can perform on the web site) are stored in a directory called Controllers in the include directory.

5.2.4 TEMPLATES DIRECTORY

This directory holds the web site templates. Templates refer to codes (mostly HTML markup) that are used to define the display logic of the web application. These are placed outside the root directory because they shouldn't be directly accessible. The figure below shows the full directory structure of the system.

```
/(document root)
|       -       /data
|       |       -       /logs
|       |       -       /tmp
|       -       /htdocs
|       -       /include
|       |       -       /Controller
|       -       /templates
```

Figure 5.1: The full view of the directory structure

5.3 IMPLEMENTING MVC WITH ZF

The Model-View-Controller (MVC) design pattern was described in chapter 2. The WAS uses the Zend_Controller class (explained below) of the Zend Framework to implement this design pattern. The paragraphs below describe the steps and classes used the in the implementation of the pattern.

5.3.1 SYSTEM BOOTSTRAPPING

Zend Framework's controller, the Zend_Controller, is designed to support websites with user-friendly URLs. To achieve this, all requests need to go through a single *index.php* file (bootstrap file). This is known as the Front Controller design pattern (Fowler 2009b). This design pattern provides a central point for all pages of the application and ensures that the environment is set up correctly for running the application. This is achieved using a file (*.htaccess file*). The use of this file is accomplished with a web server module (namely mod_rewrite in Apache web server).

The code, or the rewrite rule, for enabling the front controller design pattern in the application is shown below:

```
RewriteEngine on
Rewrite %{ REQUEST_FILENAME}! - f
RewriteRule .* index.php
```

The code snippet above is interpreted as for any URL that doesn't map to a file that exists on disk, use *index.php* file instead.

5.3.2 THE ZEND_CONTROLLER

Zend_Controller is the heart of Zend Framework's MVC system (Zend). The Zend_Controller_Front implements the Front Controller system design pattern discussed in chapter two. As a result, all requests are intercepted by the front

controller and dispatched to individual Action Controllers based on the URL requested. The bootstrap file loads and instantiates the Zend_Controller_Front class and then calls the *dispatch()* method, which will call the necessary code to handle the request. The Zend_Controller_Front is a singleton class, meaning that only one instance of the class may exist hence it is initialized in the bootstrap file and then called throughout the system. With ZF, the controller is a class that must be called *{Controller name}*Controller, where *{Controller name}* specifies the name of the controller file. This class must necessarily be named *{Controller name}*Controller.php within the Controllers directory for auto loading of the class by the ZF.

For example, if there was a controller called foo, that controller would be named *Foo*Controller and will be located in the Controllers directory. When Zend_Controller routes a user request, it automatically looks in the controller directory for a file called *Name*Controller.php where *Name* corresponds to the *{Controller name}*. To then access this controller in the application, the URL http://sitename/foo. To view a specific action within the controller, say an action called bar, it would then be accessed at http://sitename/foo/bar. The action is a method within the controller class which would be *(Gfbares)*Action() inside of *Foo*Contoller class.

5.3.3 THE ZEND_VIEW

Zend Framework's view component is called Zend_View. The view component allows the separation of the code that displays the page from the code in the actions functions. It is used to handle to view aspect of the MVC design pattern. Smarty was purposely design for this aspect of the MVC design pattern and so would be used instead.

5.3.3.1 USING SMARTY AS A METALANGUAGE

While using PHP code directly for templates (as used in Zend_View) is a perfectly viable solution, it can be very useful using a meta-language for templates instead (Zervaas). A direct advantage could be that the code is shorter and more easily readable. For example, using *{$foo}* to output the PHP variable *$foo* provides less clutter code than *<?php* echo *$foo ?>*. Even though the ZF's Zend_View component provides a templating solution, it uses native PHP files. Zend_View works similarly to Smarty, except that the templates it uses are written in PHP native codes. A class (Templater.php) is used to integrate Smarty in the code base.

5.3.4 THE ZEND_DB

The Zend_Db class is used for creating connections to a database backend. It is also used in the implementation of the model component of the MVC design pattern. The model part deals with the application's core purpose (the so-called 'business rules'). The Zend_Db class supports a number of database adapters including MySQL, Oracle, PostgreSQL and MSSQL RDBMS. This makes it possible to use a single code to interface all the different database adapters. The WAS uses this component of ZF for all its database calls and interactions.

5.3.4.1 THE DATABASEOBJECT CLASS

DatabaseObject class is a class that is heavily used by the WAS to manage database records. It was developed by Quentin Zervaas (Zervaas 2008) and acts as extra layer on top Zend_Db component. It makes simple the reading, writing and deleting of rows from the database. Essentially, each major table in the WAS extends the DatabaseObject class. So to manage records from the users table, the class DatabaseObject_User is created. Once this class is instantiated, its load() method can

fetch a record from the database, the save() method can be used to insert or update a record into the database and delete() simply removes a record from the database.

5.3.4.2 THE PROFILE CLASS

The abstract Profile class is used to manage profile data. Every profile data extends this class. It is designed such that extra information can be added to the profile data when future need require. A table such as the User_Profile table is used to store all the profile data of a user such their first name, last name, email among other. The class is also a third-party class adopted from Quentin Zervaas (Zervaas 2008).

5.4 IMPLEMENTING USER AUTHENTICATION AND AUTHORISATION

The different actors have different access privileges and functions they can perform with the system. For example, Unregistered users do not have any access right and so cannot log into the system. An attempt is made to distinguish between authentication and authorisation. Zervaas defines *authentication* as follows:

- Authentication is the process to determine whether a user is in fact who they claim to be. That is accomplished using a unique username (their identity) and a password (their credentials) (Zervaas 2008).

He also defines authorisation as that:

- Authorisation is the process to determine whether a user is allowed to access a particular resource. That happens given that the system knows who they are from the authentication process. Authorisation also determines what an unauthenticated user is allowed to do (Zervaas 2008).

Authentication in the WAS is implemented using the Zend_Auth component whereas authorization is done using the Zend_ACL of the ZF.

5.5 IMPLEMENTING FORM VALIDATION

Form data validation is an important aspect of web application system programming. Data validation ensures that a program operates on clean, correct and useful data. Various methods are designed that check for correctness, meaningfulness and security of data that are input to the system. Two types of validation are implemented with the WAS. They include:

- Server-side Validation
- Client-side Validation

5.5.1 SERVER-SIDE VALIDATION

All server-side form validations are done using the FormProcessor abstract class. All other classes that process forms (with name class FormProcessor_{name}) extend the FormProcessor abstract class which is in the include directory. The FormProcessor class holds the form values that are assigned to it as well as error messages that can be displayed to the user. These validations include methods and codes to

- 'Cleaning' submitted data from malicious codes and tags. This is done by stripping HTML tags from the string and trimming white space from the start and end of the string.
- Ensure that required fields are not empty and if empty notify the user.
- Make sure input data are submitted in the correct format.
- Checking for non-duplication in database based on submitted values

5.5.2 CLIENT-SIDE VALIDATION

Client-side validation is implemented using JavaScript and Ajax. It is really just a proxy to the FormProcessor classes mentioned above. Adding client-side validation improves usability since the user will receive feedback about any invalid form values more quickly. Prototype and Scriptaculous JavaScript libraries are used in implementing Ajax.

Specifically, each of the form fields are checked for valid values when the user clicks the submit button to send values to the server. If everything appears correct, the form is allowed to be sent to the server. The validation is such that even if a user does not have JavaScript enabled in the browser, they cannot circumvent any of the data checks. The implementation involves the following:

- Creating the options either for validating a form only or both validating and sending requests to server.
- Sending a JSON response containing any errors that occurs if the action is requested via Ajax.
- Creating a JavaScript class to trigger the form validation, as well as submitting the form once all values have been verified.

All these validations are done without post-backs. This means the validation are done without the page reloading after validation (JSON functionality). The client-side still uses the server for validation even though it is quicker than doing normal post-back since the page does not need to be reloaded.

5.6 IMPLEMENTING FUNCTIONAL REQUIREMENTS

This paragraph looks at a summary of the various pages for the different actors in the system. The system is designed such that authenticated users have different

perspectives so that they see only what they are allowed to see within the system. The paragraphs give screen shots for some of the various interfaces.

5.6.1 USER ACCOUNTS AND MANAGEMENT

All registered users have an account within the system. This account (their username and passwords) are unique to each user. The password is generated by the system and sent to an email that the user is required to provide. Once the user is logged on the system, they can decide to change their account information. A facility is also provided to allow users who forget their passwords to be given a new one.

5.6.2 THE APPLICATION HOME PAGE

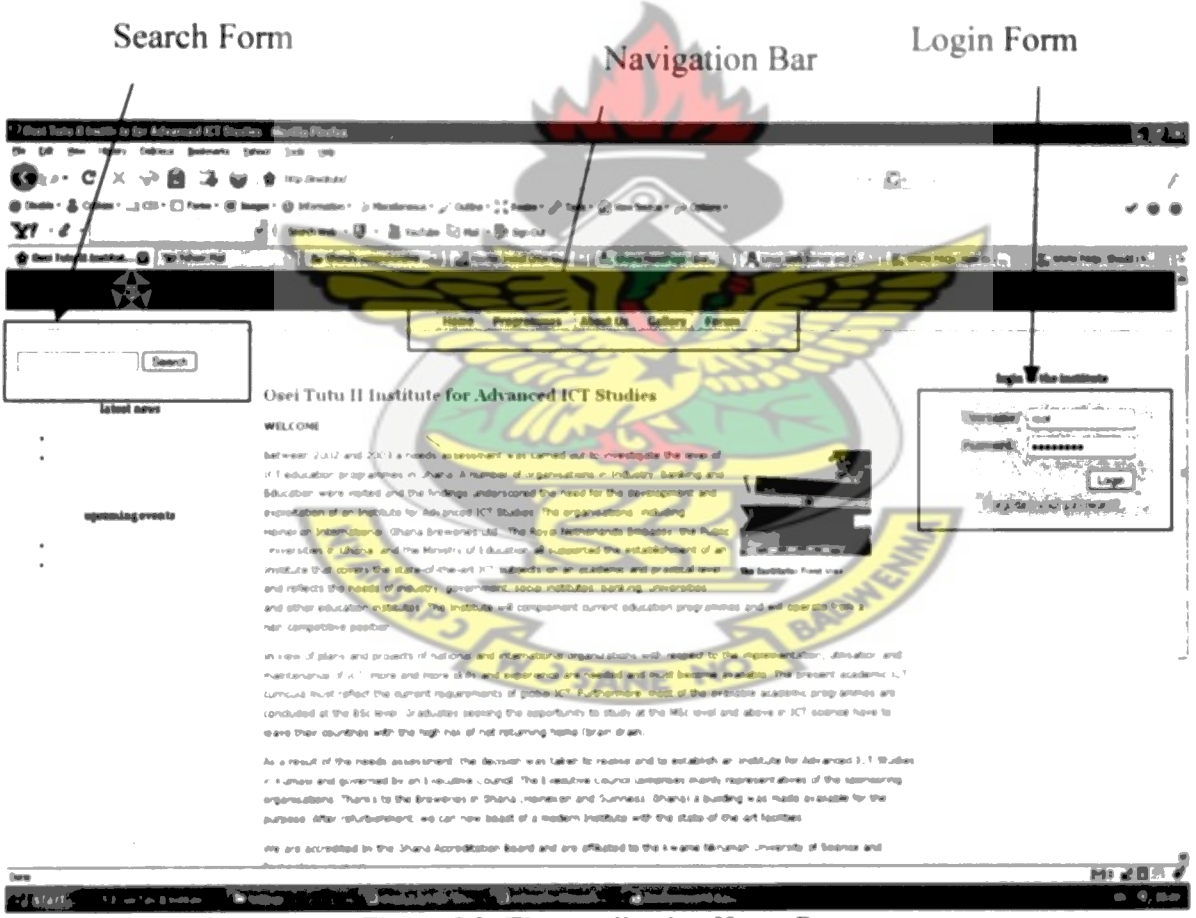


Figure 5.2: The Application Home Page

The application home page is the page that is loaded as the first web page when a user navigates to the site with a browser. All other actors that have login access also have specific homepages too. This page however, is designed for guests but is

also seen by students and lecturers. Because it is designed for guests it has all the information relating the activities of The Institute. The pages seen by actors are:

- Pages on information about The Institute (history, mission, vision, resources).
- Pages on the information about The Institute's programmes (MSc, PhD, research, fees, calendar).
- Pages on news and events within The Institute.
- Pages on The Institute's gallery.
- An enquiry form for sending quick enquiries to The Institute.
- A login form for authenticating registered users.
- A search tool for searching the site for information and events.

5.6.3 STUDENTS' PAGE(S)

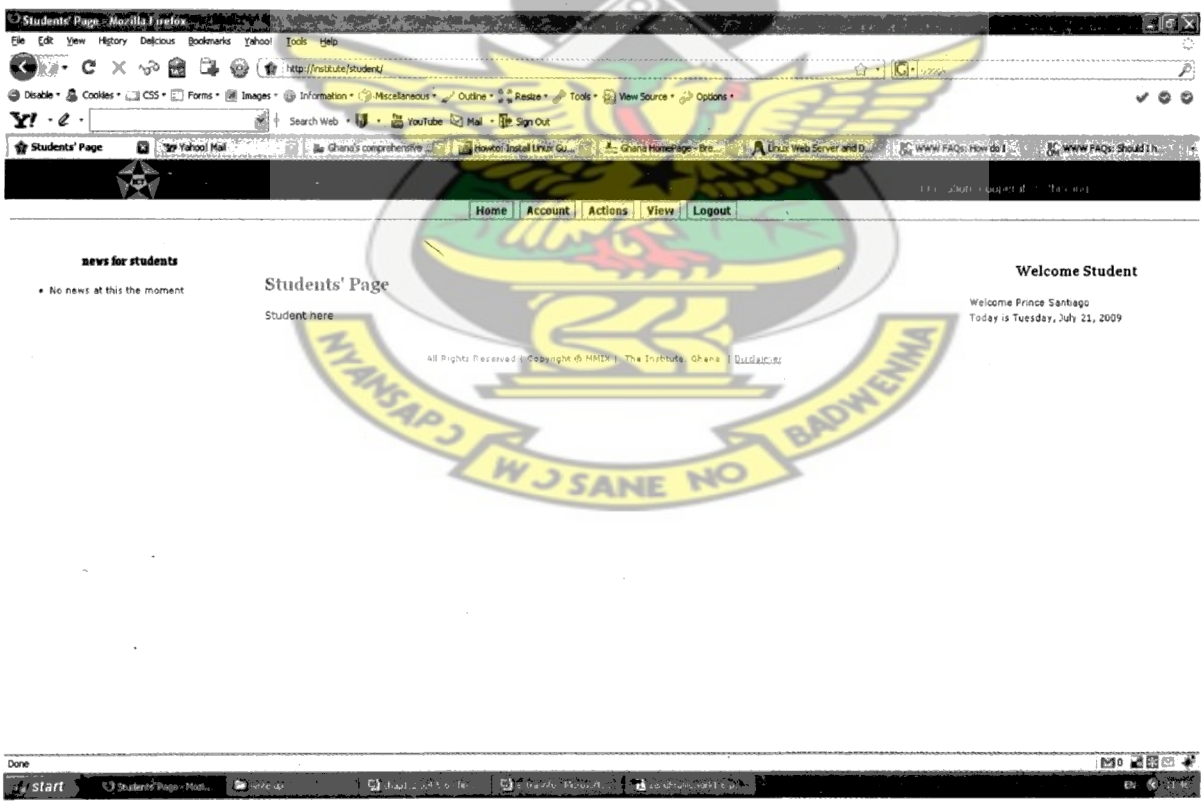


Figure 5.3: Students' Home Page

Once a student actor logs into the system, the interfaces changes to allow students do the following:

- Assess a module of a lecturer

- Check their results
- Check and print the timetable for the academic year
- Read about news and events in The Institute.
- Get the modules registered for the academic year.
- Change their account information.

5.6.4 LECTURERS PAGE(S)

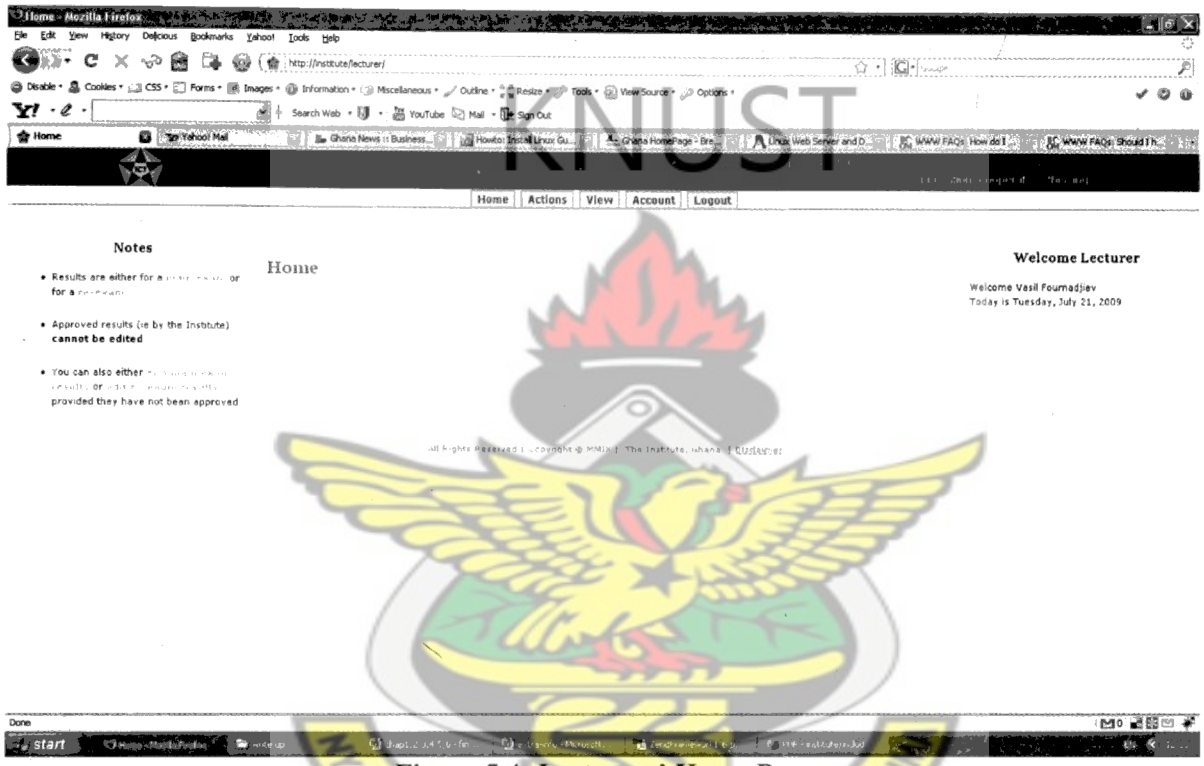


Figure 5.4: Lecturers' Home Page

A lecturer can log in into the system and once they are authenticated, they can

- Enter, edit and preview the main examination and re-examination results.
- Have lecturer news and events
- Preview timetable
- Change their account information

5.6.5 ADMINISTRATOR PAGE(S)

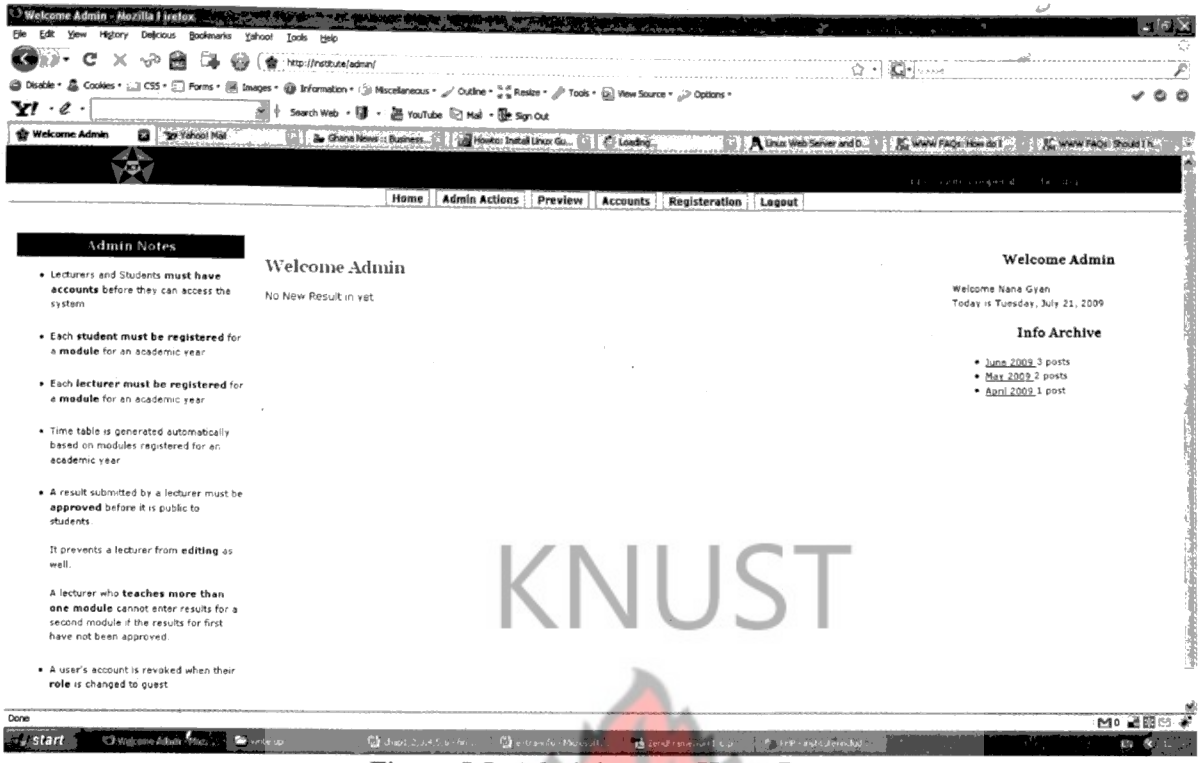


Figure 5.5: Administrator Home Page

A logged in administrator can do the following:

- Create new accounts for both students and lecturers for login
- Register a modules for an academic year
- Register both students and lecturers for a module
- Add and edit module information
- Preview submitted results
- Approve results
- Administer forum posts and users
- Add, edit and preview site information for guests.

5.7 SYSTEM ERROR HANDLING

Several kinds of errors can occur in the day-to-day running of the system. Most of the errors would be detected by different error handling codes at various sections of the application. The objectives of handling errors include the following:

- **Notify the user that an error occurred:** whether it is a system error or user error that has occurred the user should is informed that something is wrong and that their request could not be completed.
- **Record the error:** this involves writing the error to a log file or notifying the system administrator or both.
- **Roll back the current request:** if the error occurs halfway through a client request, then any performed actions should be rolled back.

The errors are grouped into three main types namely: Database errors, Application Runtime errors and HTTP errors.

5.7.1 DATABASE ERRORS

This kind of error refers to any error relating to accessing the database server or its data. For example, the following are possible errors that may occur:

- **Connection errors:** caused because the server or network may be down, the username and password are incorrect or the database name is incorrect.
- **Query errors:** caused when the SQL being used is invalid. This is reduced to the barest minimum if the application has been correctly developed and tested.
- **Data errors:** caused by violating a constraint in the database. This may occur if a duplicate value is entered in a unique field or if data is deleted form a table that is referenced elsewhere via a foreign key. This also is not expected to occur if the application has been tested correctly.

5.7.2 APPLICATION RUNTIME ERRORS

This is a fairly broad title for basically any error (including uncaught exceptions) that occurs in code. Examples of application errors that may occur include:

- File system and permission errors such as if the application tries to read a file that doesn't exist or that it is not allowed to read or it is not allowed to write to. A file that the application reads that is in the incorrect format may also cause an error.
- If the application accesses web services on remote servers, then how well the application runs is partly dependent on these servers.

5.7.3 HTTP ERRORS

These errors occur as a result of user's request. The most common HTTP error is a 404 File Not Found Error. Other errors include all errors as specified in W3C status codes (W3C).

5.8 CONCLUSION

The chapter mainly looked at the actual implementation of the web application system. The chapter looked at how files are arranged on disk in directories and the roles they play in the WAS. The use of the ZF to set the MVC design pattern was discussed. Other relevant aspects of the system such as user authentication and authorisation, form validation procedures were also mentioned. The chapter also presented some shots of pages representing the implementation of functional requirements as mentioned in chapter four. Also looked at were system error handling methods, how they might occur and the ways to deal with them.

6.0 CONCLUSION

6.1 INTRODUCTION

The concluding chapter looks at the deployment of the application onto a live or production server. Since different web hosting server offer different support for clients' applications (that is if the application is to be hosted by a third party service provider), the necessary libraries required by the WAS are mentioned. On the other, if The Institute chooses to host the application in-house, then its pros and cons are also discussed. Recommendations and future work are also discussed.

6.2 APPLICATION DEPLOYMENT

Currently, The Institute uses the web server of a third-party service provider. Application deployment involves all the necessary steps taken to make the application accessible on the internet. This step involves making sure all libraries that the WAS uses are available on the hosting server. Issues such as unrestricted access to the server anytime should be assured so changes that are made are immediately reflected on the live server. A server to host the system should have the following requirement:

- A PHP-enabled web server
- A MySQL database server
- PHP scripting language (version 5 and up)
- A Mailing functionality (SMTP)
- PEAR packages (as mentioned in chapter two)

The web server must have support for URL rewriting capabilities (accomplished in Apache web server by *mod_rewrite* module) in order for the application to function. PHP must also have the following libraries enabled:

- GD library to generate CAPTCHA images as well as manipulating images and thumbnails.
- PEAR support as discussed in chapter two.
- PDO (PHP Data Object) support
- MySQL support

6.3 WEB HOSTING: IN-HOUSE OR THIRD PARTY

In-house hosting refers to the situation where The Institute decides not to use a third party service provider to host their web content. That has its advantages and disadvantages with the greatest advantage being that they would have complete control over their web server and its content. Using the services of a third-party also has the advantage that The Institute is relieved of the all responsibility of making sure the system is not vulnerable to attacks (software updates, administration, backups, etc).

In-house hosting would mean the following requirements are met:

- A very reliable server uptime (24hours, 7 days a week).
- A very good bandwidth
- Hardware for support

6.4 SITE ADMINISTRATION

Administration of the site is a vital aspect of any web site. It is the special part of the site that allows the people who run the site to control application data or modify how the site operates. Within the WAS, two areas are used for the purpose:

An administration area for the forum and then another for the non-forum part of the application.

6.4.1 USING APPLICATION LOGS

As discussed in chapter two, application logging is implemented as a security measure for the WAS. Auditing the log file periodically could reveal possible login failures from unauthorised users and help track system abuse. Logs files can also be used as a watch in real time when new functionality is being developed. That is especially so when debugging information cannot easily be output to the browser.

6.4.2 BACKUPS

Periodical database backups are an important aspect of site administration. This involves saving all the data in the database and saving it in a secure environment. This is necessary to forestall system halt in times of breakdown or a disaster. Tools for backups are usually provided by the host of the site (that is if third-party services are used) or there are many open source solutions available for that purpose (eg. PHPMyAdmin).

6.5 RECOMMENDATIONS

Even though the WAS has been built to satisfy most of the functional requirements mentioned in chapter four, many more features could be implemented. Suggested features include:

- A reporting functionality to report data from the database. The system at the moment has no such functionality and relies on third-party tools (it uses PHPMyAdmin to simply export data from the database to many formats including Microsoft Excel format).

- An innovative way of helping lecturers to enter students' results intuitively. The system, at the moment, allows a lecturer to enter results a student at a time. That situation could be a boring and difficult thing to do if many students are involved and the internet connection is not very good.
- The system at the moment uses the forum to discuss issues. This means that both students and lecturers discuss see what the other is discussing. Future development could include distinguishing personalized information for students and lecturers as well as the staff of The Institute.

REFERENCES

1. Allen, R. (2008). Getting Started with Zend Framework 1.5.2. Retrieved 25 November 2008, from http://akrabat.com/Zend-framework-tutorial/.

2. Alshanesky, L. (2007). Security Corner: When Dictionary Attack. PHP Architect. 6: 50-55.

3. AUC. (2009). "Home Page." Retrieved May, 2009 from http://www.ashesi.edu.gh.

4. Deitel, P. J. and H. M. Deitel (1999). Java: How To Program.

5. Fowler, M. (2009a). "P of EAA: Model View Controller." Retrieved 8 May 2009, from http://www.martinfowler.com/eaaCatalog/modelViewController.html.

6. Fowler, M. (2009b). "P of EAA: FrontController." Retrieved 8 May 2009, from http://www.martingfowler.com/eaaCatalog/frontController.html.

7. Gilmore, W. J. (2007). Beginning PHP and Oracle: From Novice to Professional, Apress.

8. GTUC. (2009). "Home Page." Retrieved 5 May 2009, from http://www.gtuc.edu.gh.

9. Hayder, H. (2007). Object-Oriented Programming with PHP5, Packt Publishing

10. Kabir, M. J. (2002). Apache Server 2 Bible, Hungry Minds Inc.

11. KNUST. (2009). "Home Page." Retrieved 5 May 2009, from http://www.knust.edu.gh.

12. Levine, M. (2006). "In Search of the Holy Grail". Retrieved 18 September, 2008, from http://www.alistapart.com/articles/holygrail.

13. Looijen, M. (1998). Information Systems Management, Control and Maintenance, Kluwer BedrijfsInformatie.

14. Malan, R. and D. Bredemeyer (2001). "Functional Requirements and Use Cases." Retrieved 16 June 2009, from http://www.bredemeyer.com/pdf_files/functreq.pdf.

15. Moore, J. (2007). Test Pattern: Model View Controller PHP Architect. 6: 46-49.

16. Ohrt, M. and A. Zmievski (2007). Smarty Manual, New Digital Group Inc.

17. Powers, D. (2006). PHP Solutions: Dynamic Web Design Made Easy, Apress.

18. UCC. (2009). "Home Page." Retrieved 5 May 2009, from http://www.ucc.edu.gh.

19. UG. (2009). "Home Page." Retrieved 5 May 2009, from http://www.ug.edu.gh/index1.php?linkid=176.

20. W3C. "Status Code Definitions." Retrieved 8 April 2009, from http://www.w3.org/Protocols/rfc2616-sec10.html.

21. W3C. "The W3C Technology Stack." Retrieved 8 June 2009, from http://www.w3.org/Consortium/technology.

22. Wiesmann, A., A. v.d. Stock et al., (2005). OWASP Guide 2.0. OWASP Guide, OWASP Foundation.

23. Wikipedia. (2009). "Business Logic." Retrieved 18 May 2009, from http://en.wikipedia.org/wiki/Business_logic.

24. Wikipedia. (2009). "JavaScript." Retrieved 7 May 2009, from http://en.wikipedia.org/wiki/JavaScript.

25. Wikipedia. (2009). "XHTML." Retrieved 7 May 2009, from http://en.wikipedia.org/wiki/Xhtml.

26. Wikipedia. (2009). "Dynamic web page." Retrieved 7 May 2009, from http://en.wikipedia.org/wiki/Dynamic_web_page.

27. Wikipedia. (2009). "Web page." Retrieved 7 May 2009, from http://en.wikipedia.org/wiki/Web_page.

28. Wikipedia. (2009). "PHP." Retrieved 23 April 2009, from http://en.wikipedia.org/wiki/PHP.

29. Wikipedia. (2009). "Model-view-controller." Retrieved 28 April 2009, from http://en.wikipedia.org/wiki/Model-view-controller.

30. Wikipedia. (2009a). "Standard." Retrieved 8 June 2009, from http://en.wikipedia.org/wiki/Standard

31. Wikipedia. (2009b). "Maintainability." Retrieved 8 May 2009, from http://en.wikipedia.org/wiki/Maintainbility.

32. Wikipedia. (2009c). "Extensibility." Retrieved 8 May 2009, from http://en.wikipedia.org/wiki/Extensibility.

33. Wikipedia. (2009d). "Search engine optimization." Retrieved 22 April 2009 from http://en.wikipedia.org/wiki/Search_engine_optimization.

34. Wikipedia. (2009e). "HTML." Retrieved 6 May 2009, from http://en.wikipedia.org/wiki/HTML.

35. Wikipedia. (2009f). "Cascading Style Sheets." Retrieved 6 May 2009, from http://en.wikipedia.org/wiki/CSS.

36. Wikipedia. (2009g). "AJAX." Retrieved 7 May 2009, from http://en.wikipedia.org/wiki/AJAX.

37. Wikipedia. (2009h). "Prototype." Retrieved 7 May 2009 from http://en.wikipedia.org/wiki/Prototype_JavaScript_Framework

38. Wikipedia. (2009i). "Smarty." Retrieved 22 April 2009, from http://en.wikipedia.org/wiki/Smarty.

39. Wikipedia. (2009j). "MySQL." Retrieved 22 April 2009, from http://en.wikipedia.org/wiki/MySQL.

40. Wikipedia. (2009k) "Functional Requirement." Retrieved 23 April 2009, from http://en.wikipedia.org/wiki/Functional_requirements.

41. Zakas, N. C. (2005). Professional JavaScript for Web Developers, Wiley Publishing Inc.

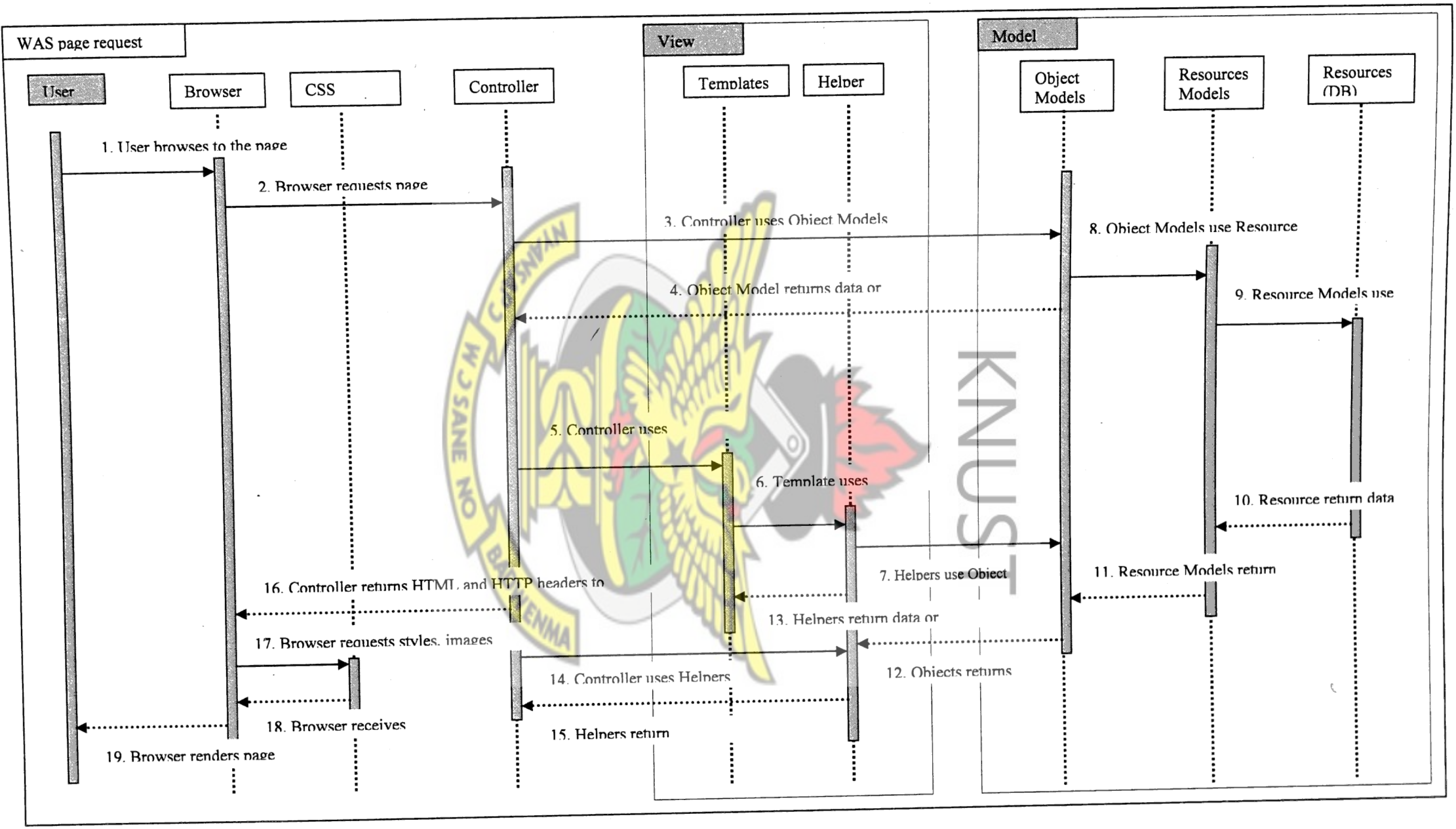
42. Zend (2008). Programmers Reference Guide: Zend Framework, Zend Technologies Inc.

43. Zervaas, Q. (2008). Practical Web 2.0 with PHP, Apress.

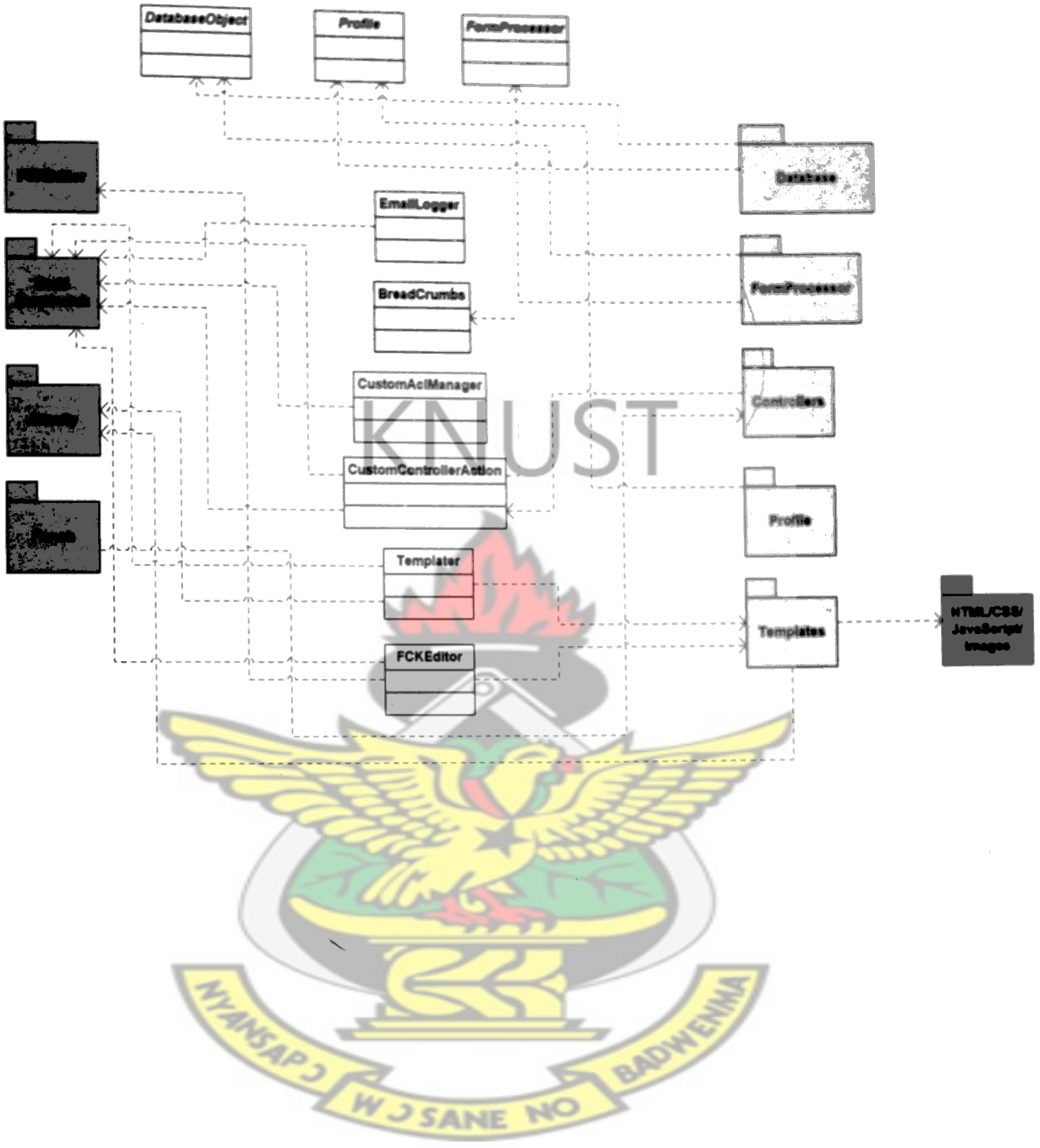
APPENDICES

- Appendix I shows the diagram of system process flows as described in paragraph 4.7.1 of chapter four.
- Appendix II shows the corresponding package dependencies also described in paragraph 4.7.2 of chapter four.
- Appendix III shows UML class diagrams and designs.

APPENDIX I – PROCESS FLOW

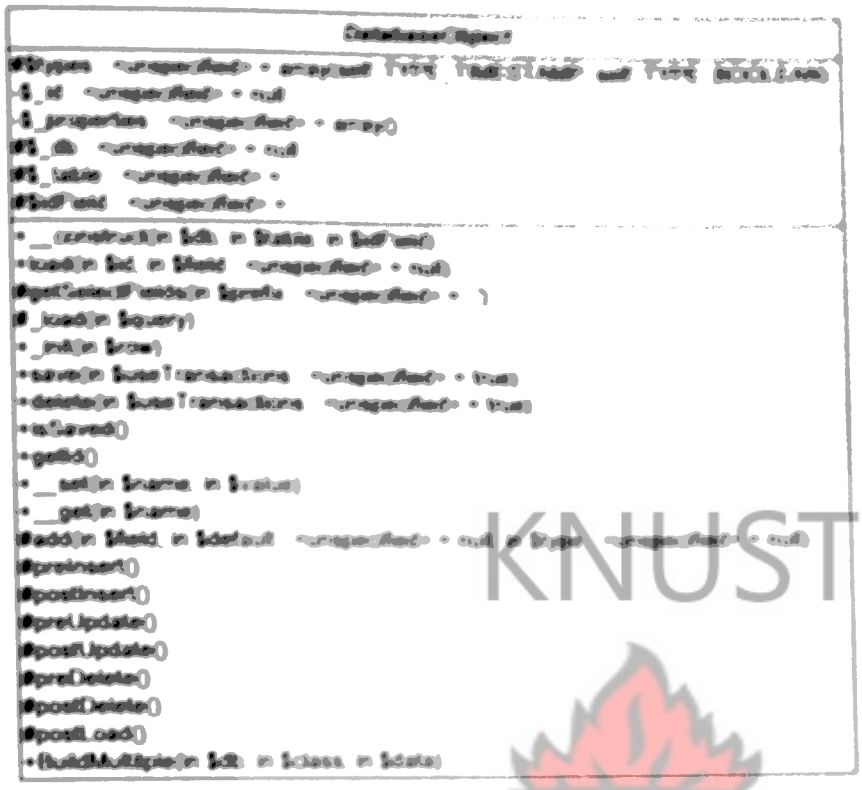


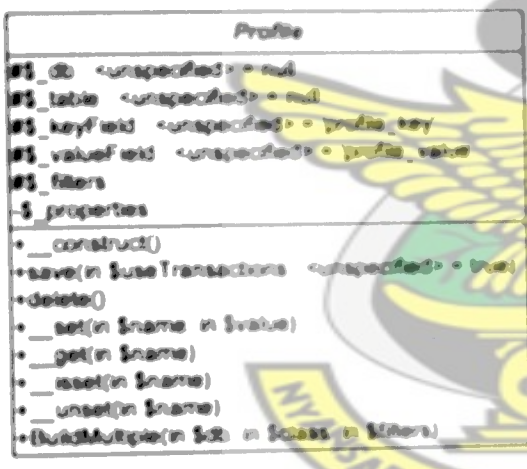
APPENDIX II – PACKAGE DEPENDENCIES

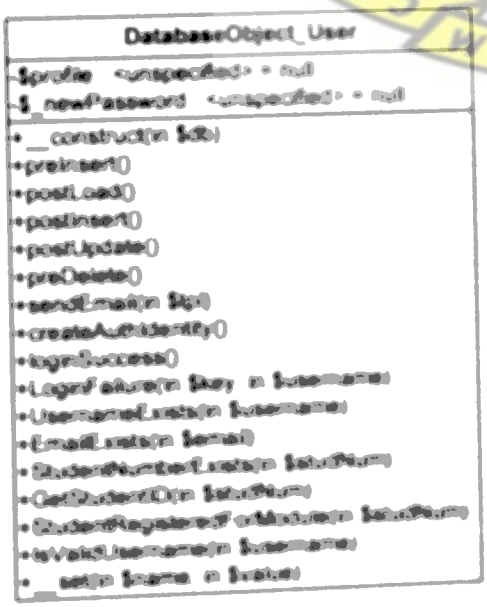


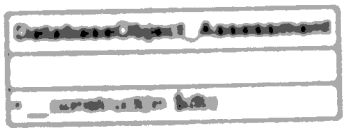
APPENDIX III - UML CLASS DIAGRAMS

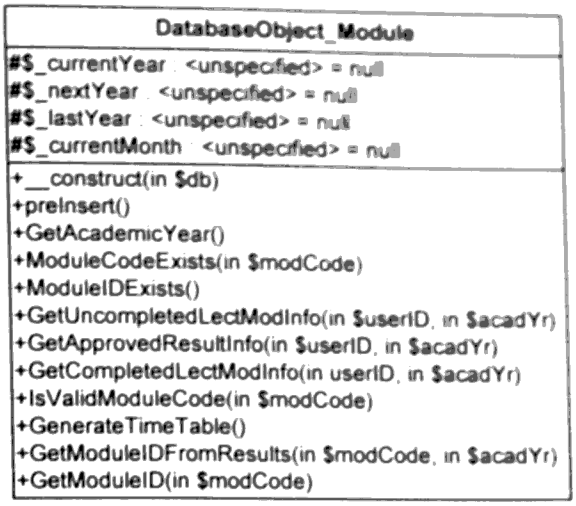
- Classes for database interactions and data manipulations

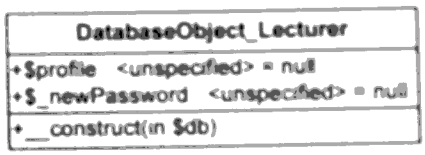


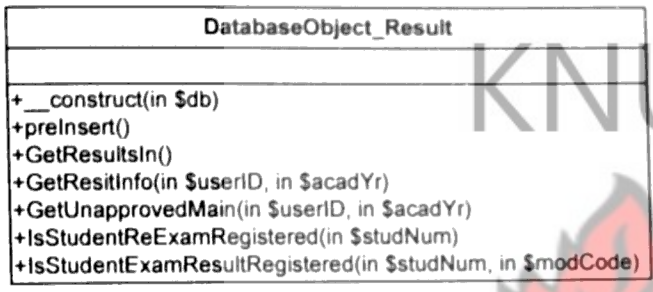


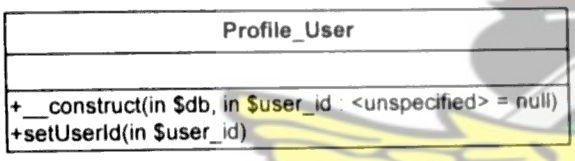




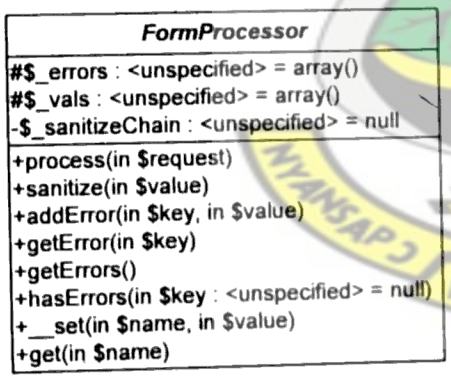


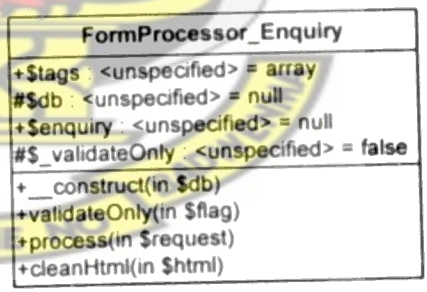


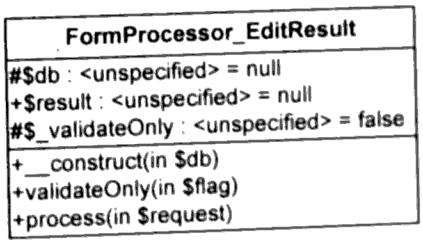


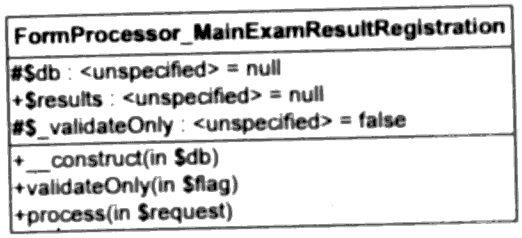


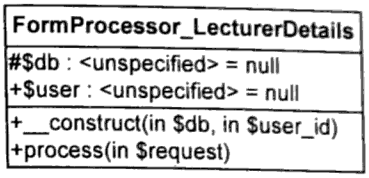
- Classes for form processing

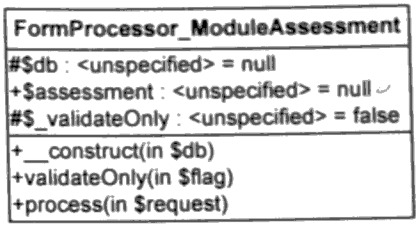


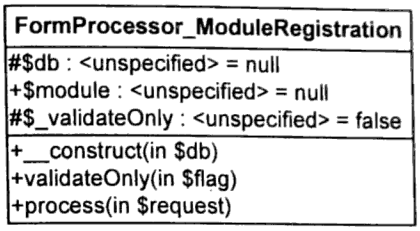


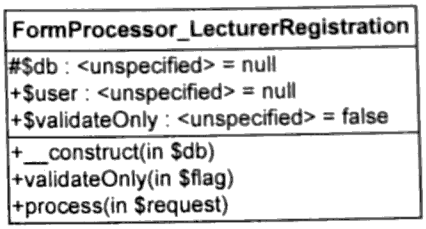


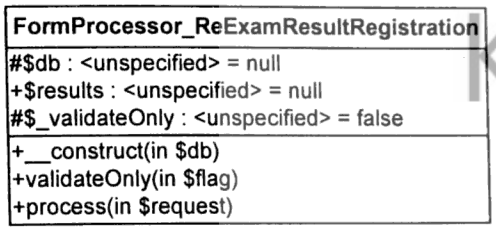


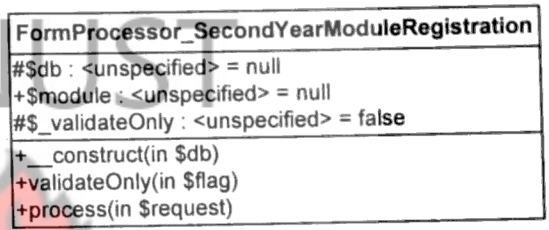


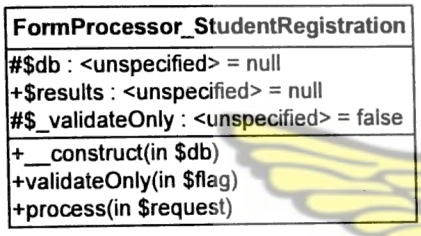


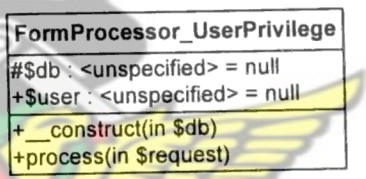


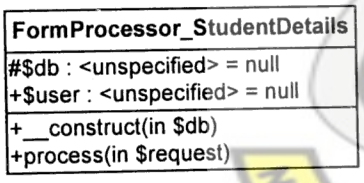




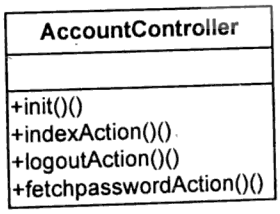


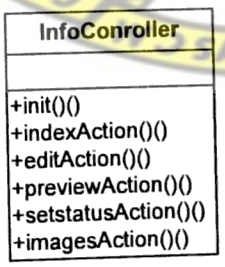


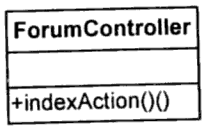


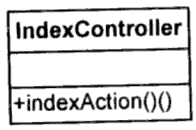


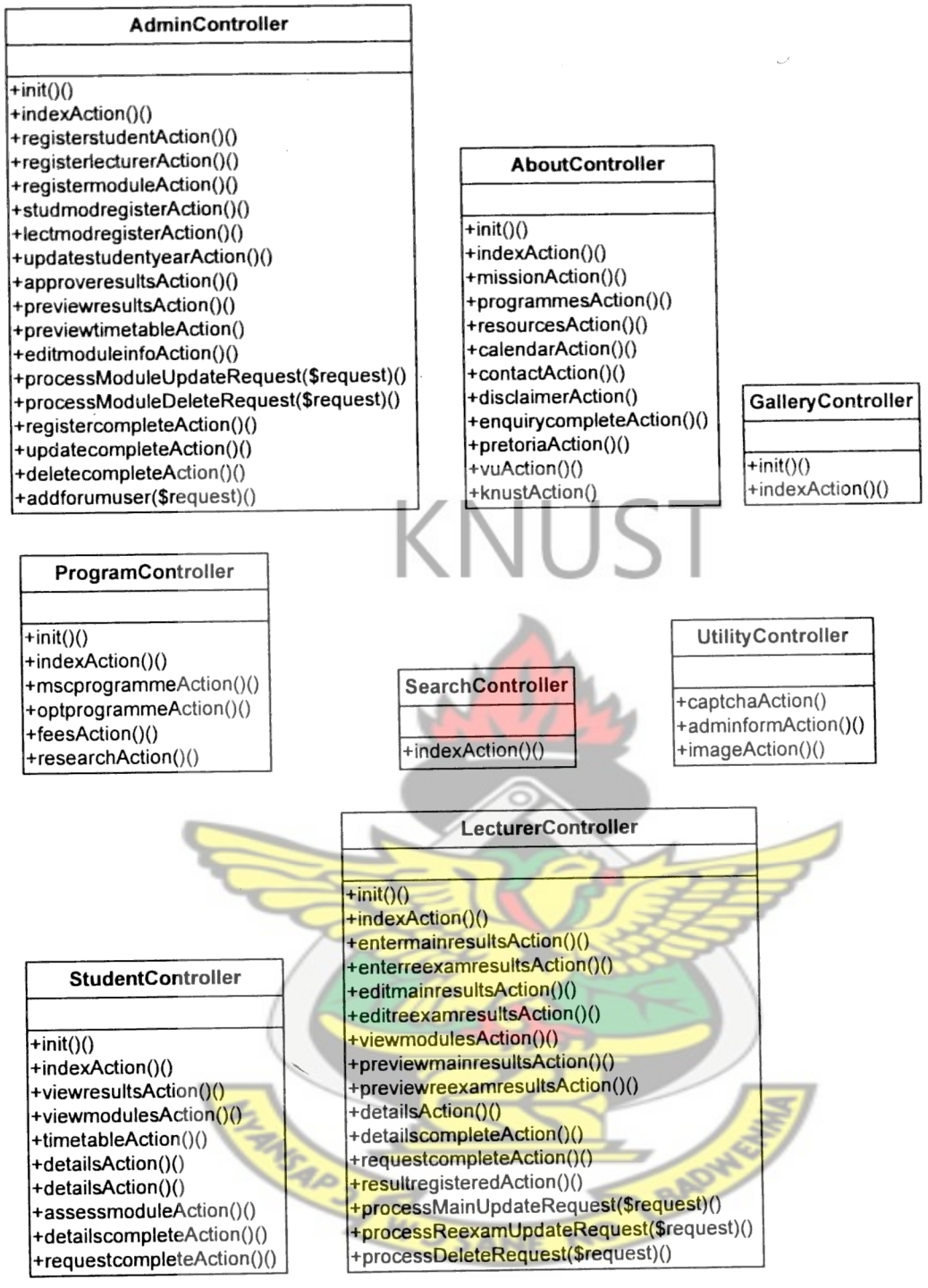
- Controller Classes



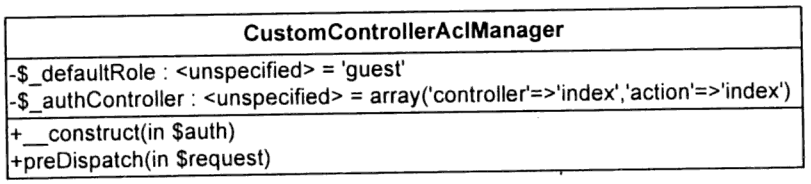








- Miscellaneous Classes



Templater
#$_path #$_engine
+__construct() : <unspecified> +getEngine()() : <unspecified> +__set(in $key, in $val) +__get(in $key) +__isset(in $key) +__unset(in $key) +assign(in $spec, in $value : <unspecified> = null) +clearVars() +render(in $name) +_run()

Breadcrumbs
-$_trail : <unspecified> = array()
+addStep(in $title, in $link : <unspecified> = '') +getTrail() +getTitle()