

**Development of a Kiswahili-user-centered Cloud-based Site Building
Platform: A Contribution to Local Content Development Initiatives in
Tanzania**

**Development of a Kiswahili-user-centered Cloud-based Site Building
Platform: A Contribution to Local Content Development Initiatives in
Tanzania**

By

Samson Josiah

**A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of
Master of Science in Information Technology and Systems (MSc-ITS) of Mzumbe
University**

CERTIFICATION

We, the undersigned, certify that we have read and hereby recommended for acceptance by Mzumbe University, a thesis entitled “**Development of a Kiswahili-user-centered Cloud-based Site Building Platform: A Contribution to Local Content Development Initiatives in Tanzania**”, in partial fulfilment of the requirements for award of the degree of Master of Science in Information Technology and System of Mzumbe University.

Major Supervisor

Internal Examiner

Accepted for the Board of _____

DEAN/DIRECTOR, FACULTY/DIRECTORATE/SCHOOL/BOARD

DECLARATION AND COPYRIGHT

I, Samson Josiah, declare that this thesis is my own work and that it has not been presented and will not be presented to any other university for a similar or any other degree award.

Signature: _____

Date: _____

This document is protected under the Berne Convention, the Copyright Act 1999 and other international and national enactments, in that behalf, on intellectual property. It may not be reproduced by any means in full or in part, except for short extracts in fair dealings, for research or private study, critical scholarly review or discourse with an acknowledgement, without the written permission of Mzumbe University, on behalf of the author.

ACKNOWLEDGEMENTS

I'm extremely grateful to God, the almighty, as all the efforts applied in this project could not work without his acquiescence. Behind this achievement there is a number of individuals to whom I should be thankful. My supervisors, Dr Mercy Mlay Komba and Mr Sensota Kivaria played a great role in making this work fruitful. I thank Dr Titus Tossy, my Research Methodologies course lecturer and all the staff in department of Computing Studies of Mzumbe University at large.

DEDICATION

I dedicate this work to my parents, Mr and Mrs Josiah Wambura.

ABSTRACT

This thesis reports about a project undertaken to develop a Kiswahili-user-centered cloud-based website building platform. An extensive study based on literature review built a basis for forming requirements for the platform. The desire to undertake this project was triggered by a call by the government of the United Republic of Tanzania on local content development, as documented in the national information and communications technology policy of 2016. The project employed a combination of certain research methodologies and system development methodologies. Literature review, experimentation and observations played a vital role in helping the researcher form a clear picture of how to develop the platform. The project employed Object-Oriented System Analysis and Design (OOSAD) approach, the incremental model of System Development Life Cycle (SDLC), entity relationship diagramming tools, web wireframes and a set of several Unified Modeling Language (UML) diagrams as system design tools. System building was accomplished using a number of languages and libraries: PHP, MySQL, HTML, CSS, JavaScript/JQuery and Bootstrap framework. System testing was done at different levels: the unit level, the integration level, and the system level. Further, white-box, black box, and grey-box testing was conducted along with dynamic testing technique. The main deliverable of this project was Kiswahili-user-centered cloud-based website building platform that is meant to serve the local community with higher level of convenience as compared to similar solutions available in the market. The researcher ultimately conducted a test with 30 participants, results of which indicated that the platform is friendly enough to serve the targeted users.

Table of Contents

CERTIFICATION	i
DECLARATION AND COPYRIGHT	ii
ACKNOWLEDGEMENTS	iii
DEDICATION	iv
ABSTRACT	v
CHAPTER ONE.....	1
INTRODUCTION	1
1.1 Background to the Study.....	1
1.2 Statement of the Problem.....	4
1.3 Research Objectives.....	5
1.3.1 General Objective.....	5
1.3.2 Specific Objectives.....	5
1.4 Research Questions	6
1.5 Significance of the Study	6
1.6 Study and System Scope	6
1.6.1 Scope of the Study.....	6
1.6.2 Scope of the System	6
1.7 Organization of the Thesis	7
1.8 Chapter Summary	7
CHAPTER TWO.....	9
LITERATURE REVIEW	9
2.1 Website Builders vs Other Web Application Building Options	9
2.2 WCMSs vs Simple Website Builders	10
2.3 Trends in Website Builders Usage.....	11

2.4 WYSIWYG Editors	13
2.5 DIY on the Web	14
2.6 PaaS: A Cloud Computing Service Model	15
2.7 Local Content Development	16
2.8 The Web Application Development Procedure	18
2.8.1 The Definition Phase	18
2.8.2 The Design Phase	18
2.8.3 The Building Phase	19
2.8.4 The Testing Phase	20
2.9 Related Works	22
2.10 Chapter Summary	23
CHAPTER THREE	24
METHODOLOGY	24
3.1 Research Methodology	24
3.1.1 Research Design	24
3.1.2 Data Collection Methods	26
3.2 System Development Methodology	34
3.2.1 Systems Development Life Cycle	34
3.2.2 Systems Analysis and Design Approach	36
3.2.3 Design Tools	37
3.2.4 Development Technologies and Tools	37
3.2.5 The Testing Methodology	38
3.3 Chapter Summary	39
CHAPTER FOUR	41
RESEARCH FINDINGS AND DISCUSSION	41
4.1 Features of WYSIWYG Website Builders	41

4.1.1 Built-in Templates.....	42
4.1.2 WYSIWYG Editor	42
4.1.3 Code Editor	43
4.1.4 Responsive Layouts.....	43
4.1.5 SEO Tools	44
4.1.6 Custom Domain Name	45
4.1.7 Social Media Integration	45
4.1.8 Form Builder	45
4.1.9 Graphical User Interface (GUI) for Adding a Map.....	45
4.2 Examination of the Identified Features in Wix, Weebly and Blogger.....	47
4.2.1 Application of Experimental Design Principles.....	47
4.2.2 Results of the Experiments.....	49
4.2.3 Discussion on the Results.....	51
4.4 Chapter Summary	53
CHAPTER FIVE	54
SYSTEM DESIGN.....	54
5.1 Analysis Modeling.....	54
5.1.1 Functional Modeling	54
5.1.2 Structural Modeling.....	57
5.1.3 Behavioral Modeling.....	59
5.2 Data Modeling	62
5.2.1 Entity Relationship Diagram.....	62
5.2.2 Normalization of Database Tables	64
5.4 Front-end Modeling	72
5.5 Chapter Summary	75
CHAPTER SIX	76

SYSTEM DEVELOPMENT AND TESTING	76
6.1 System Development	76
6.1.1 Module Building and Integration	76
6.1.2 Enhancement of the Text Editor.....	76
6.1.3 Database Implementation.....	77
6.1.4 System Builds.....	77
6.1.5 Results Obtained	79
6.2 System Testing.....	83
6.2.1 Test Cases.....	83
6.2.2 Project Achievement Test	88
6.3 Chapter Summary	93
CHAPTER SEVEN	94
CONCLUSIONS AND RECOMMENDATIONS	94
7.1 Conclusions.....	94
7.2 Recommendations.....	95
7.3 Chapter Summary	95
REFERENCES	96
APPENDICES	108
Appendix I: Equivalent Textual Representation for Figure 13.....	108
Appendix II: Project Timeline	110
Appendix III: Project Budget.....	110

List of Tables

Table 1. Yearly trends in the usage of website builders from January 1st, 2011 to April 17th, 2018 (“Historical yearly trends,” 2018)	13
Table 2. Inclusion Criteria	28
Table 3. Exclusion Criteria	28
Table 4. Summary of Reviewed Literature	29
Table 5. Research and Development Methodology Summary	39
Table 6. Inclusion of Website Builder Features in Wix, Weebly and Blogger	50
Table 7. Table Name: Project Metadata (Unnormalized).....	64
Table 8. Table Name: Project Metadata (1NF)	65
Table 9. Table Name: Owners (2NF)	66
Table 10. Table Name: Projects (2NF).....	66
Table 11. Table Name: Owners (3NF)	67
Table 12. Table Name: Projects (3NF).....	67
Table 13. Table Name: cbsbp_jumadavAtgmailDotcom_s1 (Unnormalized)	68
Table 14. Table Name: Pages (1NF)	69
Table 15. Table Name: Page Text(1NF)	69
Table 16. Table Name: Layer Colors(1NF).....	69
Table 17. Table Name: Pages (2NF)	69
Table 18. Table Name: Page Text(2NF)	70
Table 19. Table Name: Layer Colors(2NF).....	70
Table 20. Table Name: Textual Image Representation (3NF)	70
Table 21. Table Name: Pages (3NF)	71
Table 22. Table Name: Page Text (3NF)	71
Table 23. Table Name: Layer Colors (3NF).....	71
Table 24. Incremental Builds of the Platform	77

Table 25. A test case for the login module	84
Table 26. A test case for Web Project Creation module	85
Table 27. A test case for Web Project Removal module.....	86
Table 28. A test case for the Web Page Creation module	87
Table 29. A test case for Web Page Removal module	88
Table 30. Task accomplishment times with Kiswahili versus English as a Service Language	90
Table 31. Summary of Task accomplishment times for all thirty users	93

List of Figures

Figure 1: Web Application Building Options (“Web App Framework,” n.d.).	10
Figure 2: SDLC Phases (SDLC – Overview, n.d.).	35
Figure 3: The SDLC Incremental Model (SDLC - Iterative Model, n.d.).	36
Figure 4: Relationship between an external quality attribute and internal attributes of an application (Sommerville, 2011)	42
Figure 5: Sample Meta Title and Meta Description (Benjamins, 2018)	44
Figure 6: Use Case Diagram for the Website-building Platform	55
Figure 7: Activity Diagram for the Website-building Platform	56
Figure 8: Class Diagram for the Website-building Platform	58
Figure 9: State Machine Diagram for User Account	60
Figure 10: State Machine Diagram for a Web Project	61
Figure 11: State Machine Diagram for a Web Page	61
Figure 12: An ERD for the Website-building Platform	63
Figure 13: A sample image whose Textual Representation can be determined by the Platform.	72
Figure 14: A Wireframe for the Platform's Home Page	73
Figure 15: A Wireframe for the Platform's Sign in Page	73
Figure 16: A Wireframe for User's Dashboard	74
Figure 17: A Wireframe for the Page Editing/ Page Creation Interface (The WYSIWYG editor)	74
Figure 18: The template-selection page of the platform	79
Figure 19: An editable page on a WYSIWYG editor	80
Figure 20: The Code Editor of the Platform	80
Figure 21: The Advanced Options (“Opsheni za Ziada”) Modal with which a user can enter meta keywords and description, among other options.	81
Figure 22: A customizable Photo Gallery	82

Figure 23: User Dashboard with several options for each website, including the site preview (Tembelea) option..... 83

List of Abbreviations

ASP	Active Server Pages
CMS	Content Management System
COED	Concise Oxford English Dictionary
CRC	Class-Responsibility-Collaboration
CSS	Cascading Style Sheets
DBMS	Database Management System
DIY	do-it-yourself
ERD	Entity Relationship Diagram
FAQ	Frequently Asked Questions
GUI	Graphical User Interface
HISs	Human Interaction Services
HTML	Hypertext Markup Language
IaaS	Infrastructure-as-a-Service
ICT	Information and Communications Technology
IS	Information System
JSP	Java Server Pages
MySQL	My Structured Query Language
NICTP	National Information and Communications Technology Policy
OOSAD	Object-oriented Systems Analysis and Design
PaaS	Platform-as-a-Service
PC	Personal Computer
PDF	Portable Document Format
PHP	PHP Hypertext Preprocessor
PS	PostScript
RAD	Rapid Application Development

RTM	Requirement Traceability Matrix
SaaS	Software-as-a-Service
SDLC	Software Development Life Cycle
SEO	Search Engine Optimization
SRS	Software/System Requirement Specification
STLC	Software Testing Life Cycle
TBSSs	Technology-Based Self-Services
UML	Unified Modeling Language
UNECA	United Nations Economic Commission for Africa
UNESCO	United Nations Educational, Scientific and Cultural Organisation
URL	Uniform Resource Locator
URT	United Republic of Tanzania
VGA	Video Graphics Array
WCMS	Web Content Management System
WWW	World Wide Web
WYSIWYG	What You See Is What You Get

CHAPTER ONE

INTRODUCTION

This chapter presents background information related to the study, general and specific objectives of the research project, research questions, significance of the study and study and system scope.

1.1 Background to the Study

The spread of computers and activation of the Internet in the 20th century triggered a desire to own websites among individuals and organizations (Yang et al., 2016). Unfortunately, one needed to be a computer programmer, or hire a computer programming expert to have his website built (Naithani & Kaushik, 2016). This made it necessary to seek an alternative that would allow people to easily create websites, irrespective of their technical expertise. Website building applications emerged as a means to help people build websites with ease, while saving a significant amount of time (Naithani & Kaushik, 2016; Shah, 2012; Laguador et al., 2016).

Today, information technology world witnesses a couple of website construction approaches from which a person in need of a website can select one in accordance with his desire and level of skills. A person may decide to write a web application from scratch, use a feature-rich web content management system (WCMS) such as Drupal, Joomla or WordPress, or use a simple website building application such as Artisteer, SimpleSite or Adobe Dreamweaver (GeoTrust, 2011; Gambhir & Singh, 2014). In addition to these options, there exist web application frameworks such as Ruby on Rails. With such frameworks, a user has to deal with codes but has an advantage of being presented with re-usable components to assist him develop a website quickly and according to best practices (“Web App Framework,” n.d.). With WCMSs and simple website builders, website building is so simplified that even people with no computer

programming skills can make their own websites in just a matter of minutes (Martinez-Caro et al., 2018). What You See Is What You Get (WYSIWYG) editors embedded in most website builders make the web-page construction task even easier and quicker. WCMSs however go further than just building web pages (Ghorecha & Bhatt, 2013). In fact, most WCMSs are meant to allow management of web content rather than construction of web pages. However, since it is possible for one to have a fully-functioning website using a WCMS, then WCMSs qualify to be named website builders (“Types of Website Builder,” n.d.).

Website building applications range from those meant to be installed on users’ computers, such as Adobe Dreamweaver or Artisteer, to those operating online, such as Wix. There are also those which can operate on either environment: online environment as well as desktop environment. Examples of these are Joomla and WordPress (Martinez-Caro et al., 2018; Bulan, 2018; Nasit, 2018). Because the desktop applications are installed on users’ computers, they allow creation of websites even if these computers are not connected to the Internet. In this way, a website being created is saved in files which, upon completion, will be sent to a hosting company so that the website is made available online using a special computer called a “server”. Since this server stores website files, which are to be accessible over the Internet via the World Wide Web (WWW), it is referred to as a “web server” (Tomsho et al., 2007). Online website building applications on the other hand do not need to be installed on users’ computers. Instead, they reside on remote computers, and users can make use of the Internet to access and use them. Normally, companies hosting online website builders host websites as well. This makes it possible for a user to build and host a website in the same platform, and hence avoiding a burden of having to seek a hosting company after the website has been built.

The provision of an online platform that allows remote clients to build websites is a service falling under one of the three service models of cloud computing. Such a model is named Platform-as-a-Service (PaaS). The other two models are Software-as-a-Service

(SaaS) and Infrastructure-as-a-Service (IaaS). A definition given by Goyal (as cited in Shirazi & Iqbal, 2017) portray cloud computing as the delivery of computing services to a wide range of users aiming at providing them with distributed access to scalable and virtualized hardware and/or software over the Internet. PaaS is therefore the delivery of a computing platform and solution stack as a service to allow development and testing of applications in the cloud (Nazir, 2012). Operating “in the cloud” simply means operating online. Website builders operating online can thus be named cloud-based site building platforms.

Building a website using a cloud-based site building platform is one of the so-called web-based self-services. Additionally, since this service allows people to create websites on their own, it is also considered to be one of the do-it-yourself (DIY) services (GeoTrust, 2011; Wolf & McQuitty, 2011). According to Gerber and Martin (2012), people are likely to be motivated to use a certain web-based self-service if they find it to be friendly enough to make them easily obtain a product that is a successful result of their efforts. User-friendliness is therefore one of the paramount factors that can make one adopt a particular web-based self-service, such as an online website builder. Just like desktop WYSIWYG website builders, most cloud-based WYSIWYG website builders are user-friendly for they are characterized by most of the attributes that are necessary to make an application friendly to the end-user. Unfortunately, a few user-friendliness issues have not been well-addressed by most of the website builders. Familiarity is one of such issues. A website builder needs to use a language that’s familiar to a user, to be friendly to that user. Tanzania, in particular, is a country whose residents are mainly native users of Kiswahili (Chepkemai, 2017). They thus would need website-building platforms that use Kiswahili as a service language.

1.2 Statement of the Problem

Any electronic service becomes less-friendly to end-users if the language used in delivering it is not familiar to the intended recipients. Galitz (2007) and Bhaskar et al. (2011) describe familiarity as one of the fundamental features to be taken into account when attempting to have effective graphical and web user interfaces. They maintain that the use of a language that's familiar to the user is one of the core aspects to be considered to achieve the "familiarity goal". As per Tanzanian National Information and Communications Technology Policy (NICTP) of 2016, most of electronic services offered in the country use English language as a service language (United Republic of Tanzania [URT], 2016). As a result, most of the users of these services face a language-barrier problem. Following this, the government of Tanzania promises to "put in place an effective mechanism to promote the use of Kiswahili in electronic services for transformation of Tanzania into knowledge society" (URT, 2016, p. 28).

Simple WYSIWYG website building platforms are meant to enable their users easily create web pages. However, this ease is not experienced in some portion of the market due to language differences: language used on a given platform may not be a native language of the prospective platform users in a certain geographical area. As stated earlier, the NICTP of 2016 reveal the predominance of English language in electronic services in the country (URT, 2016). Web-based self-service platforms are no exception. English language has dominated most of the web-based self-service platforms, including cloud-based website builders. Most of these builders use English as a default platform language, or the only language of the platform. Despite having a considerable community of users (Chepkemai, 2017), Kiswahili is not included as one of the service languages in most of the cloud-based site building platforms.

Concisely stating, there is only a limited number of cloud-based site building platforms that include Kiswahili as one of the service languages. Among all the website builders the researcher came across, only Blogger, a blog-publishing platform owned and hosted by Google, was found to have an option of using Kiswahili as a service language.

However, Blogger is limited in varieties of templates. For instance, Blogger does not offer any built-in template with parallax effect, which is one of the commonest scroll effects in modern websites (Nordahl, 2016; Ko & Cheeyong, 2015). A user with no coding experience would therefore need to either look for a third-party template provider, or add a widget from an external site, to have a parallax effect to his Blogger site. This can be evidenced by Adam's contribution on Google product forums, who establishes that for one to have a parallax effect on his Blogger weblog, he has to tweak things to achieve such a goal (Adam, 2018).

This made it necessary to conduct a study that would investigate and reveal important features of WYSIWYG site building platforms, and thereafter developing a “Swahili platform” to incorporate these features to the largest possible extent.

1.3 Research Objectives

1.3.1 General Objective

The main objective of the study was to develop a cloud-based site building platform targeting users whose first language is Kiswahili, thereby complying with the Tanzanian national information and communications technology policy, particularly on local content development.

1.3.2 Specific Objectives

The following were specific objectives of the study:

- i. To find out important features of WYSIWYG site builders.
- ii. To examine the inclusion of these features in three selected cloud-based site building platforms: Wix, Weebly and Blogger.
- iii. To design a Kiswahili-user-centered cloud-based site building platform that incorporates as many of the features included in the examined platforms as possible.
- iv. To build the designed platform.
- v. To test the platform.

1.4 Research Questions

The study aimed at finding answers to these questions:

- i. What features are important in any WYSIWYG site builder?
- ii. Which of these features are included in Wix, Weebly and Blogger?
- iii. How can a cloud-based site building platform be designed?
- iv. How can a cloud-based site building platform be built and tested?

1.5 Significance of the Study

Findings from the study built a blueprint that guided the development of an information system (IS) that's meant to act as a platform which will enable native users of Kiswahili to build web applications with ease, and by themselves. The system will contribute to local content development efforts in Tanzania, thereby complying with the NICTP of 2016. Further, methodology employed in this research project may act as a guide that will provide a basis for development of similar systems in the future.

1.6 Study and System Scope

1.6.1 Scope of the Study

The study was limited to discovering important features of WYSIWYG site builders, and examining the inclusion of these features in only three of the prominent cloud-based website building platforms: Wix, Weebly and Blogger. The study did not make direct comparison between these platforms, nor did it recommend which one of them is the best to use.

1.6.2 Scope of the System

The system built is not a WCMS. It is a simple website builder. Being “cloud-based”, the system is meant to operate online, and no desktop version has been built. Further, not all features that were found in the named cloud-based website builders were implemented in the system. The researcher only selected those features whose implementation would be feasible with the available time and skill-set.

The project made use a few ready-made application libraries, templates, images and functions. In particular, two libraries were used: “jscolor” and “simple HTML DOM”. Jscolor was meant to add background-coloration capability to the system while the HTML DOM library was meant to enhance the system in terms of web-scraping. Building new templates was out of scope of this project. The researcher therefore decided to look for some freely-available templates and make them content-editable so that they can be used in the system. Templates normally appear stunning when they are preloaded with nice images. Therefore, the researcher made use of some attribution-free images found on the Internet. Finally, the project employed a few ready-made functions, whenever necessary and applicable.

1.7 Organization of the Thesis

The thesis is organized in seven chapters. Chapter One introduces the study by presenting background information, research objectives, research questions, significance of the study as well as scope of the study and system scope. Chapter Two reviews literature related to the topic. In addition, this chapter reviews several works/systems related to the built system. Chapter Three presents a set of methodologies employed to achieve the goal of the research, as well as methodologies for system development. Chapter Four presents findings of the study and discussion on the findings. The fifth chapter presents system design, and the sixth one documents implementation of such design. The final chapter presents conclusions and recommendations for forthcoming studies.

1.8 Chapter Summary

The introductory chapter was meant to introduce the study. This has been accomplished by presenting background information, general and specific objectives of the research, research questions, significance of the study as well as study and system scope. Background information reveals that ever since website building applications were innovated, website building has been so simple that even a person with no coding skills

can make a website by himself. This innovation however has been less advantageous to some locations on the world due to familiarity problem brought about by language-barrier problem. An instance of these locations is Tanzania, where the most commonly used language is Kiswahili. One of the issues the government of Tanzania is struggling to achieve is having Kiswahili language included in as many of the electronic services in the country as possible. This chapter gave justification to why the “Swahili website builder” was necessary in contributing to efforts that are being employed to achieve this desire of the government. The chapter that follows reviews both the literature related to the topic, and similar works that have ever been done, thereby providing a basis for formulation of the research gap.

CHAPTER TWO

LITERATURE REVIEW

In this chapter, the researcher summarizes what was found in various sources written by different authors. The summary is organized in sections and sub-sections in such a way that it forms a basis for understanding the topic under investigation. A few works that are closely related to this project are reviewed at the end of the chapter before stating the gap.

2.1 Website Builders vs Other Web Application Building Options

Website builders (WCMSs and simple website builders) can be very helpful when it comes to creating online presence. There are two other alternatives that can be used for the same purpose: using a web development framework, and coding an application from scratch. Among all these options, using a simple website builder is the easiest and quickest method. Using a WCMS is the next easier and quicker option as using a framework or writing codes from scratch would require much more efforts and time. Figure 1 portrays the workload and level of control attributed to each of these options, in comparison to the other options. In the figure, brick width portrays the level of control that each category provides while brick height portrays the workload (coding work) needed to create a site (“Web App Framework,” n.d.). By “level of control”, it means the flexibility level one would have in a given option, with respect to what he can do. A low level of control implies less flexibility in controlling what a site does whereas a full control means being able to make a site behave whichever way one likes (or do whatever desired). Coding from scratch provides full control of a site, while using a simple website builder provides least flexibility as compared to all other options (“Web App Framework,” n.d.). However, the lower the level of flexibility, the simpler to use the option is. Website builders therefore become the best option, when it comes to ease of use.

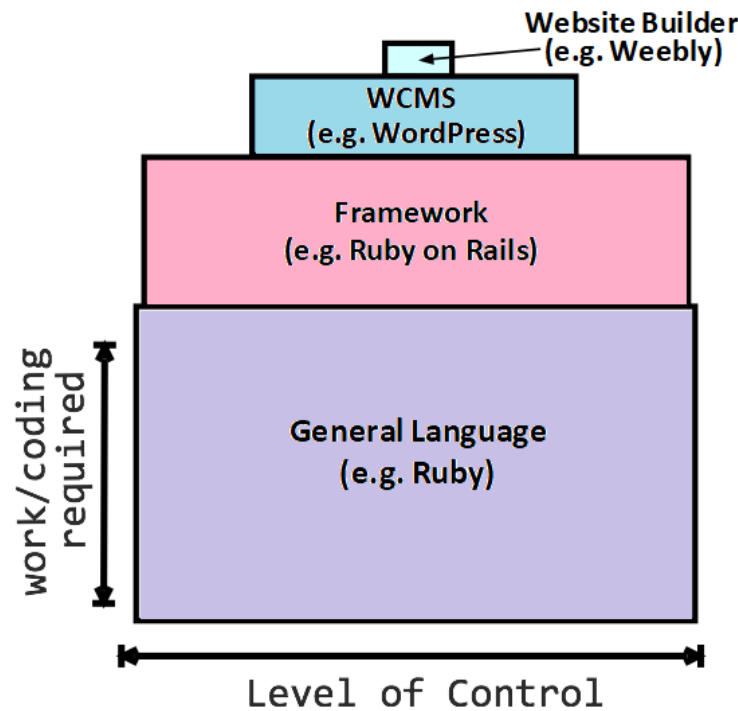


Figure 1: Web Application Building Options (“Web App Framework,” n.d.).

2.2 WCMSs vs Simple Website Builders

It is somewhat hard to give a clear-cut distinction between WCMSs and simple website builders. There is a vast number of website builders, but not all of them are WCMSs. That is, there are website builders which do not play the “content management” role well enough to be called WCMSs. These are referred to as “web-authoring tools/software” (Giri & Nirgude, 2009; Olusola & Sunday, 2013), or just “website builders”. This thesis refers to these builders as “simple website builders”.

In their paper, Srivastav and Nath (2016) define a WCMS as a program that helps in maintaining, controlling, changing and resembling content on a web-page. Baldaniya and Baldaniya (2014) define a WCMS as a bundled or stand-alone application that is used to create, manage, store and deploy content on web-pages. Additionally, Rohilla (2017) defines a WCMS as a specific CMS (Content Management System) that focus on management of digital data for web applications. It can be figured out from these

definitions that, a WCMS does more than just building websites; it also allows for advanced management of contents of a built website. On the other hand, a simple website builder is a tool meant to help a user easily put together a site using pre-built components (Litwin, 2017). Olusola and Sunday (2013) maintain that simple website builders are inadequate in situations where websites are very dynamic, with different content types.

In a nutshell, simple website builders are tools that allow creation of simple websites (and therefore are very easy to use), whereas WCMSs are meant to create powerful websites (but are relatively hard to use) (Expert Market, 2017). Further, as compared to a simple website builder, a WCMS is considered to be a more appropriate tool for a website whose content is likely to change frequently (GeoTrust, 2011).

Ambiguity in classifying website building solutions does not always hold as some applications can easily be identified as either simple website builders or WCMSs. Sometimes, though, it's hard to tell whether a certain solution is a simple website builder or a WCMS. Wix, for example, is perceived not to be a WCMS per se ("Request," n.d.). However, W3Techs lists Wix as one of the fastest growing WCMSs ("Fastest growing," 2018). This is different from tools like Adobe Dreamweaver and Joomla, whose classification should be apparent based on the definitions of WCMSs and simple website builders (Adobe Dreamweaver is a simple website builder while Joomla is a WCMS).

Some simple website builders are interoperable with some WCMSs. Examples of these include Artisteer, whose templates can be exported to Joomla, Drupal and Wordpress (Martinez-Caro et al., 2018). This complements the WCMSs, as most of them are not rich in built-in templates.

2.3 Trends in Website Builders Usage

Website builders emerged as a means to help people build websites with ease, while saving a significant amount of time (Naithani & Kaushik, 2016; Shah, 2012; Laguador

et al., 2016). Before they were invented, construction of websites was relatively difficult as coding was inevitable during the process (Naithani & Kaushik, 2016). Simple website builders were invented to simplify the web-page/website construction task (Litwin, 2017). Later on, advanced tools were needed to allow creation, management, and distribution of various forms of content on the web. This led to the invention of WCMSs (Giri & Nirgude, 2009).

While some WCMSs and simple website builders are becoming more and more popular, a few have experienced a steady decline over the last few years. As per W3Techs statistics, WordPress is the fastest growing website building platform. As of January 1st, 2011, 13.1% of all websites were powered by the platform. WordPress popularity has been drastically increasing to an extent that it was reported to be powering 30.5% of all websites as of April 17th, 2018 (“Historical yearly trends,” 2018). W3Techs, a web technology survey firm, is dedicated to collecting information about the usage of various types of technologies used for building and running websites, and publishing data on its findings (Sharwood, 2018). Table 1 is an excerpt from W3Techs tabular statistics regarding yearly trends in the usage of different simple website builders and WCMSs (although W3Techs considers all these to be WCMSs).

Table 1. Yearly trends in the usage of website builders from January 1st, 2011 to April 17th, 2018 (“Historical yearly trends,” 2018)

	2011 1 Jan	2012 1 Jan	2013 1 Jan	2014 1 Jan	2015 1 Jan	2016 1 Jan	2017 1 Jan	2018 1 Jan	2018 17 Apr
WordPress	13.1%	15.8%	17.4%	21.0%	23.3%	25.6%	27.3%	29.2%	30.5%
Joomla	2.6%	2.8%	2.8%	3.3%	3.3%	3.3%	3.4%	3.2%	3.1%
Blogger	0.6%	0.9%	1.1%	1.2%	1.1%	1.2%	1.1%	0.9%	0.9%
Wix			<0.1%	0.1%	0.1%	0.2%	0.3%	0.4%	0.6%
Adobe Dreamweaver						0.6%	0.6%	0.5%	0.5%
Weebly			<0.1%	0.1%	0.1%	0.2%	0.2%	0.2%	0.3%
DataLife Engine	0.4%	0.5%	0.5%	0.4%	0.3%	0.3%	0.3%	0.3%	0.3%
phpBB	0.4%	0.4%	0.4%	0.3%	0.2%	0.2%	0.2%	0.2%	0.2%
Discuz!	0.4%	0.5%	0.4%	0.3%	0.2%	0.2%	0.2%	0.2%	0.1%
osCommerce		0.4%	0.3%	0.3%	0.2%	0.2%	0.1%	0.1%	0.1%

The data reveals that some website builders such as Wix and WordPress are consistently gaining more market share. Others, however, are constantly losing their market share (examples of these are phpBB and osCommerce). Some, like Blogger and DataLife Engine, are experiencing rise and fall time after time. Generally, the use of website builders seems to be increasing as the percentage of websites that do not use any of the 72 website builders listed by W3Techs fell from 76.4% (as of January 1st, 2011) to 49.2% (as of April 17th, 2018) (“Historical yearly trends,” 2018).

2.4 WYSIWYG Editors

To achieve ease of use, website builders include WYSIWYG editors. These tools make the content creation and content editing task easy and quick. With WYSIWYG editors, even users with no coding skills can create and edit web content (Baldaniya & Baldaniya, 2014). As per Concise Oxford English Dictionary (COED) definition, the acronym denotes the representation of text on-screen in a form exactly corresponding to its appearance on a printout (Duckett, 2005). However, what can be represented on a WYSIWYG editor is not only text; even some other sorts of content, such as images, can be represented. Appearance of a document on a WYSIWYG editor/screen during its

creation or editing will match its appearance when printed or formatted to a printable file format such as PostScript (PS) or Portable Document Format (PDF). Such resemblance is said to be a 1:1 correspondence between the two as anything visible on the screen will appear in the printout in an identical form (Flynn, 2014). In their white paper, GeoTrust (2011) claim that using a WYSIWYG editor is as simple as using a word processing program. They further maintain that some WYSIWYG editors are so advanced that they allow arrangement of items using “drag-and-drop” tools. With these tools, a user can easily move images, videos, highlighted text, and other types of movable items to desired positions on the screen.

2.5 DIY on the Web

As cited by Wolf and McQuitty (2011), Vargo and Lusch reveal that consumer involvement in co-creation of value is one of the important marketing techniques. Some authors (Fox, 2013; Wolf & McQuitty, 2011; Williams, 2008; Watson & Shove, 2008; GeoTrust, 2011) describe this concept as “Do-it-yourself (DIY)”. The term “co-creation” denotes a situation whereby a prospective consumer of a given product or service is involved in the process of creating that product or service. This, in turn, increases sense of accomplishment among consumers of these products and services. Wolf and McQuitty (2011), Watson and Shove (2008), Williams (2008), and Fox (2013) call these consumers “DIYers”. A working paper by Norton, Mochon and Ariely, as well as a study by Williams (as cited in Fox, 2013) lists some more benefits of DIY. These include: providing a means for many people to do what previously only a few people could do, raising the level of satisfaction and pride of ownership among consumers, increasing consumers’ authority over when and where production activities are carried out, as well as lowering costs of goods and services.

Businesses increasingly offer consumers the opportunity to co-produce self-services using online technologies (van Beuningen et al., 2009). DIY and web-based self-services (also called “Internet-Based Self-Services” [Cardoso & Miller, 2012]) are closely-related concepts: people “serving themselves” different web-based services are

“doing-it-themselves”. This thesis names these people “web DIYers”. Web DIYers are interested in different web services, such as building one’s own website using another website.

A study by Kattara and El-Said (2014) explored customers’ preferences for new technology-based self-services (TBSSs), such as web-based self-services, against human interaction services (HISs) in hotels. Although the study found that customers prefer to contact an employee (i.e. HIS) rather than depending on technology (i.e. TBSS) in the majority of service encounters, it still discovered a few benefits of TBSSs: speed and ease of use. The “ease of use” advantage is supported by Gerber and Martin (2012) who argues that people are likely to be motivated to use a certain web-based self-service (which is one of the types of TBSSs) if they believe they are able to use it.

Most of the giant web-hosting companies in the market, including GoDaddy (<https://www.godaddy.com>), HostGator (<https://www.hostgator.com>), and Web Hosting Geeks (<https://webhostinggeeks.com>), supplement their services with website builders (Jiang & Dong, 2008). The builders enable users to build themselves websites online, and are thus one of the kinds of DIY web-based self-service platforms.

2.6 PaaS: A Cloud Computing Service Model

Cloud computing provides computing services while relying on a number of technologies, including the Internet and web services (Sultan, as cited in Kihoza, 2016). With the Internet and WWW, it’s possible to access a variety of online services from distant locations. For instance, a combination of these technologies makes it possible for developers of applications to build and test their applications online (Nazir, 2012). These developers are thus said to be presented with platforms that serve them on the cloud, namely PaaS.

PaaS is a development environment that allows cloud consumers to develop applications without having to download or install any software in their local machines (Kumar & Goudar, 2012; Nazir, 2012). It is the leveraging of a development language

for creation of applications (Shirazi & Iqbal, 2017). PaaS possesses development infrastructure which is characterized by features like possession of programming environment, tools for development, configuration management, and so forth (Kumar & Goudar, 2012).

Barker (2017) argues that: “if something is an environment in which we build, then it’s a platform”. For this reason, any website-building application is a platform. Since cloud-based website builders are platforms that allow their users to create applications online, they fall in the PaaS category of cloud computing. However, there is a debate on whether these builders should be considered to be in the PaaS service model or not. Lane (2010), for example, hesitates to classify Squarespace (an online website builder) in the PaaS category. On the contrary, Nayak (2016) claims that Squarespace has elements of PaaS development environment. This leads to lack of unanimity on what exactly the PaaS systems are.

2.7 Local Content Development

By “local content”, it means the expression and communication of a community’s locally generated, owned and adapted knowledge and experience that is relevant to the community’s situation and is characterized by location, culture, language and area of interest (URT, 2016; United Nations Educational, Scientific and Cultural Organisation [UNESCO], as cited in Mutula, 2008). Chivhanga (as cited in Mutula, 2008) claims that there has been an increment in local content development efforts among African nations. He maintains that this is a response to a call by the United Nations Economic Commission for Africa (UNECA) to Africans to be involved in the production process.

Although the term “content” refers to things that are contained in something (Duckett, 2005; “Content,” n.d.), it should be noted that, with respect to ISs (like websites), the phrase “local content development efforts” does not only refer to those efforts associated with preparation of things to be contained in ISs, but also the efforts associated with building such ISs. Among other things, such efforts include making

decision on what “service language” to be used in a particular system. In this context, “service language” simply means a language used to deliver service to the intended recipients. This should not be confused with content language, which is the language of the content itself.

Tanzania has been taking a number of initiatives to boost local content development. Such initiatives are witnessed in various sectors, including the Information and Communications Technology (ICT) industry. The Tanzanian NICTP of 2016 recaps the ninth objective of its predecessor (the NICTP of 2003), which focused on promoting local creation and development of ICT applications and multimedia content. This desire still persists in the NICTP 2016, which admits that, despite all the efforts that have been taken in response to the NICTP 2003 objective, the country still depends on foreign importation of content, software and hardware (URT, 2016). However, it’s nice to appreciate little that has been done to contribute to local content development efforts, including the development of the first open source Kiswahili application software named “Jambo Office”, which was developed and launched in Tanzania by the University of Dar es Salaam (UDSM) in collaboration with the Royal Swedish IT Consultancy Company in 2008 (URT, 2016). The need of an office application that majority of Tanzanians would be familiar with is basically what led to the creation of Jambo Office. Familiarity is one of the fundamental features to be taken into account when attempting to have an effective user interface for an application, like a web application. One of the core aspects to be considered to achieve the “familiarity goal” is the use of a language that’s familiar to the intended users of the application (Galitz, 2007; Bhaskar et al., 2011).

In their study, Bhaskar et al. (2011) found that the use of language that is familiar to end-users to be the main contributor to familiarity in most of the websites they were dealing with. For example, for social networking websites, 55% of overall familiarity was found to be contributed by the use of a language known to the user, while 45% was contributed by making the web interface look like real-world situation. For job

websites, 56% of overall familiarity was found to be contributed by the use of a language familiar to the user, whereas 44% was contributed by matching the interface to real-world situation. Similarly, for e-mailing websites, 56% of overall familiarity was contributed by the use of a language familiar to the user, against 44% which was contributed by matching the interface to real-world situation.

2.8 The Web Application Development Procedure

A web application is any application developed using browser-supported languages and that can be accessed over a network using a web browser (Al-Fedaghi, 2011). Web application development entails a lot of issues which need careful handling. These issues may fall into four phases as described in the following sub-sections.

2.8.1 The Definition Phase

Before designing any IS, it is important to clearly define what the system should feature. In project management, this step is referred to as “project scope definition”. The step involves setting boundaries for the project’s size and the range of business functions or processes that will be involved (Brown et al., 2012). When setting the boundaries, project exclusions are defined. These are things or activities that are likely to be thought to be part of a project, but are not (Archbold, 2014). Failure to clearly define what should be incorporated in a system and what should not, means failure to define project scope. Hence, development of an IS, like a website builder, should be preceded by a clearly-defined scope.

2.8.2 The Design Phase

The design step is a very vital component of the procedure for developing any web application. Having a well-designed web application means getting rid of challenges related to poor design, and fulfilling quality of service requirements like speed and scalability of the application (Sridaran et al., 2009). A web application is typically designed to collect and process data. It is meant to enable a user perform some task, and

possibly save something thereafter (Galitz, 2007). Cloud-based website builders usually let their user save their sites after they have been built. As such, they are web applications, and thus should follow principles of good web interface design such as clarity, consistency, directness, predictability, familiarity, forgiveness, recovery, flexibility, simplicity, and positive first impression (Bhaskar et al., 2011; Galitz, 2007). They should also follow principles of good database design like proper definition of data types and constraints, minimizing data redundancy through normalization, having a scalable database structure and optimization of physical performance of a database application (“Basic Design Principles,” n.d.).

2.8.3 The Building Phase

Building an application involves producing computer programs and developing or enhancing databases and files to be used by the system (Brown et al., 2012). Regarding web applications, the building process entails choosing right web development technologies, among other things. Such technologies are constantly changing, making it inevitable for web programmers to pay enough attention when it comes to deciding which technology to use. Alnaqeib et al. (2010) consider Hypertext Markup Language (HTML) to be the key enabling technology for the WWW. Java Server Pages (JSP), Active Server Pages (ASP or ASP.NET) and Hypertext Preprocessor (PHP) are just some of other popular technologies/languages used for web programming (Botwe & Davis, 2015).

Like any other project involving computer programming, web application development projects often experience difficulties, particularly difficulties relating to coding. Web development frameworks were invented to ease the web application development/building task, thereby lessening these difficulties (Botwe & Davis, 2015; Novak & Švogor, 2016; Salas-Zárte et al., 2015). A framework is a solution that facilitate software reuse by allowing for sharing common functions and generic logic of a domain application. Examples of web development frameworks include CakePHP,

Ruby on Rails, Django, and Catalyst (Botwe & Davis, 2015; Salas-Zárate et al., 2015). It is indisputable that web development frameworks simplify the web application development process, but website builders (WCMSs and simple website builders) simplify the process further as they can be used even by those with no computer programming expertise (“Web App Framework,” n.d.).

2.8.4 The Testing Phase

Application testing is an activity aimed at evaluating the quality of a program by identifying defects and problems prior to dealing with them (Salas-Zárate et al., 2015; Ghuman, 2014). It is a very important phase in software development life cycle (SDLC). The process of testing a software in a well-planned and systematic way is referred to as software testing life cycle (STLC). Software/application testing is done to verify that its specified requirements are well met (Dhiman & Sharma, 2016; Girgis et al., 2014; Goel & Gupta, 2014). Among major challenges facing web applications is the defect rate. As cited by Javed, Minhas, Abbas and Riaz (2016), Dogan, Betin-Can, and Garousi reveal that 70% of web applications contain defects. To minimize the number of defects in a web application, it is very important to exhaustively test it before its deployment.

Web application testing involves different kinds of tests such as performance testing which focuses on checking website’s speed, reliability and load handling capacity (Dhiman & Sharma, 2016; Sharmila & Ramadevi, 2014), and security testing which is aimed at maintaining data confidentiality, checking against any information leakage and maintaining proper functionality of an application (Jaiswal et al., 2014). Application testing, such as web application testing is oftentimes done in three levels that precedes the user acceptance testing: the unit level, integration level, and system level. Unit testing occurs at the unit/module level; integration testing is ran when several units are combined to test their interrelatedness; and system testing is done to test the whole system in order to evaluate its compliance with the specified requirements (Kumari, 2014; Goel & Gupta, 2014; Ghuman, 2014). There are three main methods of testing a

software product (white-box testing, black-box testing and grey-box testing) and two main testing techniques (static testing and dynamic testing). White-box testing is regarded to be a detailed investigation as the tester is knowledgeable of the internal working of the code whereas black-box testing is done without having any knowledge of the interior workings of an application. Grey-box testing is done with limited knowledge of the internal workings of an application (Kumari, 2014). As for the testing techniques, static testing is solely an analysis of an application that is done without actually executing the programmed code while dynamic testing involves executing the program or code with sample input, if any, and comparing the resulting output or behavior with expected output or behavior (Ghuman, 2014).

Documentation is very important in software testing. It involves documenting the artifacts that should be developed before or when testing the software. Several tools are available to help accomplishing such a purpose like the test plan, test case and requirement traceability matrix (RTM). A test plan is simply a summary of the test procedure stating the test strategy to be applied, the environment in which the tests will be ran, resources needed in testing the application, limitations of the testing and the timeline of the test activities. Test cases are focused on checking whether an application passes or fails a given test. RTM traces the way requirements are met or not met during system development (“Software Testing,” n.d.).

Among other factors, the dynamic nature of web applications introduces difficulties in testing them (Javed et al., 2016; Girgis et al., 2014). Consequently, programmers often spend much time and effort finding and fixing bugs, especially when they conduct manual tests (Salas-Zárate et al., 2015; Sharmila & Ramadevi, 2014). The website testing process can be simplified by making use of tools built for this purpose. The tools automate the process and free the testing personnel from applying a lot of efforts and spending too much time in testing. These tools include HttpRider, Grinder, NeoLoad and Apache JMeter, to mention a few (Dhiman & Sharma, 2016; Sharmila & Ramadevi, 2014).

Like other web applications, online website building applications need to be tested before they are launched. An exhaustive STLC must take into considerations and decide on the kind of test to be ran, whether to automate the testing process or not and tools to use in case automated testing is the choice, the levels in which the application will be tested, as well as the test method and technique to be applied. Moreover, the overall test procedure has to be documented.

2.9 Related Works

Shah (2012) conducted a study to investigate some of the existing open-source WCMSs. In the study, he also suggests factors to be considered when it comes to deciding whether to build or use an existing WCMS. He eventually builds a WCMS which, among other things, includes a web-page builder. The web-page builder includes an embedded WYSIWYG editor (CKEditor) with which a user can build a page without having to code. Shah's WCMS, however, includes very few built-in templates. This leaves a user with a burden of having to create extra templates, or look for them elsewhere, if he needs numerous templates to select from.

In their study, Luzardo, Avendaño, Chiluita and Verdezoto (n.d.) reviewed several existing WCMSs from which they selected one and suggested how it could be improved. Since the selected WCMS was an open source project, such improvements could easily be done, but as per the scope of their study, they just proposed improvements to be done to make the WCMS capable of allowing users easily create websites that fulfill accessibility standards.

Wojcik (2015) built a personal web platform for academic professionals as his capstone project at Harvard Extension School. Due to its functionalities, he considered the platform to be a balanced amalgam of simple websites builders and WCMSs. The platform was meant to help university students and graduates create web presences by allowing them to build themselves professional, academic websites that would convey their educational/research background, interests, and achievements. All this was meant

to help them as they enter the job market. Although Wojcik's project was a promising one, it was a website-building solution whose targeted community of users was very limited, as it was structured in such a way that made it appropriate only for academic professionals' websites.

There have been many research projects in the area of website builders. Different occasions in different studies have brought about the need for development of online website builders, some being just simple site builders; others being WCMSs. However, among all the study the researcher came across, no study dealt with development of a website builder that is primarily meant for native users of Kiswahili. Thus, this study focused on finding out important features of WYSIWYG web builders and examine their inclusion in Wix, Weebly and Blogger, and thereafter developing a Kiswahili-user-centered cloud-based site building platform that incorporates as many of these features as possible. The language (Kiswahili) is used on the front-end, to help users understand design options/menus available to them.

2.10 Chapter Summary

This chapter has made a review of several materials related to the topic under discussion. An empirical literature review and a review of few works related to this research project alongside the research gap, have been presented. The following chapter discusses a set of methodologies adopted for this research project.

CHAPTER THREE

METHODOLOGY

This chapter discusses both research methodology and system development methodology. Research methodology provided basis for meeting the first and the second research objective. On the other hand, system development methodology provided a clear picture of how the website building platform would be designed, built and tested, thereby meeting the third, fourth, and fifth research objective.

3.1 Research Methodology

Research methodology involves studying the steps the researcher adopts in studying the research problem along with the logic behind these steps (Kothari, 2004). This section describes the methodology adopted for this study in four sub-sections: research design, research materials, data collection methods and experimental setup.

3.1.1 Research Design

Research design serves as a tool dedicated to ensuring that evidence obtained in a given study enables the researcher to address the research problem logically and as unambiguously as possible. There are numerous types of research designs, some of which being case study design which typically narrows down a large field of research into one or a few easily researchable examples; an experimental design which is basically a blueprint of the procedure that enables the researcher to maintain control over all factors that may affect experimental results; and a descriptive design which is aimed at obtaining information concerning the current status of a certain phenomena by providing answers to the questions of “who”, “what”, “when”, “where”, and “how” regarding the research problem, without conclusively ascertaining answers to “why”. More designs include a literature review design which reviews existing literature; an exploratory design which is adopted when there is a limited number of preceding studies (or no preceding study at all) that would act as a reference point in an attempt to predict an outcome; the historical design that deals with evidence from the past in order

to establish facts that defend or refute a given hypothesis; the longitudinal design in which the same sample is tracked over time in order to measure changes; and the observational design which is aimed at making conclusive statements by draws by comparing subjects against a control group, in cases where the researcher has no control over the experiment (“Organizing,” 2018).

The study is based on literature review, experimental investigations and observations. Literature review was employed to gather relevant information in an attempt to answer the first research question. With respect to the second research question, the researcher employed observations and experimental design: some features only needed observation to notice their presence or absence while others required the researcher to conduct simple experiments to be able to tell about their inclusions in a given website builder. The experimental design was selected as it offers a few notable benefits. According to Minton (2013) one of the benefits of the experimental research design is that, at the end of the experiment, the researcher would be able to make the cause-and-effect statements. Although this particular study was not so much interested in the “cause” part of a cause-and-effect statements, it was interested in the “effect” part as all the experiments that were ran focused on eventually being able to tell whether a certain feature was found in a given platform (website builder) or not. Another major benefit connected to this design is that that it allows the researcher to have control over variables. The observational research design was selected as it gave the researcher an opportunity to be able tell whether certain WYSIWYG website builder features are included in the selected website builders by just witnessing (without having to run experiments).

The study employed the principle of replication when running some of the experiments. This principle emphasizes that an experiment should be repeated over and over again so as to increase its statistical accuracy. In this way, experimental results will be more reliable in comparison to the conclusion one would draw without applying the principle (Kothari, 2004). To make the experimental results even more reliable, the principle of

randomization was applied as well. The goal of this principle is to prevent the effect of extraneous variables in an experiment by randomization (Kothari, 2004). Extraneous variables are variables influencing the outcome of an experiment, although they are not the variables that are actually needed in that experiment (“Extraneous and Confounding,” n.d.). Randomization increases reliability of an experiment simply because by randomly picking a group member, bias is eliminated as all members will have an equal chance of being chosen (“Random Sampling,” n.d.). Application of the principle of randomization to avoid undesired effects of extraneous variables normally occurs when such variables are unknown or are uncontrollable. But what should happen when the extraneous variables are known prior to an experiment, and can be controlled? The researcher should simply control them (Stanley, 2012). Hence, the third principle applied in this study was the principle of local control. This principle requires the researcher to control any known source of variation, other than the factors being tested (Stanley, 2012).

3.1.2 Data Collection Methods

Data collection methods used in this study helped to gather requirements for the platform. They are thus requirements collection methods as well. There are direct and indirect data/requirement collection methods. Direct methods involve meeting (or calling, via telephone) the prospective user of an IS and hearing his comments in person and firsthand. On the other hand, indirect methods involve an intermediary between the person who collects data and the prospective user of the system. Individual face-to-face interviews, observational field study, telephone interviews, traditional focus groups and facilitated team workshops are just some of the direct methods. Indirect ones include the use of a Management Information Services (MIS) intermediary, paper survey or questionnaire, electronic survey or questionnaire, document review, literature review, experimental observations, electronic focus group, as well as e-mail, bulletin boards, or guest book (Galitz, 2007).

Out of all these methods, only two were selected: literature review and experimental observations. These were selected with some justification: literature review was selected as it facilitates an in-depth understanding of the topic, whereas experimental observations (experimentation and observations) method was selected due to the fact that by running the experiments (and observe by himself), the researcher will be able to certainly discuss about the inclusion of website builder features in the selected platforms.

3.1.2.1 Literature Review

Literature Review was employed in an attempt to find an answer to the first research question. This section presents information regarding the search strings used when searching for literature sources, the databases from which these sources were obtained, data selection, summary of reviewed literature, and the data analysis/synthesis methodology.

3.1.2.1.1 Search Strings

The researcher surfed the internet using different search strings to retrieve sources relating to the first research question. On each “search attempt”, the researcher used different keywords so as to retrieve appropriate information to the largest possible extent. The keywords/search strings used are: ("essential") or ("necessary") or ("important") and (“features”) or (“characteristics”) and (“WYSIWYG website builders”) or (“website builders”) or (“site builders”) or (“website building applications”) or (“web authoring software”).

3.1.2.1.2 Databases

The data was collected from different databases: Science Direct, IEEE Explore, ACM Digital Library and Springer-Link. The researcher also used a few internet sources (web pages) whenever they were found to contain information that is more relevant to the

issue being addressed, as compared to information contained in official publications the researcher came across.

3.1.2.1.3 Data Selection

The researcher had to set some selection criteria to ensure only the most suitable sources of data are used. Decision as to whether a certain source should be included or not was based on criteria presented in Table 2 and Table 3.

Table 2. Inclusion Criteria

Inclusion Criteria	
IC1	The paper is available in full text
IC2	The paper is written in English language
IC3	The paper was published within ten years prior to the date of conducting the study (i.e. not earlier than 2008).
IC4	The title or abstract of the paper discusses about website builders
IC5	A blog post or web page related to the topic provided that no official publication addressing the same issue was obtained
IC6	A white paper from a reputable author

Table 3. Exclusion Criteria

Exclusion Criteria	
EC1	The paper is a duplicate of an already included paper.
EC2	Studies that are only available as abstracts or PowerPoint presentations
EC3	A paper with no citations or reference list
EC4	Book chapters

3.1.2.1.4 Summary of Reviewed Literature

Table 4 Summarizes the reviewed literature specifying the title of the source, its type, year of publication, place of publication (or name of the journal, if the source is a journal paper) and reason for selection of that particular source.

Table 4. Summary of Reviewed Literature

Title	Type	Year of Publication	Place of Publication/ Journal	Reason for Selection
5 Website Builder Features Your Small Business Needs.	Blog post	2017	N/A	The source has more relevant information for answering the first research question as compared to most official publications the researcher came across.
What is Website Builder Software: Analysis of Features, Benefits and Pricing.	Web page	2018	N/A	The source has more relevant information for answering the first research question as compared to most official publications the researcher came across.
Lesson 1: Basic Features of Web Authoring Software.	Web page	n.d.	N/A	The source has more relevant information for answering the first research question as compared to most official publications the researcher came across.
Features of Web Authoring Software	Blog post	2016	N/A	The source has more relevant information for answering the first research question as compared to most official publications the researcher came across.
Searching for the Perfect Website Builder? Here's What Matters	Web page	2016	N/A	The source has more relevant information for answering the first research question as

				compared to most official publications the researcher came across.
Creating an E-Commerce Web Site: A Do-It-Yourself Guide	White paper	2011	Mountain View, CA	The source explains further about WYSIWYG editors (which is one of the essential features of website builders).
WordPress vs. Joomla: Which is the Right CMS for you?	White paper	n.d.	Austin, TX	The source explains further about WYSIWYG editors (which is one of the essential features of website builders).
Website builders: features you should look out for.	Web page	2015	N/A	The source has more relevant information for answering the first research question as compared to most papers the researcher came across.
Responsive website design for higher education utilizing mobile centric features.	Journal article/paper	2014	Online Journal of Applied Knowledge Management	The source explains further about Responsive Layouts (which is one of the essential features of website builders).
Designing Responsive Websites Using HTML and CSS.	Journal article/paper	2014	International Journal of Scientific & Technology Research	The source explains further about Responsive Layouts (which is one of the essential features of website builders).
Responsive Website: A Transformation in Web Designing.	Journal article/paper	2015	International Journal of Engineering Technology, Management	The source explains further about Responsive Layouts (which is one of the essential features of

			and Applied Sciences	website builders).
Analysis of User Experience Quality on Responsive Web Design from its Informative Perspective.	Journal article/paper	2014	International Journal of Software Engineering and Its Applications	The source explains further about Responsive Layouts (which is one of the essential features of website builders).
Google's mobile-first indexing is finally rolling out: Here's what you need to know	Blog post	2018	N/A	The source explains about mobile-first indexing, which is closely related to Responsive Layouts (which is one of the essential features of website builders).
What you Need to Know about Google's Mobile-First Indexing Rollout	Blog post	2018	N/A	The source explains about mobile-first indexing, which is closely related to Responsive Layouts (which is one of the essential features of website builders).
How Does Mobile-First Indexing Work, and How Does It Impact SEO?	Blog post	2018	N/A	The source explains about mobile-first indexing, which is closely related to Responsive Layouts (which is one of the essential features of website builders).
What's the Best Website Builder For SEO?	Web page	2018	N/A	The source has more relevant information for answering the first research question as compared to most papers the researcher came across (particularly on SEO).

The Foremost Guidelines for Achieving Higher Ranking in Search Results through Search Engine Optimization.	Journal article/paper	2013	International Journal of Advanced Science and Technology	The source explains further about SEO (which is one of the essential features of website builders).
The Impact of Search Engine Optimization Dimensions on Companies Using Online Advertisement in Jordan.	Journal article/paper	2015	American Journal of Business and Management	The source explains further about SEO (which is one of the essential features of website builders).
Open Source Content Management Systems: A Canvass.	Journal article/paper	2012	International Journal of Multidisciplinary Sciences and Engineering	The source lists a number of features of CMSs.
Analytical Study of Domain Name System, its disputes and legal issues.	Conference paper	2013	N/A	The source explains further about domain names/custom domain names (which is one of the essential features of website builders).
Review of Domain Name System	Conference paper	2011	N/A	The source explains further about domain names/custom domain names (which is one of the essential features of website builders).
Personal Learning Journal – Course Design for Using Weblogs in Higher Education.	Journal article/paper	2008	The Electronic Journal of e-Learning	The source explains further about blogging (which is one of the essential features of website builders).
A Comparative Study of Content	Journal article/paper	2015	International Journal of	The source compares three WCMSs:

Management Systems: Joomla, Drupal and WordPress.			Advance Research in Science and Engineering	Joomla, Drupal and WordPress, and discusses about their features/capabilities.
Using Domain Ontology to Implement a Frequently Asked Questions system.	Conference paper	2009	N/A	The source explains further about FAQs (which is one of the essential features of website builders).

3.1.2.1.5 Data Synthesis

In order to draw conclusions regarding the first research questions, data synthesis had to take place. During the process, the researcher identified commonly mentioned website builder features and prepared brief description for each feature.

3.1.2.2 Experimentation and Observations

The second research objective, and consequently the second research question, required running a number of experiments. The experiments eventually equipped the researcher with evidence that made him able to tell whether certain features were found in a given cloud-based website builder, or not. To conduct the experiments, the researcher needed certain tools and devices, and had to prepare appropriate environment for running the experiments.

3.1.2.2.1 Tools and Devices

The study needed a variety of tools/devices to be accomplished. A laptop computer was a very basic tool for running the experiments. Certain internet connection devices, such as modems, were needed to connect this laptop to the Internet as the experiments involved online website building platforms. Computing devices of different screen resolutions were used to test responsiveness of web pages that were built by the website builders. These included computer monitors/screens of varying screen resolutions, mobile phones and tablets.

3.1.2.2.2 Experimental Setup

To conduct the experiments, the researcher used a laptop computer, a desktop computer monitor, a tablet and two mobile phones. The desktop computer monitor used was not connected to any system unit. Hence, to make use of it, the researcher used a Video Graphics Array (VGA) cable to extend laptop display onto this monitor. This was primarily meant to offer an extra screen for testing web-page responsiveness. A modern web browser (Google Chrome version 66.0.3359.139) was used to access the website builders. Further, the researcher created an account in each of the host-websites of the aforementioned website builders (Wix, Weebly and Blogger), so as to be able to run the experiments.

3.2 System Development Methodology

The system development methodology guides system development projects from the analysis to system deployment phase. This section summarizes methodology adopted in developing the platform.

3.2.1 Systems Development Life Cycle

IS projects typically follow systems development life cycle (SDLC) steps. The abbreviation (SDLC) could also be considered to be referring to software development life cycle, when a system being dealt with is a software. An SDLC is a stepwise technique employed in software projects for systematic design, development and maintenance of a software while ensuring all user requirements are fulfilled with least amount of resource consumption (Lehman & Sharma, as cited in Barjtya et al., 2017; Sharma, 2017). There exist so many variations of SDLC representation, one of which being Figure 2. The figure represents a total of six development phases in a circular shape. The very first phase is a planning phase, whereby, among other things, system feasibility is examined. Feasibility analysis involves an assessment of system's economic, operational and technical feasibility. In the following phase, system requirements are identified and documented in a Software/System Requirement Specification (SRS) document, which serves as a reference for designers to come out

with a suitable product design. The building phase succeeds the system design phase. It is in this phase where system development actually occurs. The testing phase then follows, after which system deployment is to take place (Brown et al., 2012; “SDLC - Overview,” n.d.).

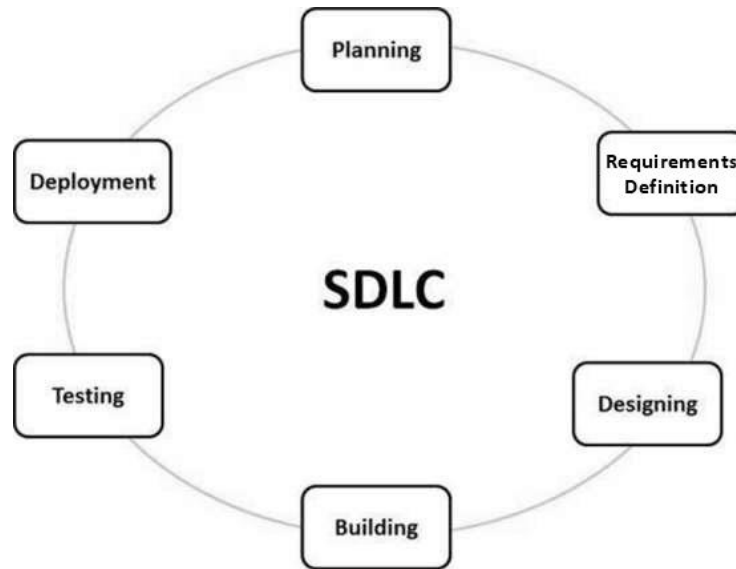


Figure 2: SDLC Phases (SDLC – Overview, n.d.)

The SDLC is a generic guide for system development projects. It has been a basis for formulation of a number of specific models, including the waterfall model, the prototyping model, the incremental model, the V-shaped model, the spiral model, rapid application development (RAD) and a set of agile methodologies (Barjtya et al., 2017; Brown et al., 2012; Mishra & Dubey, 2013; Sami, 2012). This project employed the incremental model. The model allows development of a system in small chunks, which are updated in each iteration to accommodate additional functionalities (Barjtya et al., 2017). Thus, the reason for this model to be named “incremental” is the fact that it entails repetitively “enlarging” a system until all requirements are well met. Such repetition makes this model one of the iterative models. The choice of this model over all the other models is backed up by the fact that it allows the start of a development process even if requirements have not been fully defined (i.e. some requirements will be

defined and accommodated in future iterations), a working version of the system can be experienced early in the development process, testing is easier when a system is still small (when iterations are still few), and the fact that it allows developers to take advantage of what was learned during the development of earlier versions/iterations of the application so as to well-build the forthcoming versions (Barjtya et al., 2017; Mishra & Dubey, 2013; Sami, 2012). Figure 3 is a simple representation of this model.

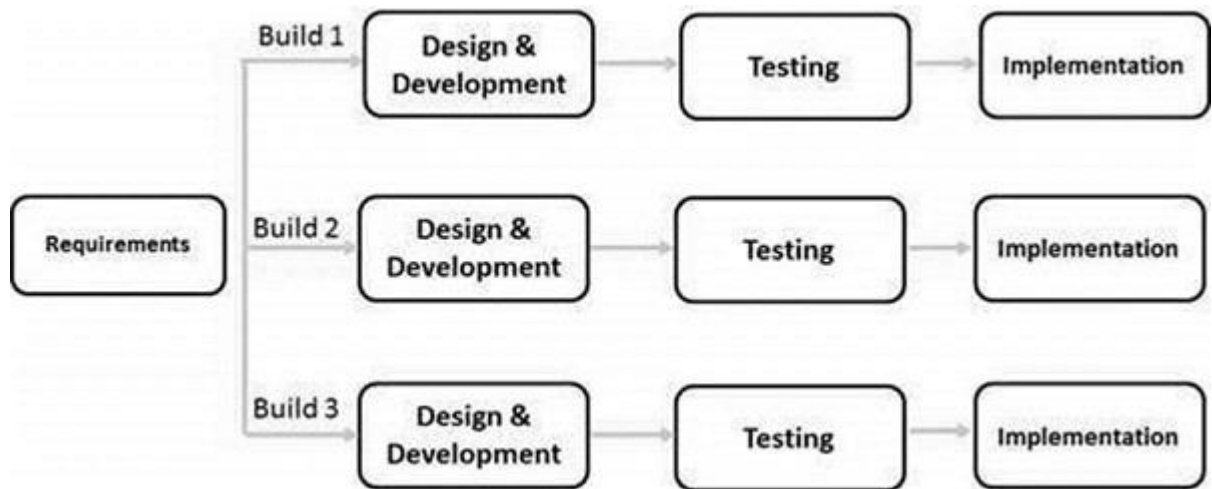


Figure 3: The SDLC Incremental Model (SDLC - Iterative Model, n.d.)

3.2.2 Systems Analysis and Design Approach

This project followed the object-oriented systems analysis and design (OOSAD) approach. This approach enables decomposition of problems on objects that contain both data and processes. It enables system analysts to break a large system into smaller components. These components are often referred to as modules. The analyst will then have to work on the modules individually, before piecing them back together to form a large, complete system (Dennis et al., 2009). Strengths of this approach justify its use in this project. The strengths stem from its level of modularity, something that's not well handled in traditional approaches like structured design. The strengths include the fact that it makes system development easier to grasp, it facilitates creation of reusable

components (which might be of use in future projects), and it leads to production of systems that are easy to debug (Dennis et al., 2009; “Importance of Modularity,” 2018).

3.2.3 Design Tools

Prior to the development stage, software projects usually involve system modeling using a variety of diagrams. For professional modeling, this project made use of the Unified Modeling Language (UML). The reason behind selection of this modeling tool is that it is a powerful and flexible language for analysts and developers (Dennis et al., 2009), that is universally accepted. The UML is “a standard set of diagramming techniques that provide a graphical representation rich enough to model any systems development project from analysis through implementation” (Dennis et al., 2009, p. 34). Further, entity relationship diagrams (ERDs) were used for data modeling.

Entity relationship (ER) and UML diagramming is convenient if appropriate software tools meant for that purpose are used. This project made use of Astah Professional software to draw such diagrams. In addition, Microsoft Office Word and Ms Paint played part in portraying the appearance of web-pages (preparation of wireframes).

3.2.4 Development Technologies and Tools

Website development projects require utilization of a number of technologies associated with web programming. PHP, My Structured Query Language (MySQL), HTML, Cascading Style Sheets (CSS) and JavaScript (and JQuery) are technologies used in this particular project. PHP and MySQL served as server-side languages. PHP served as a scripting language to manage dynamic content that interacts with database and files while MySQL acted as the database management system (DBMS). HTML, CSS and JavaScript were used on the client side, whereby HTML was used to define content of web pages, CSS to specify web page layout and JavaScript to program the behavior of web pages. To ease the development work at the front-end, Bootstrap was used as a framework.

HTML, CSS and JavaScript were selected for the front-end as they are very popular technologies, and hence a lot of documentation and forum discussions about them exist. Extensive documentation and forum discussions also fueled the selection of PHP, in addition to the fact that the language (PHP) is platform-independent which makes it advantageous over languages such as ASP.net, which is built to run only on Windows platform. Further, the researcher is experienced in PHP than any other language that would play a similar role. The selection of MySQL has also been influenced by researcher's level of skills in the DBMS and the fact that it has bigger community of users (hence much forum discussion) and extensive documentation just like the aforementioned technologies. The same justification stands for the framework selected for this project, Bootstrap.

A few software tools were needed in development efforts: a text editor, a web server and a database server. Notepad++ played the text-editing role whereas XAMPP was used as web server. Notepad++ was selected because of its ease of use and familiarity to the researcher/developer. XAMPP was selected as it is one of the most-popular freely available server software, that offers both a web server (Apache) and a database server (MySQL) in the same environment/package. Additionally, Google chrome web browser was used to check appearance and functionalities of the platform, as it is being built.

3.2.5 The Testing Methodology

An exhaustive STLC must take into considerations and decide on the kinds of test to be ran, whether to automate the testing process or not and tools to use in case automated testing is the choice, the levels in which the application will be tested, as well as the test method and technique to be applied. Moreover, the overall test procedure should be documented.

Tests that were ran in this project are mainly connected to functional requirements of the application. The tests were ran at three levels: the unit level (unit testing), the “collection of units” level (integration testing), and the system level (system testing). A

testing method that was used extensively is white-box testing, as the tester was the programmer himself. However, some functionalities of the platform were implemented with the help of ready-made libraries. Hence, black box-testing was employed in testing the working of these libraries, and grey-box testing was used whenever the developer had only a limited knowledge about how the libraries have been built, and the code involved therein. Dynamic testing technique was used in this project as the tests involved running/executing the code. Ultimately, the test results were documented using test cases.

3.3 Chapter Summary

This chapter has presented a description of methods used to conduct the study and to develop the system. Literature review, experimental investigations and observations have been described as research designs adopted in this research project. The chapter has also mentioned a set of materials used in the study. Moreover, a brief description regarding how data relevant to this study was collected has been given. The environment in which the experiments were ran has also been described. A description of system development methodology has been given to state the SDLC model selected (the incremental model), the systems analysis and design approach, design tools, development technologies and tools as well as system testing methodology. Table 5 summarizes methodologies adopted in meeting each of the five objectives.

Table 5. Research and Development Methodology Summary

s/n	Specific Objective	Methodology
1	To find out important features of WYSIWYG site builders.	Literature review
2	To examine the inclusion of these features in three selected cloud-based site building platforms: Wix, Weebly and Blogger.	Experimentation and observations

3	To design a Kiswahili-user-centered cloud-based site building platform that incorporates as many of the features included in the examined platforms as possible.	SDLC Incremental Model, OOSAD, ERDs, UML, UML software tool (Astah Professional), Microsoft Office Word, Microsoft Paint.
4	To build the designed platform.	PHP, MySQL, HTML, CSS, JavaScript/JQuery, Bootstrap framework, Notepad++ editor, XAMPP server (for Apache web server and MySQL server).
5	To test the platform.	STLC, Google Chrome web browser, unit testing, integration testing and system testing, white-box testing, black box-testing, grey-box testing, dynamic testing technique, test cases.

The next chapter reports findings of the study, thereby meeting the first two research objectives.

CHAPTER FOUR

RESEARCH FINDINGS AND DISCUSSION

This chapter is aimed at presenting answers to the first and the second research question. The first research question sought to discover features that are perceived to be important in any WYSIWYG website building platform. The second one was aimed at examining if such features are included in the named online website builders. Three principles of experimental designs were applied during such examination: the principle of replication, the principle of randomization and the principle of local control.

4.1 Features of WYSIWYG Website Builders

To identify features attributed to WYSIWYG website builders, the researcher had to conduct a thorough literature review across a number of sources. In general, most features were found to be recurring across the sources. This section presents the features the researcher found in the literature.

Features listed hereunder are mostly functional requirements of WYSIWYG website builders. Some sources list both non-functional requirements (such as simplicity, as mentioned by Johansson [2016]) and functional requirements as essential features of website builders. Most of the non-functional requirements, however, are very difficult to measure. As such, this study will deal with the functional requirements (as they can be examined with ease), and only a few non-functional ones, that can practically be measured/examined (such as Search Engine Optimization (SEO) and responsiveness). Non-functional requirements are closely related to application's external quality attributes. Such attributes relate to how developers and users perceive an application, and thus, cannot be measured objectively. Instead, they can be weighed by measuring internal attributes of an application, which are usually measurable. For example, to figure out how usable an application is, one would need to measure several internal attributes of that application, such as number of error messages generated by the application and length of user manual, as shown in Figure 4 (Sommerville, 2011).

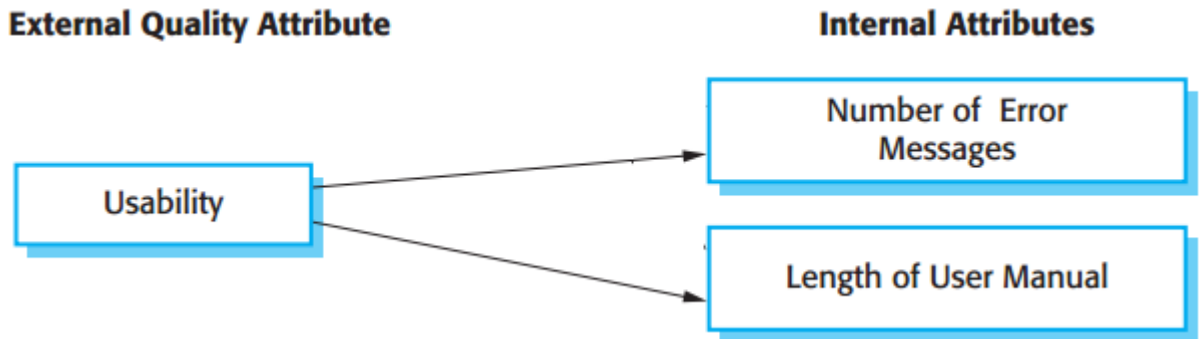


Figure 4: Relationship between an external quality attribute and internal attributes of an application (Sommerville, 2011)

4.1.1 Built-in Templates

One of the very basic things a website builder should be equipped with is a collection of ready-made templates for a user to customize according to his needs. Despite the fact that most website builders provide an option to start with a blank page, in addition to an option to edit a ready-made template, the latter option makes more sense because the builders are primarily meant to simplify the website-building task (Vissers, 2017; “What is Website,” 2018).

4.1.2 WYSIWYG Editor

To achieve ease of use, website builders include WYSIWYG editors. These tools make the content creation and editing task easy and quick. With WYSIWYG editors, even users with no coding skills can create and edit web content (Baldaniya & Baldaniya, 2014). For example, instead of writing HTML code, a user is able to assign a heading tag to text by simply picking a respective option from a drop-down list of headings, and the rest (the code) is handled in the background (“Lesson 1: Basic Features,” n.d.; Webb, 2016). Just like word processing programs, WYSIWYG editors are perceived to be very easy to use (GeoTrust, 2011). Some would even argue that some of these editors are easier to use than the word processors, such as Microsoft Word (Schäferhoff, n.d.). Some website builders’ WYSIWYG editors are so advanced that they allow arrangement of items using “drag-and-drop” tools (GeoTrust, 2011).

4.1.3 Code Editor

Most website builders provide an option to allow a user to switch between the coding and WYSIWYG interface, to provide him with flexibility of being able to edit the code whenever he needs to do so. Some builders present the user with one more view, the split view. With this view, the WYSIWYG and the code interfaces are displayed simultaneously in separate windows (“Lesson 1: Basic Features,” n.d.).

4.1.4 Responsive Layouts

The number of visitors visiting websites with handheld devices is drastically increasing worldwide. To provide them with optimal viewing experience, websites need to be responsive (Klein & Gubic, 2014; Yadav & Barwal, 2014; Singh et al., 2015). Responsive web design refers to a set of techniques employed by web designers in order to make a website adapt itself to any device or width of screen. This means that website’s appearance will change to well-fit screen size of the device from which it is accessed (Lestari et al., 2014; Singh et al., 2015). A website builder should be able to produce responsive layouts (“Website builders,” 2015). This will make websites produced with these builders friendly to mobile devices (which are mainly smartphones and tablets), which, according to Singh et al. (2015), are nowadays more widely used than personal computers (PCs).

These days, mobile-friendliness has something to do with SEO. This is due to Google’s recent decision to roll out mobile-first indexing. This new way of indexing began in March, 2018. By “mobile-first indexing”, it means that a mobile version of a website becomes the starting point for what Google search engine includes in its index, and the baseline for how the engine determines the website’s rankings (Robles, 2018; Cosper, 2018; Randolph, 2018; Benjamins, 2018). That said, the use of a website builder that’s capable of generating responsive layouts/templates will not only provide assurance of having a good look of web pages in different devices, but also boost website ranking in search results.

4.1.5 SEO Tools

SEO relates to a set of techniques that are aimed at enhancing the probability of website access by optimizing web-pages to make them friendlier to acquire higher ranking in search results (Rehman & Khan, 2013). Proper SEO management increases the possibility of a web application to be popular (Farooq et al., 2012). It's a wish of most owners of websites and/or webmasters to see their sites outrank those of their competitors in search results. For instance, in the marketing industry, SEO is widely used as a strategy for raising the volume of customer traffic to a company's website (Khraim, 2015). It's important for a website building platform to be equipped with SEO-friendly web design technologies in order for a user to be confident that his site will always rank higher in search results (Johansson, 2016). Benjamins (2018) insists that, for search-engine optimization, a website builder should be characterized by several features, including: allowing for a user to customize meta titles and descriptions (see Figure 5), offering Secure Sockets Layer (SSL) certificates, and possession of mobile-friendly themes (as described in the aforementioned feature).

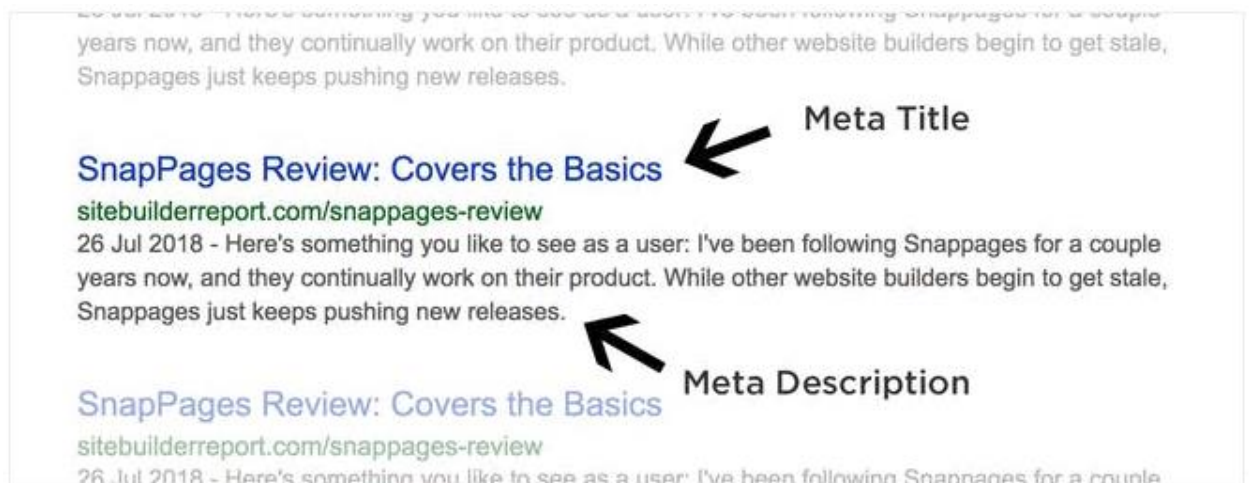


Figure 5: Sample Meta Title and Meta Description (Benjamins, 2018)

4.1.6 Custom Domain Name

A domain name is simply a name used to identify a particular website on the Internet (Kadam, 2013). Out of a single domain, there can be formed several sub-domains. A name of a domain that's a sub-domain of another domain ends in the other domain's name (Singh & Maini, 2011). Normally, website builders (online ones) offer their users with free sub-domains to uniquely identify their sites. It's nice for a website builder to allow for purchasing a site's custom domain name (i.e. a domain name that's not part of the website builder's domain) directly from itself. All the popular web-based website builders have this option (Visser, 2017).

4.1.7 Social Media Integration

Most website builders possess this feature. The feature makes it possible for a website to be integrated with social networking sites like Instagram, Facebook and Twitter. Such integration allows a website to perform certain actions, such as pulling in the social network feeds and sharing website's content to the visitor's social network ("Website builders," 2015; "What is Website," 2018).

4.1.8 Form Builder

It is common practice to include contact forms on websites. These, plus other kinds of forms, help grabbing information from website visitors. A good website builder should feature form-building tools. Wix's 123 Form Builder, for example, offers several types of forms: contact form, order form, sales lead form and many other types (Visser, 2017; "What is Website," 2018; "Website builders," 2015).

4.1.9 Graphical User Interface (GUI) for Adding a Map

Nice website builders enable their users to add maps, such as Google maps, to their sites by using graphical tools (without having to directly deal with codes). A user may be presented with a default map which he should be able to change by simply putting his address or location information ("Website builders," 2015).

4.1.10 Blogging

A blog (also known as a weblog) is an information-sharing platform that enables people to record their thoughts in diary form and publish those diaries as web pages without having to code (Hain & Back, 2008). Some website builders, like Blogger, are specially designed for blogging. Even the initial release of WordPress was meant for blogging (Sheokand & Vinay, 2015). While not all website builders existing in today's market feature the blogging component ("What is Website," 2018), most of them do.

4.1.11 Web Analytics Tools

It is important for a webmaster or owner of a website to be knowledgeable about how his site is being visited. Web analytics tools help in tracking web visits. With these tools, it is possible to measure the amount of traffic a website receives and the amount of time spent by visitors on each page. The tools also help to know the most-visited pages of a website, and locations the traffic comes from. Google Analytics is one of the most-used web analytics tools. Website builders normally allow integration with web analytics solutions, like Google Analytics, or provide their own solutions (Visser, 2017; "What is Website," 2018).

4.1.12 Customer Support

A user needs to be assured of support whenever problems arise or when he needs clarification on how to go about performing a certain action, when using a given website builder. Customer support can be provided using different means of communication, including live chat (Johansson, 2016). A website builder should therefore present information describing how a customer can get help whenever the need arises. One of the most-commonly used means of providing help to users of web services, including users of website builders, is the inclusion of a Frequently Asked Questions (FAQ) section or knowledge base that contain information that would be of use to the user (Hsu et al., 2009).

4.1.13 Video and Image Galleries

Depending on the kind of website, images and videos may be very essential in a website. Many of today's websites include at least one of the two. A website builder should provide different image manipulation options such as image resize and crop. It's also important for a site builder to allow the organization of images and videos in galleries. Some website builders can work with video sites such as YouTube and Vimeo, hence a user's website can pull videos therefrom ("What is Website," 2018; "Website builders," 2015).

4.1.14 Preview Mode

With this mode, the user is able to view a built web-page in a web browser (Webb, 2016). Thus, this feature can be used to witness if WYS (what you see) in a particular website builder is really WYG (what you get).

4.2 Examination of the Identified Features in Wix, Weebly and Blogger

The second research question was aimed at examining if the aforementioned features are included in the three selected cloud-based website builders. This was accomplished by running a series of simple experiments whose results are summarized in Table 6. The three basic principles of experimental designs (principle of replication, principle of randomization and principle of local control) were applied in these experiments.

4.2.1 Application of Experimental Design Principles

4.2.1.1 The Principle of Replication

The principle of replication was applied when checking responsiveness of templates, when testing SEO capabilities of the website builders, and when checking the map-addition GUIs of the builders. This principle requires that, for reliability purpose, an experiment should be repeated more than once. When testing template responsiveness in each of the three website builders, the exercise was not only repeated several times, but also repeated on different screen resolutions. Without applying the principle of replication, it would have been adequate to test responsiveness of a given template

using as few as two devices of varying screen resolutions. But since the principle is concerned with running experiments repeatedly, the researcher used five devices of different screen resolutions to observe how the templates would behave on each device. For each website builder, a built website was viewed on a 1366 x 768 resolution laptop (1366 pixels screen width and 768 pixels screen height), a 1024 x 768 resolution monitor (1024 pixels screen width and 768 pixels screen height), a 320 x 570 resolution mobile phone (320 pixels screen width and 570 pixels screen height), a 320 x 534 resolution mobile phone (320 pixels screen width and 534 pixels screen height), and a 600 x 1024 resolution tablet (600 pixels screen width and 1024 pixels screen height).

When testing the website builders' SEO tools, the principle of replication was applied when the researcher conducted SEO tests on different website pages; that is to say, in each website, more than one web-page was tested for SEO performance. Further, the principle of replication was applied when the researcher was testing functionality of GUIs used for adding maps to web-pages. For each GUI tested, the researcher performed a few trials, each time trying to add a different Earth location, so as to be satisfied that any (or almost any) given point on Earth can be shown on a map, on a web-page.

4.2.1.2 The Principle of Randomization

This principle was applied when picking templates to work with, in each website builder. If, for instance, the researcher decided to deal with only the templates appearing first in the template-selection page (no randomization), then he would have left a room for extraneous factor(s), if any, to affect the experiments conducted. A reason for the presence of extraneous factor(s) in this case would be, for example, the best templates would have been located in such a way that they appear first (in the template-selection page) and the low-quality ones being located some place that need some browsing efforts to reach. By randomly picking a template, all the templates had an equal chance of being selected.

4.2.1.3 The Principle of Local Control

The principle of local control was applied when testing the website builders' SEO capabilities. In the SEO arena, it generally takes a while for SEO to work. Knowing this, the researcher had to wait for some time before testing the impact of SEO tools, after using such tools to optimize web-pages for search engines. On Blogger, for instance, the effect of SEO was experienced after a few hours; the SEO efforts wouldn't show an instant impact on Google search results.

4.2.2 Results of the Experiments

Table 6 presents results obtained when checking inclusion of the identified website builder features in Wix, Weebly and Blogger. In the table, a check mark (✓) indicates that the feature under study was found in a given builder; a cross (✗) indicates a missing feature; and a cross and check mark with an asterisk (✗* and ✓*) indicate that there is further discussion on the feature, on the result discussion part that follows the table.

Table 6. Inclusion of Website Builder Features in Wix, Weebly and Blogger

		Wix	Weebly	Blogger
Features	Built-in Templates	✓	✓	✓
	WYSIWYG Editor	✓	✓	✓
	Code Editor	✓	✓	✓
	Responsive Layouts	✓	✓	✓
	SEO Tools	✓	✓	✓
	Custom Domain Name	✓	✓	✓
	Social Media Integration	✓	✓	✓
	Form Builder	✓	✓	✗
	GUI for Adding a Map (Google Map)	✓	✓	✗ *
	Blogging	✓	✓	✓
	Web Analytics Tools	✓	✓	✓
	Customer Support	Live Chat		✗ *
		Forum		✓
		FAQ Section/ Knowledge Base		✓
		Video Tutorials		✗ *
	Video and/or Image Gallery		✓	✗ *
	Preview Mode		✓ *	✓

4.2.3 Discussion on the Results

4.2.3.1 GUI for Adding a Map (Google Map)

While adding a Google Map to a Wix and Weebly site is done right on the platform, doing the same on Blogger requires one to navigate to Google Maps site and copy the “embed code” therefrom. The user will then have to paste this code to Blogger’s HTML editor. Blogger has no GUI for adding a map.

4.2.3.2 Live Chat

Live Chat service was not found in Wix’s free plan (If Wix has this service, it may be featured in premium plans). Similarly, the service was not found on Weebly, nor Blogger. Weebly’s help center has a section entitled “chat with us”, but it is not a “live” chat section because when a user makes an inquiry on this section, he has to wait for some time before the feedback is delivered to him, through his e-mail address.

4.2.3.3 Video Tutorials

There may be numerous video tutorials on usage of Blogger, but the tutorials are not presented on the platform. They may be available elsewhere (for example, on sites like YouTube).

4.2.3.4 Video and Image Galleries

Although Blogger has several pictorial templates, it has no tools specially designed to build and manage video and image galleries. This is different from Wix and Weebly, which feature gallery management applications.

4.2.3.5 Preview Mode

Although Weebly has a preview interface for mobile appearance of a site, it has no an explicit option to preview the site’s desktop appearance. Weebly lets a user preview the desktop appearance of a site on the editor (simply because the editor is WYSIWYG). But this is not an “exact” preview mode as what’s being displayed in the WYSIWYG editor is not only the site, but also the site-editing tools. This makes it necessary for a

user to “manually” navigate to a new tab, or another web browser, and type in his site’s Uniform Resource Locator (URL), to see how it would appear to its visitors. Otherwise, the user has to click the publish button to be able to see a window with a clickable link, which, upon being clicked, will open the built website in a new browser tab.

4.3 Functionalities of the Platform

The major goal of investigating features of WYSIWYG website builders and examining their inclusion in the three selected cloud-based website builders was to build a basis for setting functional requirements of the cloud-based website builder. The target here was to check the possibility of implementing all the features found in the three platforms. However, as stated earlier in this document, if the set of features discovered cannot be implemented in its entirety, then the researcher had to select only those features whose implementation would be feasible with the available time and skill-set. That said, features selected for this project were: inclusion of built-in templates, a WYSIWYG editor, responsive layouts, simple SEO tools, customer support (knowledge base), image galleries and a site preview mode.

Thus, functional requirements of the platform are:

- i. Presenting a set of built-in templates.
- ii. Inclusion of a WYSIWYG editor.
- iii. Inclusion of a code editor (an HTML code editor).
- iv. Provision of an interface for a user to specify meta descriptions and keywords for SEO.
- v. Presentation of a knowledge base for user support.
- vi. Provision of tools for managing/customizing web page image galleries.
- vii. Inclusion of a site preview mode.

4.4 Chapter Summary

This chapter was aimed at answering the first and the second research question. In responding to the first research question, several features of WYSIWYG website builders have been discussed, as found in the literature. This has been followed by a presentation of results of the experiments whose target was to examine inclusion of the features in Wix, Weebly and Blogger, in finding answers to the second research question. All this was meant to build a basis for formulation of functional requirements for the platform. Subsequent chapters show how the platform was designed, built, and tested.

CHAPTER FIVE

SYSTEM DESIGN

This chapter presents design of the platform targeting to reflect the functional requirements specified in the preceding chapter. Methodology adopted for designing the platform incorporates the OOSAD approach, ERDs, UML and software tools for UML diagramming and preparation of wireframes.

5.1 Analysis Modeling

With analysis modeling, a system analyst focuses on portraying who will be the user of a given system, what functions will the system perform, where the system will be used as well as when is it going to be used. The major goal here is to learn about the system. Analysis modeling is achieved by performing three basic types of modeling: functional modeling, structural modeling and behavioral modeling (Dennis et al., 2009).

5.1.1 Functional Modeling

Functional modeling is targeted at producing the so-called functional models, like activity diagrams and use-case descriptions and diagrams. Such models describe business processes and the interaction of an information system with its environment. In other words, they describe the functionality of an information system. In this project, the researcher used activity and use-case diagrams for functional modeling.

5.1.1.1 Use-case Diagram

Figure 6 shows the use-case diagram drawn for the platform.

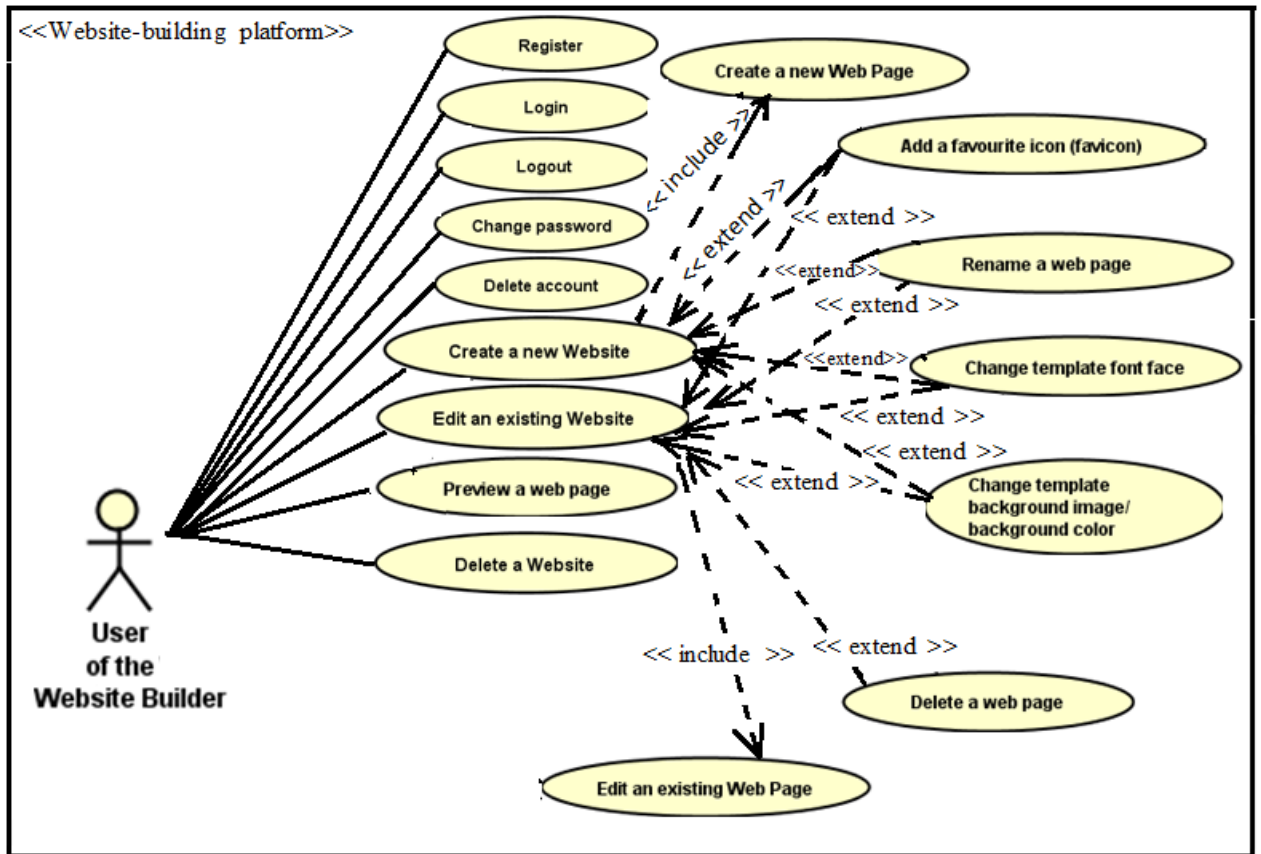


Figure 6: Use Case Diagram for the Website-building Platform

5.1.1.2 Activity Diagram

Activity diagrams are used to depict system's flow of action just like flowcharts and data flow diagrams. The selection of activity diagrams was influenced by its strength over data flow diagrams: activity diagrams are more appropriate tool when it comes to portraying concurrent activities and complex decision processes (Dennis et al., 2009). Following is the activity diagram for the platform.

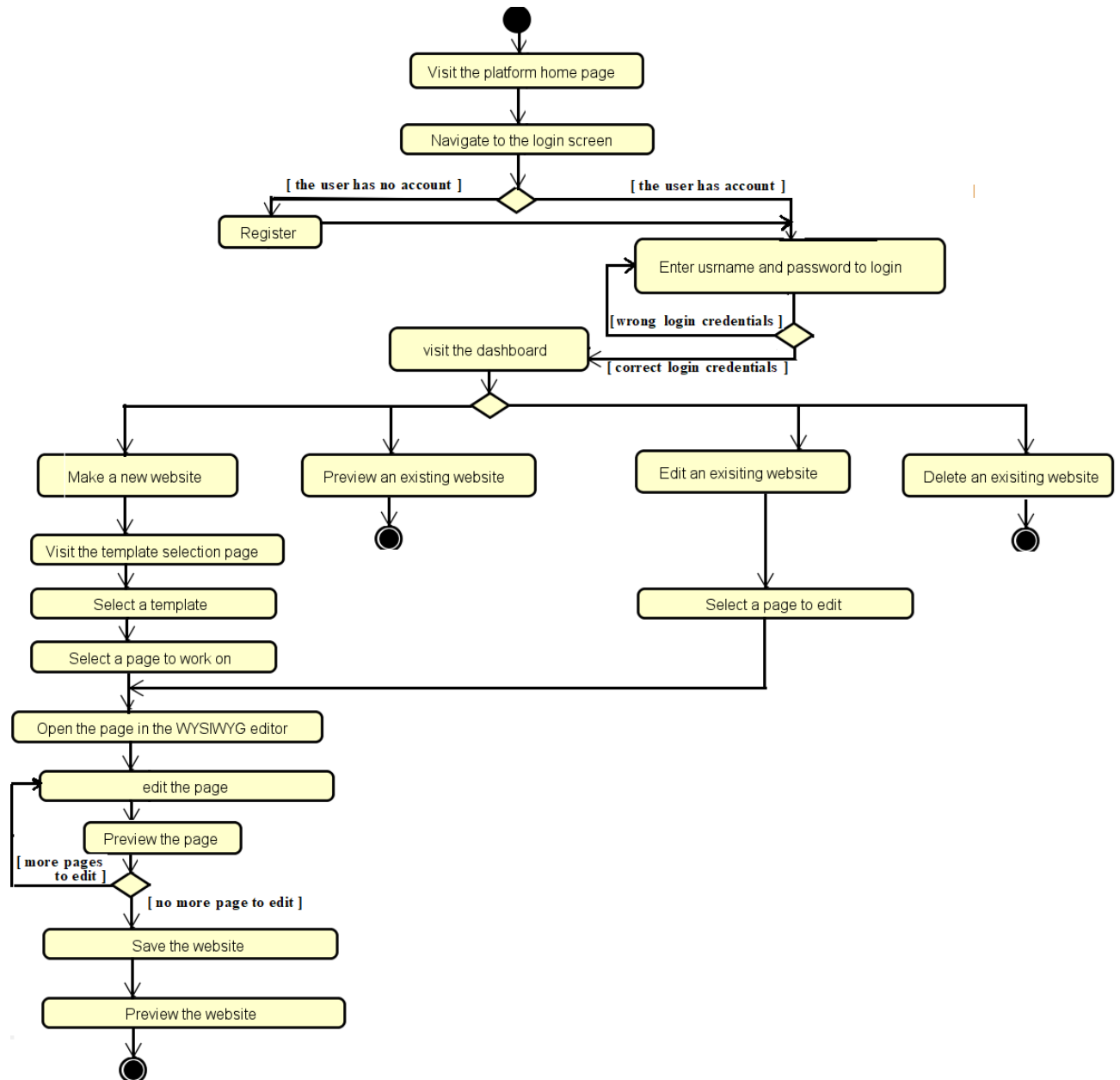


Figure 7: Activity Diagram for the Website-building Platform

5.1.2 Structural Modeling

Structural modeling is meant to produce structural models (including class-responsibility-collaboration (CRC) cards and class & object diagrams). A structural model (or conceptual model) describes structure of the data that supports business processes in an organization (Dennis et al., 2009). In this project, a class diagram shown in Figure 8 was used for structural modeling. With this diagram, the OOSAD aspect of the project is reflected.

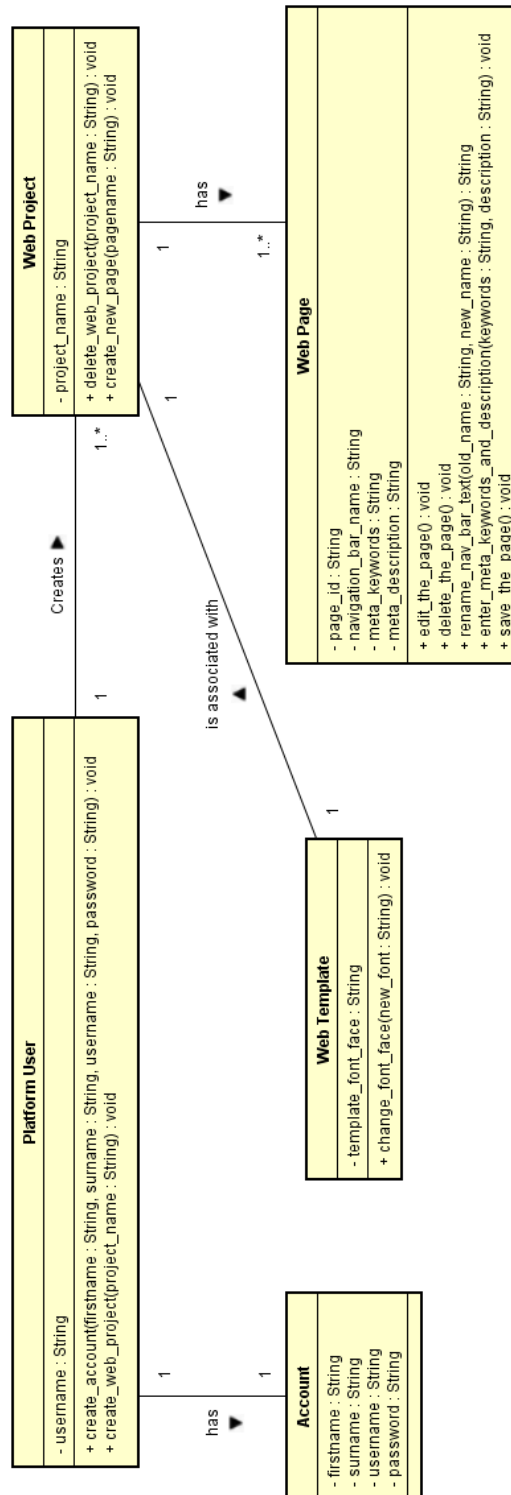


Figure 8: Class Diagram for the Website-building Platform

5.1.3 Behavioral Modeling

During behavioral modeling, behavioral models (like sequence diagrams, communication diagrams and behavioral state machines) are produced. These models depict the internal dynamic aspects of an IS. In this project, state machine diagrams were used for behavioral modeling. State machine diagrams are used when one wants to depict complex objects that can be formed when creating instances of a class (instantiation). The researcher prepared the following set of state machine diagrams for the platform.

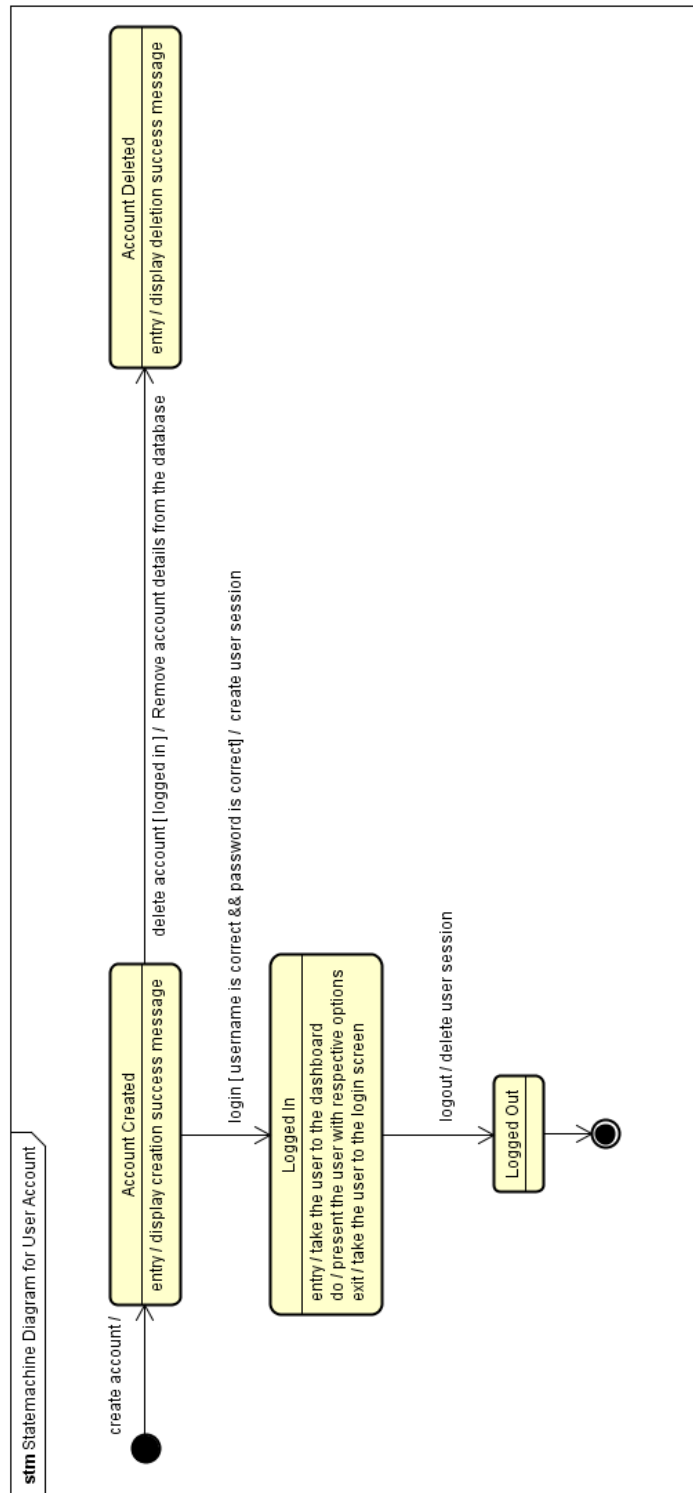


Figure 9: State Machine Diagram for User Account

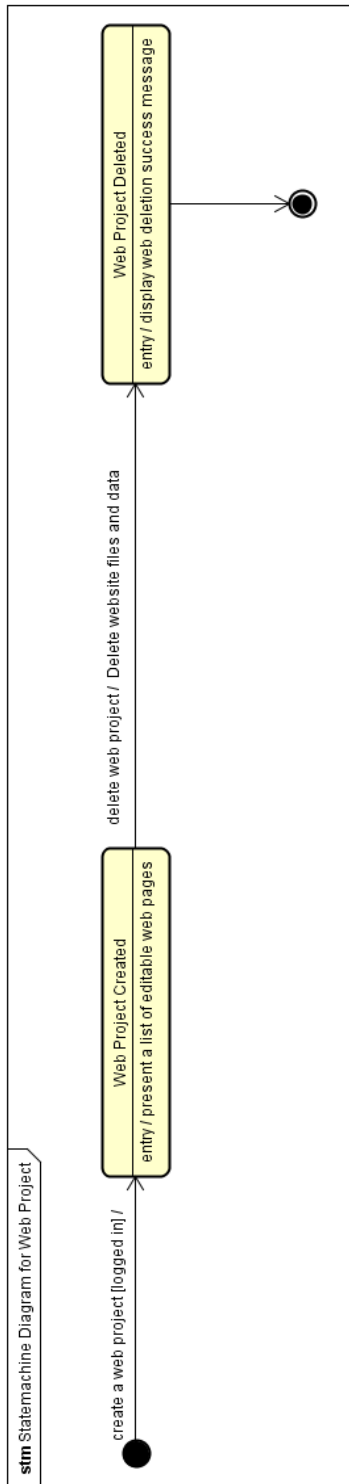


Figure 10: State Machine Diagram for a Web Project

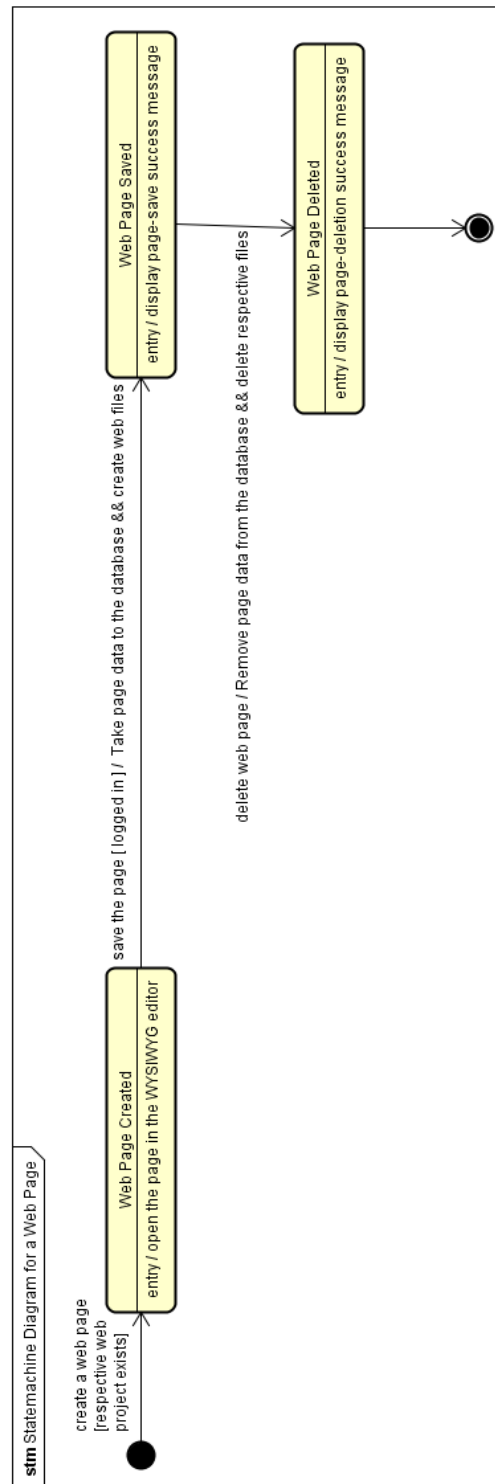


Figure 11: State Machine Diagram for a Web Page

5.2 Data Modeling

This kind of modeling involves creation of data models. The most popular method used for data modeling is ER diagramming, and the researcher decided to employ the same.

5.2.1 Entity Relationship Diagram

Figure 12 is an ERD for this project, which is closely related to the project's class diagram.

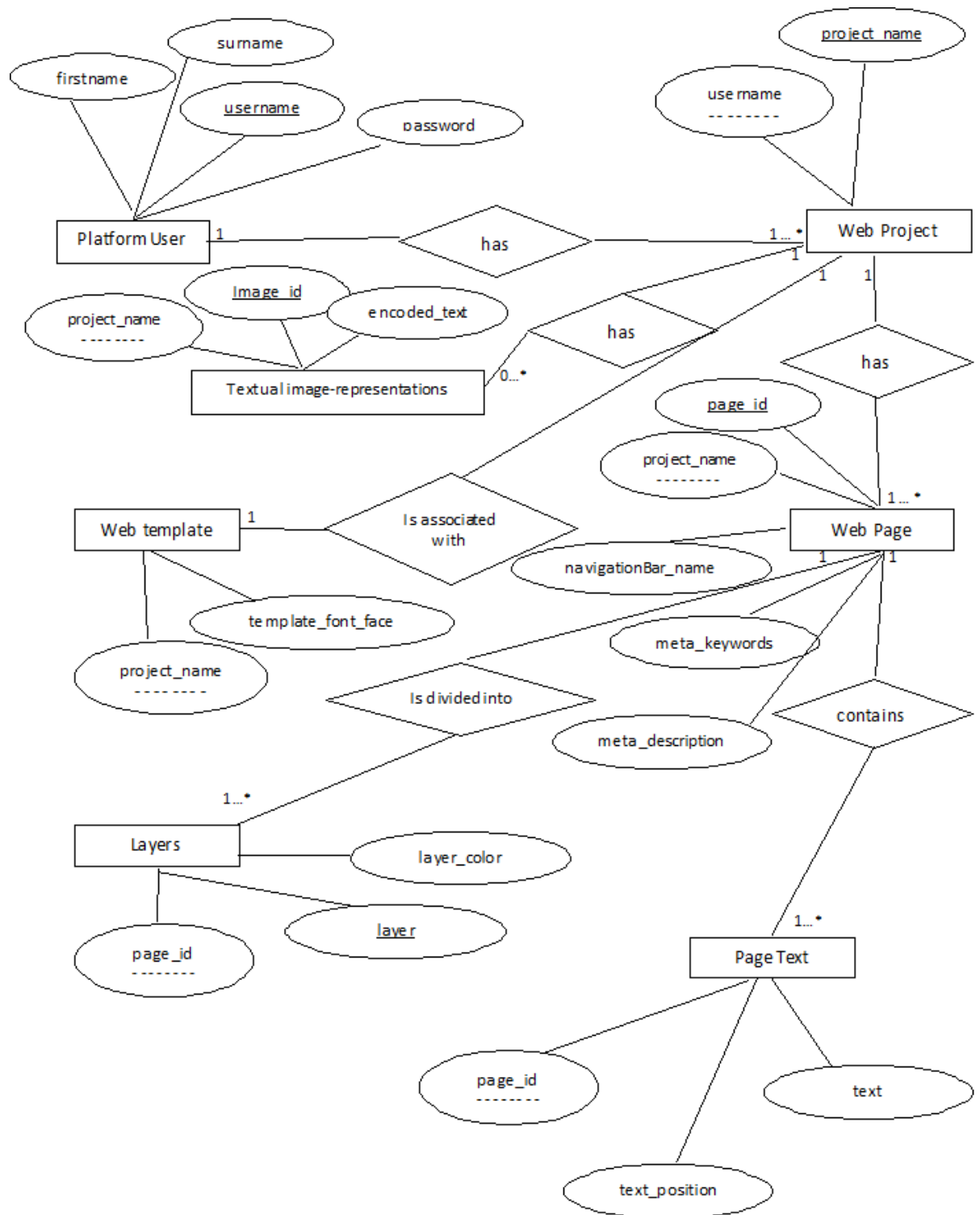


Figure 12: An ERD for the Website-building Platform

5.2.2 Normalization of Database Tables

Normalization entails restructuring database tables into a format that is free from data redundancy (or possess little data redundancy). In this particular project database tables that were initially formed were not well structured. This gave rise to the need for normalization.

There two types of database instances in this project. The first one is the “website-owners’ database” which houses account details of users of the platform. The second one is made up of numerous database instances, each of which being created when its user creates a new project. In a nutshell, the platform has only one database (instance of a database) by default, which hold account details of all the users, and whenever a user creates a new web project, a respective database (instance of a database) is automatically created to hold data for that project.

5.2.2.1 Normalizing the Website-Owners’ Database

The following table shows the look and sample data of the website-owners’ database. The table is unnormalized. It has multivalued attributes and hence is not even in the first normal form.

Table 7. *Table Name: Project Metadata (Unnormalized)*

firstname	surname	<u>username</u>	password	project
Juma	David	jumadav@gmail.com	pass123	jumadav@gmail.com_s1 jumadav@gmail.com_s2 jumadav@gmail.com_s3
Mayo	David	mayodavid@gmail.com	myPass345	mayodavid@gmail.com_s1
Juma	Makwega	jumamak@mzumbe.ac.tz	PassN3	jumamak@mzumbe.ac.tz_s1 jumamak@mzumbe.ac.tz_s2
Lulu	Pombe	lulupombe@yahoo.com	Pass123	lulupombe@yahoo.com_s1

Removing multivalued attributes from an unnormalized table takes the table to the first normal form (1NF). The Project Metadata table needed to be flattened so as to get rid of the multivalued attributes. The following is the resulting table.

Table 8. *Table Name: Project Metadata (1NF)*

firstname	surname	<u>username</u>	password	<u>project</u>
Juma	David	jumadav@gmail.com	pass123	jumadav@gmail.com_s1
Juma	David	jumadav@gmail.com	pass123	jumadav@gmail.com_s2
Juma	David	jumadav@gmail.com	pass123	jumadav@gmail.com_s3
Mayo	David	mayodavid@gmail.com	myPass345	mayodavid@gmail.com_s1
Juma	Makwega	jumamak@mzumbe.ac.tz	PassN3	jumamak@mzumbe.ac.tz_s1
Juma	Makwega	jumamak@mzumbe.ac.tz	PassN3	jumamak@mzumbe.ac.tz_s2
Lulu	Pombe	lulupombe@yahoo.com	Pass123	lulupombe@yahoo.com_s1

Note that the username attribute stood as the table primary key in the first place. However, after flattening the table, it can no longer qualify to be a primary key as it cannot uniquely identify rows. The key was therefore extended to obtain a composite key (made up of username and project attributes) as shown in Table 8.

A table in second normal form (2NF) should contain no partial key dependency (no attribute should be dependent on only part of the composite key). If there is such dependency, it should be removed by splitting the table into several parts depending on dependencies. Regarding Table 8, the first step was to separate out all the attributes that are solely dependent on username (part of the composite key). This step led to the following table.

Table 9. *Table Name: Owners (2NF)*

<u>username</u>	firstname	surname	password
jumadav@gmail.com	Juma	David	pass123
jumadav@gmail.com	Juma	David	pass123
jumadav@gmail.com	Juma	David	pass123
mayodavid@gmail.com	Mayo	David	myPass345
jumamak@mzumbe.ac.tz	Juma	Makwega	PassN3
jumamak@mzumbe.ac.tz	Juma	Makwega	PassN3
lulupombe@yahoo.com	Lulu	Pombe	Pass123

The next step is to separate out all the attributes that are solely dependent on project (part of the composite key). Fortunately, there was no attribute that is solely dependent on project. The last step involves separating out all the attributes that are dependent on the whole key. Again, there is no such attribute. Hence, the only attribute that was remaining is project. This was combined with the username attribute to form a new table. The username attribute therefore plays a role of relating these two tables, as shown in Table 10. The two attributes form a composite key for the table.

Table 10. *Table Name: Projects (2NF)*

<u>username</u>	<u>project</u>
jumadav@gmail.com	jumadav@gmail.com_s1
jumadav@gmail.com	jumadav@gmail.com_s2
jumadav@gmail.com	jumadav@gmail.com_s3
mayodavid@gmail.com	mayodavid@gmail.com_s1
jumamak@mzumbe.ac.tz	jumamak@mzumbe.ac.tz_s1
jumamak@mzumbe.ac.tz	jumamak@mzumbe.ac.tz_s2
lulupombe@yahoo.com	lulupombe@yahoo.com_s1

The third normal form (3NF) deals with transitive dependencies. This is a situation in which a non-key attribute determines another non-key attribute. In the tables obtained in

the 2NF, there was no non-key attribute that's being determined by another non-key attribute. The tables therefore remain as they are in the 3NF.

Table 11. *Table Name: Owners (3NF)*

<u>username</u>	firstname	surname	password
jumadav@gmail.com	Juma	David	pass123
jumadav@gmail.com	Juma	David	pass123
jumadav@gmail.com	Juma	David	pass123
mayodavid@gmail.com	Mayo	David	myPass345
jumamak@mzumbe.ac.tz	Juma	Makwega	PassN3
jumamak@mzumbe.ac.tz	Juma	Makwega	PassN3
lulupombe@yahoo.com	Lulu	Pombe	Pass123

Table 12. *Table Name: Projects (3NF)*

<u>username</u>	<u>project</u>
jumadav@gmail.com	jumadav@gmail.com_s1
jumadav@gmail.com	jumadav@gmail.com_s2
jumadav@gmail.com	jumadav@gmail.com_s3
mayodavid@gmail.com	mayodavid@gmail.com_s1
jumamak@mzumbe.ac.tz	jumamak@mzumbe.ac.tz_s1
jumamak@mzumbe.ac.tz	jumamak@mzumbe.ac.tz_s2
lulupombe@yahoo.com	lulupombe@yahoo.com_s1

5.2.2.2 Normalizing the Project Database(s)

As stated earlier, the second instance of the database is the one that is being formed automatically to accommodate user's website data, whenever a new web project is created. These databases (database instances) are given the names of their respective project names as shown in Table 12, being appended on the string: cbsbp (i.e. Cloud-Based Site Building Platform). Additionally, the "@" sign in these names is replaced with "At" and the "." sign is replaced with "Dot". Hence, for instance, when Juma

David (jumadav@gmail.com) logs in to the system and creates his first web project, a database named cbsbp_jumadavAtgmailDotcom_s1 will be automatically created in the DBMS ready to accommodate Juma’s website data. The following section shows how such databases (like cbsbp_jumadavAtgmailDotcom_s1) were normalized.

Table 13. Table Name: cbsbp_jumadavAtgmailDotcom_s1 (Unnormalized)

page_id	nav_bar_name	meta_keywords	meta_desc	1stTxt_pos	1stTxt	2ndTxt_pos	2ndTxt	3rdTxt_pos	3rdTxt	1stLayer	1stLayerbg	2ndLayer	2ndLayerbg
pg1	Uk1	metKw1	metdes1	pos1	aaa	pos2	bbb			L1	maroon	L2	lightblue
pg2	Uk2	metKw2	metdes2	pos1	aaa	pos2	ddd	pos3	eee	L1	pink	L2	green
pg3	Uk3	metKw3	metdes3	pos1	ccc					L1	transparent	L2	maroon

In the table (Table 13), the page_id field uniquely identifies each page, the nav_bar_name field holds the navigation bar menu name/text for each web page, meta_keywords and meta_desc fields are for page’s meta keywords and description respectively, 1stTxt_pos (i.e “1st text” position) holds an ID for the first text segment in a webpage, 1stTxt (i.e “1st text”) is the text itself, 1stLayer holds an ID for the first layer (i.e. HTML div element) of a web page, 1stLayerbg stores the background-color for the first layer/div (the background color of a layer can be changed by the user on a WYSWG editor).

The table contains repeating groups/fields, as such it is in an unnormalized form. Normalization to the 1NF splits the table into three smaller tables, which are linked by the page_id field, which acts as a primary key for the Pages table and a foreign key for the Page Text and Layer Colors table as shown in Table 14, 15 and 16. Despite being a foreign key, this field is combined with the text_position field in the Page Text table to form a composite key, and with layer field in the Layer Colors table to form a composite key as well.

Table 14. *Table Name: Pages (1NF)*

<u>page_id</u>	nav_bar_name	meta_keywords	meta_desc
pg1	Uk1	metKw1	metdes1
pg2	Uk2	metKw2	metdes2
pg3	Uk3	metKw3	metdes3

Table 15. *Table Name: Page Text(1NF)*

<u>page_id</u> -----	<u>text_position</u>	text
pg1	pos1	aaa
pg1	pos2	bbb
pg2	pos1	aaa
pg2	pos2	ddd
pg2	pos3	eee
pg3	pos1	ccc

Table 16. *Table Name: Layer Colors(1NF)*

<u>page_id</u> -----	<u>layer</u>	bg_color
pg1	L1	maroon
pg1	L2	lightblue
pg2	L1	pink
pg2	L2	green
pg3	L1	transparent
pg3	L2	maroon

All these tables remain the same in the second normal form as none of them contains any partial key dependency:

Table 17. *Table Name: Pages (2NF)*

<u>page_id</u>	nav_bar_name	meta_keywords	meta_desc
pg1	Uk1	metKw1	metdes1
pg2	Uk2	metKw2	metdes2
pg3	Uk3	metKw3	metdes3

Table 18. *Table Name: Page Text(2NF)*

<u>page_id</u> -----	<u>text_position</u>	text
pg1	pos1	aaa
pg1	pos2	bbb
pg2	pos1	aaa
pg2	pos2	ddd
pg2	pos3	eee
pg3	pos1	ccc

Table 19. *Table Name: Layer Colors(2NF)*

<u>page_id</u> -----	<u>layer</u>	bg_color
pg1	L1	maroon
pg1	L2	lightblue
pg2	L1	pink
pg2	L2	green
pg3	L1	transparent
pg3	L2	maroon

The tables maintain their structure further in the 3NF as they contain no transitive dependencies. There was an additional table in this database instance, the “Textual Image Representation” table. This table maintained its structure throughout the normal forms as it meets all the conditions to be in the 3NF. Appearance of this table is shown in Table 20. Hence, the final set of cbsbp_jumadavAtgmailDotcom_s1 database tables in 3NF is as shown in Table 20, 21, 22 and 23.

Table 20. *Table Name: Textual Image Representation (3NF)*

<u>image_id</u>	<u>encoded_text</u>
img1	data:image/jpeg;base64,/9j/4A [truncated]
img2	data:image/png;base64,iVBOR..... [truncated]
img3	data:image/jpeg;base64,/9j/1A [truncated]

Table 21. *Table Name: Pages (3NF)*

<u>page_id</u>	nav_bar_name	meta_keywords	meta_desc
pg1	Uk1	metKw1	metdes1
pg2	Uk2	metKw2	metdes2
pg3	Uk3	metKw3	metdes3

Table 22. *Table Name: Page Text (3NF)*

<u>page_id</u> -----	<u>text_position</u>	text
pg1	pos1	Aaa
pg1	pos2	Bbb
pg2	pos1	Aaa
pg2	pos2	Ddd
pg2	pos3	Eee
pg3	pos1	Ccc

Table 23. *Table Name: Layer Colors (3NF)*

<u>page_id</u> -----	<u>layer</u>	bg_color
pg1	L1	maroon
pg1	L2	lightblue
pg2	L1	Pink
pg2	L2	Green
pg3	L1	transparent
pg3	L2	maroon

What the Textual Image Representation table stores is encoded text that represent a web page image. When a user uploads an image to his web page, the image is not going to be stored in web directories (for example the web “images” folder). Instead of storing the image using this common practice, the platform will take the textual representation of the image and store it in the database. A special JavaScript function was written to enable the upload of image to a WYSIWYG webpage. As this function works (as the

image is uploaded to the WYSIWYG editor) the image is converted to an equivalent text that is made up of so many characters. Therefore, a high-precision data type is required to hold the image-equivalent string in the database. The “MEDIUMTEXT” data type was used for this purpose. Hence, the “encoded_text” field of the “Textual Image Representation” table is of type MEDIUMTEXT. This data type can hold up to 16,777,215 bytes of data which is approximately equivalent to 16 Megabytes (MBs). Figure 13 shows an image that would be uploaded to the database and its equivalent textual representation:

IMAGE Original size: 7.34 KB Extension: jpg	EQUIVALENT TEXTUAL REPRESENTATION Length: 10,059 Characters
 Figure 13: A sample image whose Textual Representation can be determined by the Platform.	See Appendix I for EQUIVALENT TEXTUAL REPRESENTATION of this image.

5.4 Front-end Modeling

Web pages are typically modeled using wireframes. These are visual representation of how a web page would look like. In this particular project, several wireframes were built to guide the front-end development. These are as shown in the following figures:

Nembo ya Tovuti					
	Mwanzo	Templeti	Msaada	Ingia	Mawasiliano

Taarifa za Mwanzo

Templeti

Templeti I	Templeti I
Templeti III	Templeti IV
Templeti V	Templeti VI
Templeti VII	Templeti VIII

Taarifa za Mawasiliano

Sehemu ya Chini

Figure 14: A Wireframe for the Platform's Home Page

Nembo ya Tovuti					
	Mwanzo	Templeti	Msaada	Ingia	Mawasiliano

Ingia katika akaunti yako

"Placeholder": Barua pepe

"Placeholder": Nywila

Ingia

Kama hauna akaunti tafadhali [Bofya Hapa Kujisajili](#)

Sehemu ya Chini

Figure 15: A Wireframe for the Platform's Sign in Page



Figure 16: A Wireframe for User's Dashboard

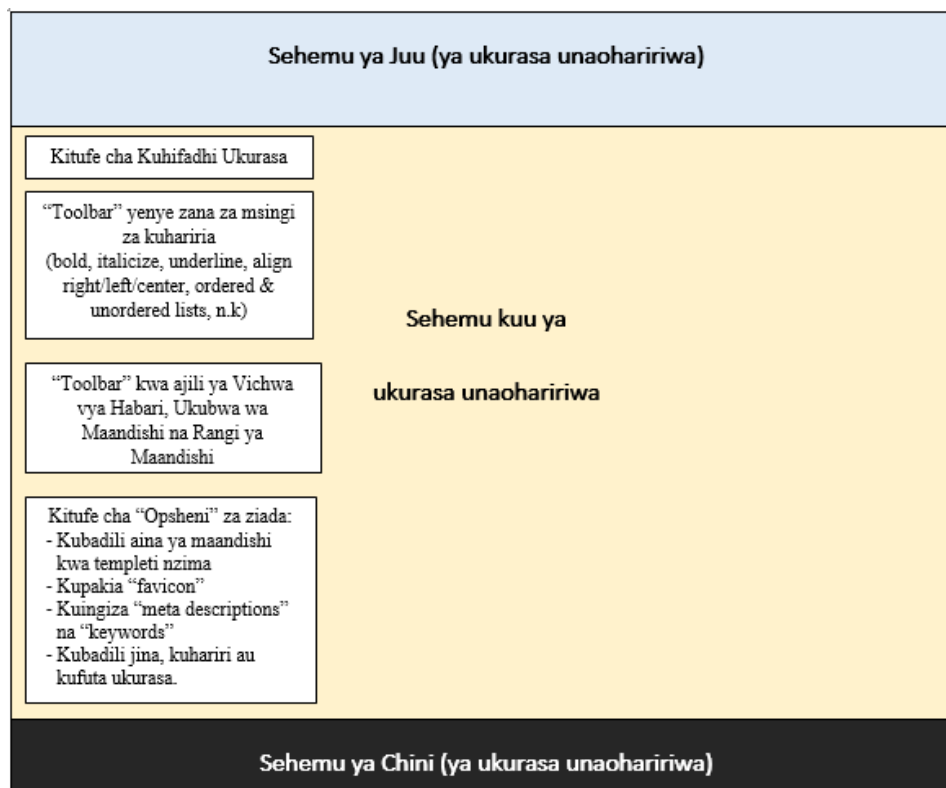


Figure 17: A Wireframe for the Page Editing/ Page Creation Interface (The WYSIWYG editor)

5.5 Chapter Summary

The Chapter has presented blueprint of the system in three sections: the first one has presented analysis modeling using a set of several UML diagrams; the second one dealt with data modeling using an ERD while ensuring database tables are normalized to the third normal form; and the final one has depicted the platform's user interface using wireframes. These designs played a very important role as they built a basis for building the platform. With this chapter, the researcher meets the third research objective, consequently answering the third research question. The following chapter describes the overall development procedure of the platform.

CHAPTER SIX

SYSTEM DEVELOPMENT AND TESTING

This chapter shows how the system blueprint was converted into a real working system. It describes about system building, that was guided by the prepared models. It also explains about the testing procedure undertaken in this project.

6.1 System Development

The system was developed using a number of languages: PHP, MySQL, JavaScript (and JQuery), HTML, and CSS. There are few occasions in which ready-made libraries were found to be useful and therefore were incorporated into the system.

6.1.1 Module Building and Integration

Ready-made libraries and the framework used contributed only a small portion of the system. Most of the modules were “custom-built” by the researcher. Even the ready-made modules and libraries had to be studied and understood before being integrated to the platform. This facilitated the construction of a well-functioning system. Two libraries were used in this project. These are “simple HTML DOM” which enabled extraction of data from web pages (web scraping) and “js color” library which enabled coloration of div elements (layers) on a web page.

6.1.2 Enhancement of the Text Editor

A simple rich text editor was used in this project. By default, the editor allows basic formatting of textual content in a content-editable HTML div element (an HTML layer/division whose content can be edited). The developer added several features to this editor to improve its WYSIWYG capability. These features include the background-coloration capability which allows a user to change background colors of page sections, the image upload feature which allows the user to change template-default images and see the result without having to reload the page as well as a

JavaScript function that allows the user to change the default font face of a web page by just picking the desired font from a given list.

6.1.3 Database Implementation

The developer reviewed the ERD of the system and the normalized tables before building the respective databases. The platform has only one primary database, which holds user's metadata. Any additional database is created upon creation of a new web project/ website. MySQL server was selected to house the databases.

6.1.4 System Builds

This project adopted the SDLC incremental model. The model involves building a system in several iterations, which are also referred to as system builds. Each of these builds plays part in enlarging the system until it becomes a product that is ready to use. Table 24 summarizes the builds involved in the project.

Table 24. Incremental Builds of the Platform

Build/Phase	Activities Done
Build I	- Designing the page-editing interface.
	- Adding form elements to the rich text editor to enhance its content-extraction capability
	- Testing basic functionalities of the editor
Build II	- The rich text editor was resized to fit 100% of any screen onto which it will be used
	- The PHP code to grab content from the rich text editor was written.
	- Interrelatedness between the text editor and the PHP code was tested.
Build III	- The look of the rich text editor was modified using some CSS styles.
	- The PHP code to grab content from the rich text editor was improved.
Build IV	- Addition of some text-formatting capabilities to the editor and removal of undesired functionalities from it
	- Rearrangement of text-formatting tools of the editor
	- Testing the modified version of the editor
Build V	- Introduction of a few sections/layers to the editor, that are not content-editable

	- Testing the functionality of these layers
Build VI	- Planning and designing how web-scraping is going to be accomplished
	- Testing the working of the “simple HTML DOM” library (web scraping library)
	- Design/planning for the positioning of a web page on the editor
	- Writing PHP code capable of taking ready-made web pages to the editor.
	- Taking a web page to the editor and testing if its components can be edited (checking if it is content-editable).
	- Testing if the edited page can be saved
Build VII	- Selecting web templates that are compatible to the editor and making little modifications to them so that they can conveniently be edited.
Build VIII	- Planning for background coloration (layer coloration) capability
	- Incorporating the “jscolor” library to the editor
	- Testing layer coloration capability
Build IX	- Design of how some CSS styles in external style sheets would be modified
	- Writing PHP code for modifying the CSS styles so as to replace the default layer colors with user-specified colors
	- Testing the functionality of this re-coloration code
Build X	- Design for image replacement functionality
	- Adding the “on-the-fly” image upload capability to the editor
	- Writing PHP code for extraction of an uploaded image in form of text
	- Testing the image replacement module
Build XI	- Database design - Implementation of database design in DBMS - Connecting the database to the web application - Checking if the web application and the database work well together
Build XII	- Design of directory structure for user files - Writing the code to enable placement of web files to respective user’s directory - Testing if each user’s site data and files are separated from other users’ data and files

6.1.5 Results Obtained

Development efforts led to production of a system with the desired functionalities. These are based on functional requirements specified in chapter four of this thesis, as shown in the following sections.

6.1.5.1 Built-in Templates

Several templates were included in the platform. They are built-in and thus the user needs to only log in to the system and start using them to make web pages/web site. Figure 18 shows a screenshot of the template-selection page of the platform.

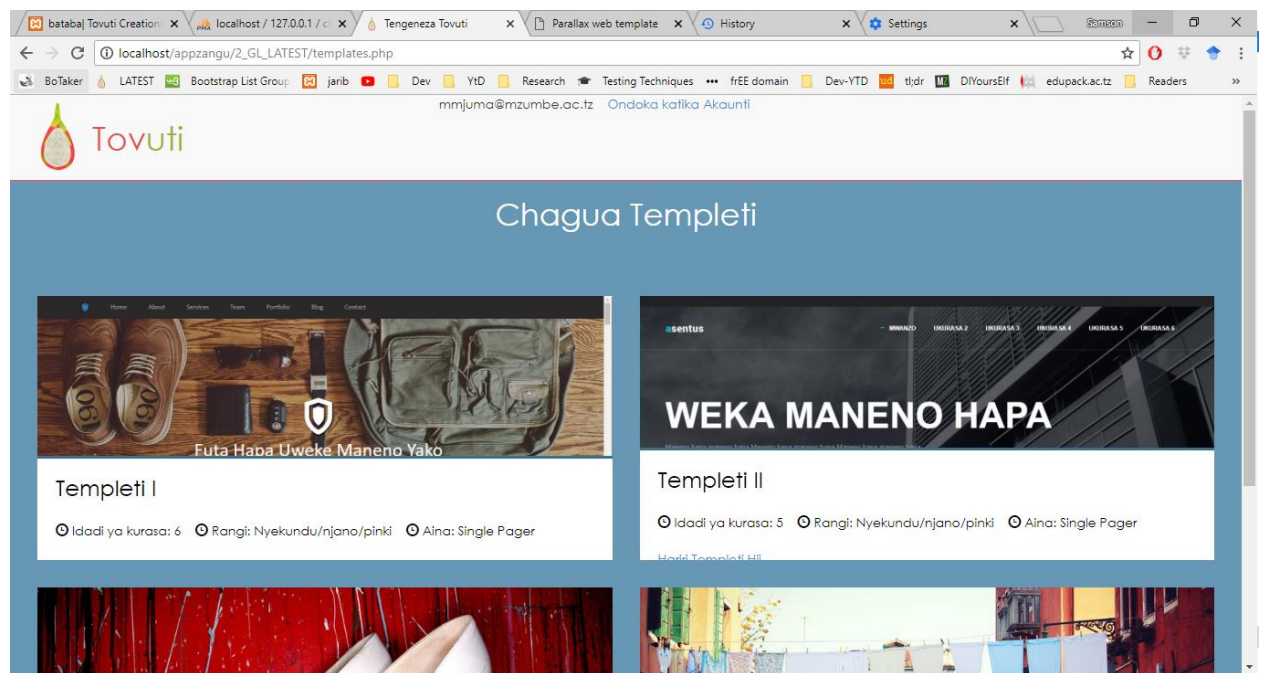


Figure 18: The template-selection page of the platform

6.1.5.2 WYSIWYG editor

The WYSIWYG editor of the platform presents a web page that's being edited in the background and several toolbars for site-editing on the front. Figure 19 shows such appearance.

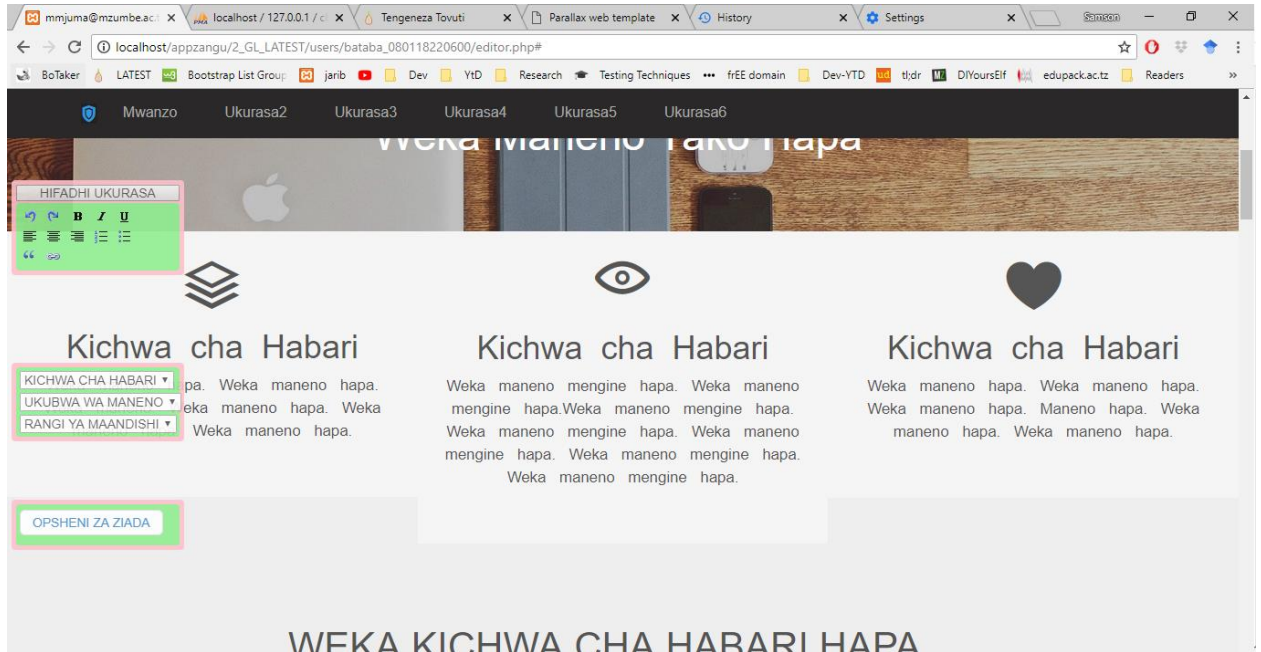


Figure 19: An editable page on a WYSIWYG editor

6.1.5.3 Code Editor

The platform features a code editor as well. This allows a user who would like to interact and change code (front-end code) to do so by simply checking the “Onyesha edita ya HTML” checkbox, as shown in the following figure.

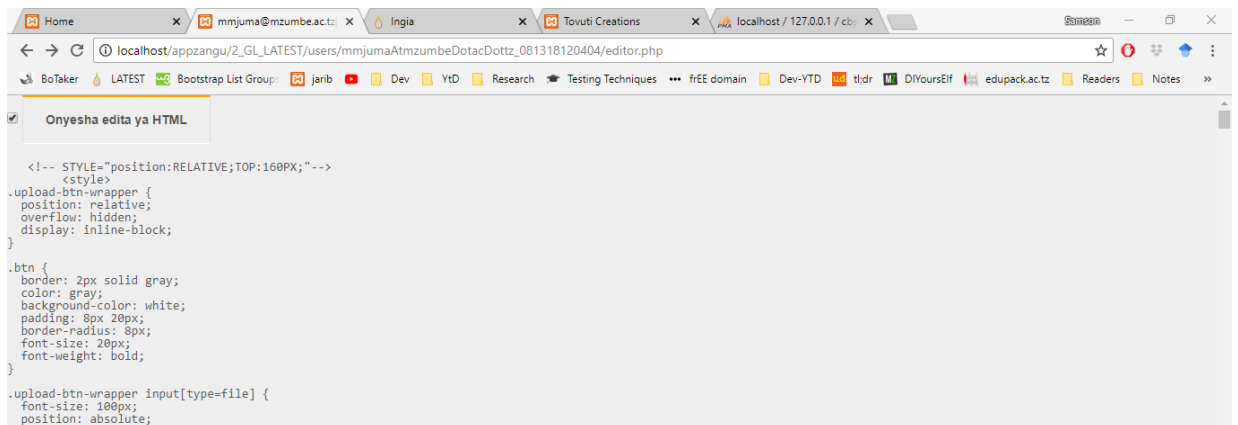


Figure 20: The Code Editor of the Platform

6.1.5.4 Meta Descriptions and Keywords

The platform allows a user to specify meta keywords and description as one of the strategies to optimize a web page for search engines.

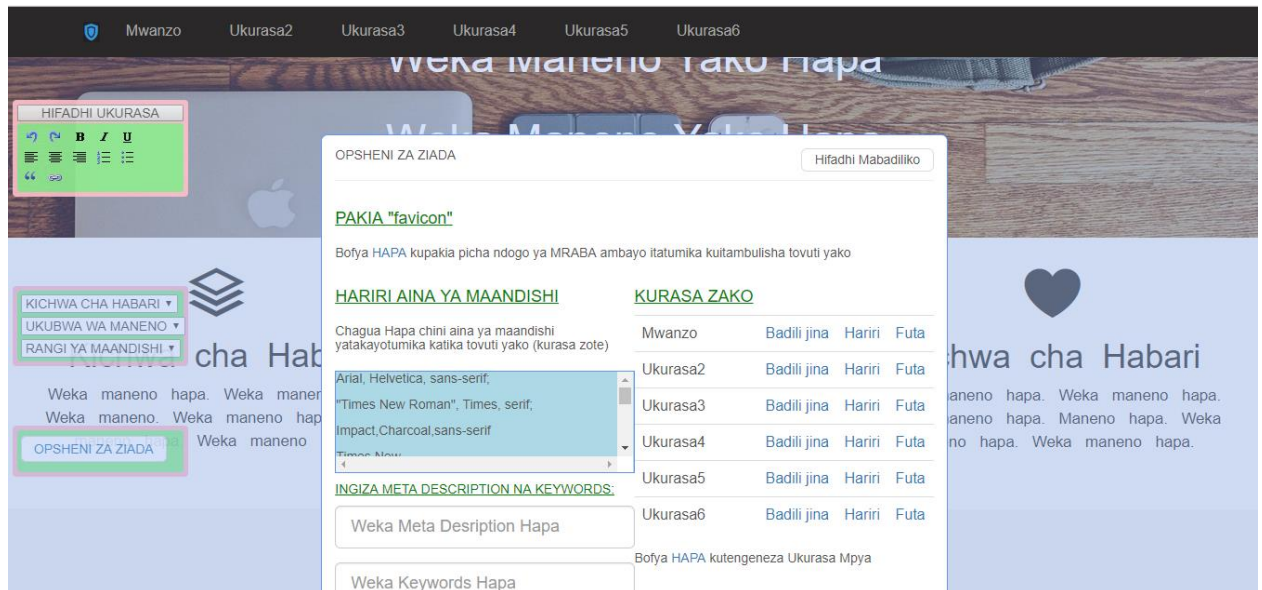


Figure 21: The Advanced Options (“Opsheni za Ziada”) Modal with which a user can enter meta keywords and description, among other options.

6.1.5.5 The Knowledge Base

To guide the user on how to go about performing certain actions in the platform. A knowledge base was prepared and presented to the user in an HTML page. This was meant to be the quickest way to help a user on usage of the platform. For instance, in situations where the user doesn’t know the meaning and use of meta keywords and description, the knowledge base would be an important repository of helpful information.

6.1.5.6 Image Gallery

In the platform, it is possible to edit the default images preloaded with templates. This is done by simply clicking a button that is presented in front of an image that’s to be edited. Figure 22 shows a page being edited with an option to replace default gallery images by simply clicking the “Badili Picha” button. This functionality was

implemented in such a way that the uploaded image is instantly viewed in the gallery (where the default image was) instantly without waiting for a user to press the submit button (i.e. the “HIFADHI UKURASA” button).

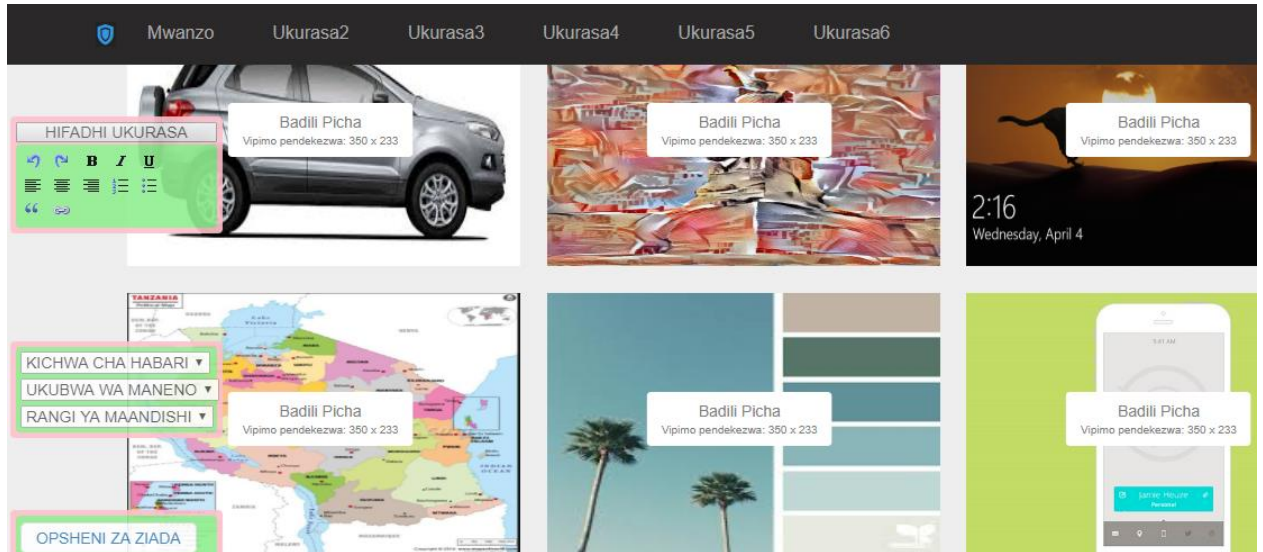


Figure 22: A customizable Photo Gallery

6.1.5.7 Site Preview Mode

The platform allows its users to preview the websites (web pages) as they are being built. This feature was implemented by just presenting the user with a link that can take him to a built site/page (the link named “Tembelea”) as shown in Figure 23.



Figure 23: User Dashboard with several options for each website, including the site preview (Tembelea) option

6.2 System Testing

A number of tests were ran during and after system building. Most of the tests were ran after the system was built. However, there are few occasions where the developer had to test a module or segment of code before building another module or code segment. This is because the modules are interdependent.

6.2.1 Test Cases

Test results were documented using test cases as shown in the following tables:

Table 25. A test case for the login module

Test Case I						
Test Case ID: T1 Test Priority: High Module Name: Login Test Title: Testing the functionality of the login module Description: The tester has to supply a valid username and password to be able to login. Again, the user should supply invalid username and/or password and the system should reject these credentials and give appropriate feedback.				Test Designed by: Samson Josiah		
				Test Design date: July 1 st , 2018		
				Test Executed by: Selected User		
				Test Execution date: July 2 nd , 2018		
Pre-condition: i) The user has valid credentials of an existing account (for successful login). ii) The user has no valid credentials (for unsuccessful login attempt).						
Steps and Outcome						
s/n	Step Description	Test Data	Expected Result	Actual Result	Test Status	Notes
1	Visit the login page	Sample account	Successful	Successful	Test	Done
2	Supply a valid user name and password (for successful login).	credentials (valid username and password) (for successful login).	login (for valid credentials).	login (for valid credentials).	Passed	
	Supply an invalid user name and password (for unsuccessful login attempt).	No data (for unsuccessful login attempt)	Login failure (for invalid credentials)	Login failure (for invalid credentials)		
3	Click the login button					
Post-condition: i) The user with valid username and password should be taken to the dashboard page. ii) The user with no valid username and password should be rejected to proceed to the dashboard and instructions on how to open an account (or restore an existing account) should be displayed.						

Table 26. A test case for Web Project Creation module

Test Case II						
Test Case ID: T2 Test Priority: High Module Name: Web Project Creation Test Title: Testing the functionality of the Web Project Creation module Description: The tester should try to create a new web project after successful login			Test Designed by: Samson Josiah			
			Test Design date: July 1 st , 2018			
			Test Executed by: Selected User			
			Test Execution date: July 2 nd , 2018			
Pre-condition: The user has logged in successfully and has an access to the dashboard						
Steps and Outcome						
s/n	Step Description	Test Data	Expected Result	Actual Result	Test Status	Notes
1	Access the dashboard		A new web project is created	A new web project was created	Test Passed	Done
2	Click the web project creation option (i.e. the “Bofya Hapa kutengeneza tovuti nyingine/mpya” option)					
3	Access the web project creation screen					
Post-condition: A new web project should be created for the user						

Table 27. A test case for Web Project Removal module

Test Case III						
Test Case ID: T3 Test Priority: High Module Name: Web Project Removal Test Title: Testing the functionality of the Web Project Removal module Description: The tester should try deleting an existing web project			Test Designed by: Samson Josiah			
			Test Design date: July 1 st , 2018			
			Test Executed by: Selected User			
			Test Execution date: July 2 nd , 2018			
Pre-condition: The user/tester has at least one web project						
Steps and Outcome						
s/n	Step Description	Test Data	Expected Result	Actual Result	Test Status	Notes
1	Access the dashboard		The selected web project is deleted	The selected web project was deleted	Test Passed	Done
2	Click the web project deletion option (i.e. the “Futa” option)					
3	Delete the project and get appropriate feedback					
Post-condition: The selected web project is removed from user’s account and the user gets informed of this.						

Table 28. A test case for the Web Page Creation module

Test Case IV						
Test Case ID: T4 Test Priority: High Module Name: Web Page Creation Test Title: Testing the functionality of the Web Page Creation module Description: The tester should try to create a new web page after creation of a web project			Test Designed by: Samson Josiah			
			Test Design date: July 1 st , 2018			
			Test Executed by: Selected User			
			Test Execution date: July 2 nd , 2018			
Pre-condition: The user has created a web project						
Steps and Outcome						
s/n	Step Description	Test Data	Expected Result	Actual Result	Test Status	Notes
1	- Go to the web page creation option <i>(if the page is to be added to an existing website)</i> - Go to the template selection page/option <i>(if a new website is to be built)</i>		A new web page is created and saved	A new web page was created and saved	Test Passed	Done
2	Access the web page on the WYSIWYG editor					
3	Edit the page					
4	Save the page					
Post-condition: A new web page should be created and appropriate data (text and images) should be saved in the database and respective files.						

Table 29. A test case for Web Page Removal module

Test Case V						
Test Case ID: T5 Test Priority: High Module Name: Web Page Removal Test Title: Testing the functionality of the Web Page Removal module Description: The tester should try to delete an existing web page			Test Designed by: Samson Josiah			
			Test Design date: July 1 st , 2018			
			Test Executed by: Selected User			
			Test Execution date: July 2 nd , 2018			
Pre-condition: The user has at least one existing web page in his project						
Steps and Outcome						
s/n	Step Description	Test Data	Expected Result	Actual Result	Test Status	Notes
1	Go to the web page deletion option		A web page is removed	A web page was removed	Test Passed	Done
2	Click on the deletion (“Futa”) option					
4	Get appropriate feedback/notification					
Post-condition: Data (text and images) of the selected web page should be removed from files and from the database.						

6.2.2 Project Achievement Test

To know whether the project was fruitful or not, the researcher conducted “project achievement test” whose intention was to check whether the “local content” aspect of the project can really ease the website-building task to native users of Kiswahili. To accomplish this, the researcher prepared two versions of the platform: one of them using Kiswahili as a service language; the other one using English language. The researcher then selected thirty native Swahili users who are not competent in English language, but with basic knowledge on websites and the Internet at large. Thereafter, the researcher

tested how the users could understand the website building options presented to them in Kiswahili versus those presented to them in English language. Functionality of the options remained the same. The only difference was the language in which the options were presented/labelled. For example, in a Swahili platform the page-saving button was labelled “HIFADHI UKURASA” while in the English version it was labelled “SAVE THE PAGE”.

The researcher first conducted the test with Kiswahili as a service language. The researcher then re-labelled the options with English words, and rearranged each option to different screen position. The reason for this rearrangement is the fact that if they were not rearranged, it would be easy for a user to guess what each option is intended to do. This is because the users had already used the same options, in Kiswahili version. Table 30 shows results of this test. The table presents the time duration used by each user in accomplishing the specified tasks. Only some of the platform “editing options” were used for this test. Such editing options are: change font size (badili ukubwa wa maandishi), change font color (badili rangi ya maandishi), change font background color (badili rangi ya nyuma ya maandishi), save the page (hifadhi ukurasa), navigate to edit another page (hariri ukurasa mwingine), rename the page (badili jina la ukurasa), and create a link (tengeneza linki).

Table 30. Task accomplishment times with Kiswahili versus English as a Service Language

TASK	Task Accomplishment Time (seconds)																			
	U1		U2		U3		U4		U5		U6		U7		U8		U9		U10	
	K	E	K	E	K	E	K	E	K	E	K	E	K	E	K	E	K	E	K	E
Change font size	8	24	10	12	7	8	9	9	11	34	5	5	8	10	8	12	11	11	8	9
Change font color	6	11	7	10	6	7	8	16	7	22	6	6	7	7	7	19	7	9	8	9
Change background color of the font	9	13	9	14	7	9	8	9	8	12	7	18	10	11	14	16	16	16	9	12
Save the page	11	9	18	19	11	7	7	5	7	7	6	6	6	9	14	13	5	12	6	6
Navigate to edit another page	13	19	14	15	11	9	16	12	12	12	8	11	9	16	22	24	18	16	12	17
Rename the page	20	22	18	29	14	14	19	21	21	27	20	26	22	28	34	39	29	29	19	32
Create a link (to an external website)	X	X	X	X	112	118	123	109	X	X	105	108	135	144	X	X	123	188	X	X

TASK	Task Accomplishment Time (seconds)															
	U11		U12		U13		U14		U15		U16		U17		U18	
	K	E	K	E	K	E	K	E	K	E	K	E	K	E	K	E
Change font size	9	12	8	16	8	8	9	12	12	15	8	7	9	13	14	27
Change font color	7	16	15	14	8	9	7	11	8	11	8	10	11	11	12	19
Change background color of the font	17	33	10	15	12	12	15	13	14	16	9	11	14	15	12	23
Save the page	7	23	8	12	7	6	8	11	10	10	7	8	10	11	13	15
Navigate to edit another page	18	22	14	36	15	17	19	18	12	14	11	17	14	17	19	27
Rename the page	28	47	25	26	22	27	19	21	24	24	21	27	23	22	29	36
Create a link (to an external website)	202	200	X	X	X	X	X	X	X	X	120	123	X	X	X	X

TASK	Task Accomplishment Time (seconds)															
	U21		U22		U23		U24		U25		U26		U27		U28	
	K	E	K	E	K	E	K	E	K	E	K	E	K	E	K	E
Change font size	10	9	7	11	8	11	9	9	10	11	12	15	8	14	6	9
Change font color	8	9	7	9	16	32	8	15	7	19	6	9	12	12	7	8
Change background color of the font	12	14	7	15	21	34	8	9	7	11	9	12	14	17	7	9
Save the page	7	8	6	15	14	22	5	6	7	9	5	5	6	8	4	5
Navigate to edit another page	17	17	16	18	13	19	17	19	16	20	21	26	22	19	14	15
Rename the page	23	19	26	29	34	43	21	18	17	19	22	25	24	25	18	18
Create a link (to an external website)	X	X	X	X	244	296	X	X	X	X	X	X	X	X	X	X

In the table, U1, U2, U3.... U30 represent the users. The letter “K” represents the Kiswahili version of the platform while “E” represents the English version. A cross sign (X) indicates that the respective user failed to perform a specified task. Results show that most of the users could not create a link (to an external website). This is possibly because they had only mediocre knowledge on how the Internet/WWW works, and possibly they had no idea on what it means by “link” or how links work. Table 31 summarizes all these results.

Table 31. Summary of Task accomplishment times for all thirty users

Task	Average Task Accomplishment Time for all thirty users (seconds)	
	With Kiswahili as a Service Language	With English as a Service Language
Change font size	9.1	13.1
Change font color	8.4	12.9
Change font background color	10.9	14.6
Save the page	8.2	10.1
Navigate to edit another page	15.2	18.1
Rename the page	23.0	26.6
Create a link (to an external website)	-	-

Based on the summary, a native user of Swahili language can build a website faster on a website builder that uses Kiswahili as a service language as compared to the one using English as a service language. For instance, the average time required for a user to change font size on a web page is 9.1 seconds on a platform using Kiswahili as a service language; and 13.1 seconds on a platform using English as a service language. This reveals that the project achieved the intended goal.

6.3 Chapter Summary

This chapter was focused on meeting the fourth and the fifth research objective and answering their respective question. It briefly reports on how the platform was built, results obtained, and the testing procedure undertaken in the project. The following chapter concludes the study and presents researcher’s recommendations.

CHAPTER SEVEN

CONCLUSIONS AND RECOMMENDATIONS

This chapter presents conclusions and recommendations the researcher would like to convey to respective parties.

7.1 Conclusions

This research project sought to find answers to four research questions which were formulated out of specific objectives of the study. The first two objectives were directly connected to the “research part” as they involved seeking knowledge and conducting simple experiments to determine the cause-and-effect relationships. The other three objectives were met through system design, building and testing. The two groups of objectives relate as the “research part” involved discovering new knowledge that served as input to the design, building and testing phases of system development. In short, seeking answers to the first two research questions was basically system analysis, which is typically conducted before the design and building phase of any information system.

All these efforts were meant to meet the general objective of the project. The objective sought to develop a cloud-based website building platform targeting native users of Kiswahili, thereby complying with the Tanzanian national information and communications technology policy, on local content development. The project achieved its intended goal: being able to deliver a WYSIWYG website builder that is friendlier to native users of Kiswahili with low level of competence in English language (and other languages commonly used as service languages in website builders).

Forthcoming studies may be focused on what has not been covered in this project. There may be a need to conduct a study on WCMSs and investigate if there is a need to have a Kiswahili-user-centered WCMS. This may be essential because WCMSs are more powerful tools as compared to simple website builders such as the one built in this project. Further, there may be a need to conduct a study aiming at building a similar

platform (a Kiswahili-user-centered simple website builder or WCMS) that is capable of operating offline so that it can be used even with those with limited internet access.

7.2 Recommendations

The researcher recommends that professionals in the research and development industry, particularly those in the information technology industry should consider the local-content aspect before building any information system. Majority of users of information systems in Tanzania are less competent in the language that is used in most of these systems (English) as compared to their level of competence in Kiswahili. Predominance of English language as a service language in most information systems makes the systems less helpful to some users. Preparing products and services that would be easily accessible to majority of local users is important.

7.3 Chapter Summary

This final chapter has concluded the study by describing achievement of the primary goal of the project. The researcher also gave opinions regarding what should be done as the Tanzanian government seeks to increase the level of localization of IT products and services in the country.

REFERENCES

- Adam, A. (2018). Are there any Parallax templates/themes on Blogger? [Online forum comment]. Retrieved from https://productforums.google.com/forum/?utm_medium=email&utm_source=footer#!msg/blogger/n3crEuLimyw/pJTJklFZEgAJ
- Al-Fedaghi, S. (2011). Developing Web Applications. *International Journal of Software Engineering and Its Applications*, 5(2), 57-68.
- Alnaqeib, R., Alshammari, F.H., Zaidan, M.A., Zaidan, A.A., Zaidan, B.B., & Hazza, Z.M. (2010). An Overview: Extensible Markup Language Technology. *Journal of Computing*, 2(6), 177- 181.
- Archbold, K. (2014). What are project exclusions? [Online forum comment]. Retrieved from <https://www.linkedin.com/pulse/20140701184106-1556206-what-are-project-exclusions>
- ASP.NET Web Application Project Deployment Overview. (n.d.). Retrieved from <https://msdn.microsoft.com/library/dd394698%28VS.100%29.aspx>
- Baldaniya, R. H., & Baldaniya, H. J. (2014). Web Development Using Content Management System. *International Journal of Emerging Research in Management & Technology*, 3(4), 166-171.
- Barjtya, S., Sharma, A., & Rani, U. (2017). A detailed study of Software Development Life Cycle (SDLC) Models. *International Journal of Engineering and Computer Science*, 6(7), 22097-22100.
- Barker, D. (2017). An Unofficial Guide to Whatever-as-a-Service [Blog post]. Retrieved from <https://gadgetopia.com/post/9981>

- Basic Design Principles. (n.d.). Retrieved from <http://www2.amk.fi/digma.fi/www.amk.fi/opintojaksot/0303011/1142845462205/1142847802793/1142848532809/1142848746737.html>
- Benjamins, S. (2018). *What's The Best Website Builder For SEO?*. Retrieved from <https://www.sitebuilderreport.com/>
- Bhaskar, N.U., Naidu, P.P., Babu, S. R. R. C., & Govindarajulu, P. (2011). General Principles of User Interface Design and Websites. *International Journal of Software Engineering*, 2(3), 45-60.
- Botwe, D.A., & Davis, J.G. (2015). A Comparative Study of Web Development Technologies Using Open Source and Proprietary Software. *International Journal of Computer Science and Mobile Computing*, 4(2), 154-165.
- Brown, C.V., DeHayes, D.W., Hoffer, J.A., Martin, E.W., & Perkins, W.C. (2012). *Managing Information Technology* (7th ed.). Upper Saddle River, NJ: Prentice Hall.
- Bulan, V. (2018). *How to install WordPress on Xampp* [Video file]. Retrieved from <https://www.youtube.com/watch?v=RoFxcP5dl28>
- Cardoso, J., & Miller, J.A. (2012). *Internet-Based Self-Services: from Analysis and Design to Deployment*. Paper presented at *IEEE International Conference on Services Economics (SE 2012)*, IEEE Computer Society, Hawaii, USA.
- Chepkemoi, J. (2017). *What Languages are Spoken in Tanzania?*. Retrieved from <https://www.worldatlas.com/articles/what-languages-are-spoken-in-tanzania.html>
- Content. (n.d.). In *Merriam Webster Online*, Retrieved from <https://www.merriam-webster.com/dictionary/content>
- Cosper, C. (2018). What you Need to Know about Google's Mobile-First Indexing Rollout [Blog post]. Retrieved from <https://www.portent.com/blog/seo/googles-mobile-first-indexing-rollout.htm>

- Coyier, C. (2013). *Website Deployment: Let Us Count The Ways!*. Retrieved from <https://css-tricks.com/deployment/>
- Dennis, A., Wixom, B.H., & Tegarden, D. (2009). *Systems Analysis and Design with UML Version 2.0: An Object-Oriented Approach* (3rd ed.). Indianapolis, IN: Wiley.
- Dhiman, S., & Sharma, P. (2016). Performance Testing: A Comparative Study and Analysis of Web Service Testing Tools. *International Journal of Computer Science and Mobile Computing*, 5(6), 507 – 512.
- Duckett, B. (2005). *Concise Oxford English Dictionary* vol. 19 (11th ed.). Oxford: Oxford University Press.
- Expert Market (2017). *Website Builders vs CMS: Key Differences Explained* [Video file]. Retrieved from <https://www.youtube.com/watch?v=dRohObPFFSA>
- Extraneous and Confounding Variables and Systematic vs Non-Systematic Error. (n.d.). Retrieved from <https://web.mst.edu/~psyworld/extraneous.htm>
- Farooq, A., Javed, F., Hussain, M., Abbas, T., & Hussain, A. (2012). Open Source Content Management Systems: A Canvass. *International Journal of Multidisciplinary Sciences and Engineering*, 3(10), 38-43.
- Fastest growing content management systems since 1 March 2018. (2018). Retrieved from <https://w3techs.com/>
- Flynn, P. (2014). *Human interfaces to structured documents: The usability of software for authoring and editing* (Doctoral dissertation, National University of Ireland, Cork, Ireland). Retrieved from <https://cora.ucc.ie/bitstream/handle/10468/1690/Human-Interfaces-to-Structured-Documents.pdf#page=69>

- Fox, S. (2013). Paradigm shift: do-it-yourself (DIY) invention and production of physical goods for use or sale. *Journal of Manufacturing Technology Management*, 24(2), 218-234.
- FTP Client Tutorial. (n.d.). Retrieved from https://www.siteground.com/tutorials/ftp/client/#FTP_Client_Software
- Galitz, W. O. (2007). *The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques* (3rd ed.). Indianapolis, IN: Wiley.
- Gambhir, D., & Singh, H. (2014). Creation of an Intranet Website for an Educational Institute Using Open Source Joomla CMS. *International Journal of Computer Science and Technology*, 5(1), 40-42.
- GeoTrust, Inc. (2011). *Creating an E-Commerce Web Site: A Do-It-Yourself Guide* [White paper]. Retrieved February 4, 2018, from <https://www.geotrust.com/resources/guides/creating-ecommerce-website.pdf>
- Gerber, E. M., & Martin, C. K. (2012). Supporting creativity within web-based self-services. *International Journal of Design*, 6(1), 85-100.
- Ghorecha, V., & Bhatt, C. (2013). A guide for Selecting Content Management System for Web Application Development. *International Journal of Advance Research in Computer Science and Management Studies*, 1(3), 13-17.
- Ghuman, S.S. (2014). Software Testing Techniques. *International Journal of Computer Science and Mobile Computing*, 3(10), 988-993.
- Girgis, M.R., Mahmoud, T.M., Abdullatif, B.A., & Zaki, A.M. (2014). An Automated Web Application Testing System. *International Journal of Computer Applications*, 99(7), 37– 44.
- Giri, K.K., & Nirgude, K.R. (2009). *Open Source Content Management Software: A Comparative Analysis*. Paper presented at 7th International CALIBER-2009, Pondicherry University, Puducherry (pp. 184 – 193).

- Goel, R., & Gupta, N. (2014). Survey on Acceptance Testing Technique. *International Journal of Software and Web Sciences*, 8(1), 20-23.
- Hain, S., & Back, A. (2008). Personal Learning Journal – Course Design for Using Weblogs in Higher Education. *The Electronic Journal of e-Learning*, 6(3), 189-196,
- Historical yearly trends in the usage of content management systems for websites. (2018). Retrieved from https://w3techs.com/technologies/history_overview/content_management/all/y
- Hsu, C., Guo, S., Chen, R., & Dai, S. (2009). *Using Domain Ontology to Implement a Frequently Asked Questions system*. Paper presented at 2009 World Congress on Computer Science and Information Engineering.
- Importance of Modularity in Programming [Blog post]. (2018). Retrieved from <http://aosd.net/importance-of-modularity-in-programming/>
- Jaiswal, A., Raj, G., & Singh, D. (2014). Security Testing of Web Applications: Issues and Challenges. *International Journal of Computer Applications*, 88(3), 26– 32.
- Javed, H., Minhas, N.M., Abbas, A., & Riaz, F.M. (2016). Model Based Testing for Web Applications: A Literature Survey Presented. *Journal of Software*, 11(4), 347–361.
- Jiang, Y., & Dong, H. (2008). *TEB: A Template-based E-Commence Website Builder for SMEs*. Paper presented at 2008 Second International Conference on Future Generation Communication and Networking Symposia.
- Johansson, A. (2016). *Searching for the Perfect Website Builder? Here's What Matters*. Retrieved from <https://www.entrepreneur.com/article/285916>

- Kadam, V.N. (2013). *Analytical Study of Domain Name System, its disputes and legal issues*. Paper presented at *Proceedings of National Conference on Emerging Trends: Innovations and Challenges in IT*, 19 -20, April 2013.
- Kattara, H. S., & El-Said, O. A. (2014). Customers' preferences for new technology-based self-services versus human interaction services in hotels. *Tourism and Hospitality Research*, 13(2), 67-82.
- Khraim, H.S. (2015). The Impact of Search Engine Optimization Dimensions on Companies Using Online Advertisement in Jordan. *American Journal of Business and Management*, 4(2), 76-84.
- Kihoza, P. D. (2016). *A Framework for Online Resources and E-Learning Implementation (OREI) in Tanzania Secondary Schools* (Doctoral dissertation, Nelson Mandela African Institution of Science and Technology, Arusha, Tanzania). Retrieved from <http://scholar.mzumbe.ac.tz/bitstream/handle/11192/2037/Patrick%20Daniel%20Kihoza%20dissertation.pdf?sequence=1>
- Klein, D., & Gubic, A. (2014). Responsive website design for higher education utilizing mobile centric features. *Online Journal of Applied Knowledge Management*, 2(1), 69-81.
- Ko, H., & Cheeyong, K. (2015). Application Possibility of Parallax Scrolling Technique for Website Users' Emotional Experience. *Journal of Korea Multimedia Society*, 18(2), 277-286.
- Kothari, C.R. (2004). *Research Methodology: Methods and Techniques* (2nd ed.). New Delhi, India: New Age International Publishers.
- Kumar, S., & Goudar, R. H. (2012). Cloud Computing – Research Issues, Challenges, Architecture, Platforms and Applications: A Survey. *International Journal of Future Computer and Communication*, 1(4), 356-360.

- Kumari, K.B.M. (2014). Software Testing for Finding Errors. *International Journal of Scientific Research and Education*, 2(2), 283-289.
- Laguador, J.M., Tupaz, A., De Castro, E.L., & Dotong, C.I. (2016). Development and Evaluation of an Institutional Research Repository Using Content Management System. *International Journal of Applied Engineering Research*, 11(6), 4017-4023.
- Lane, K. (2010). Cloud Platform as a Service (PaaS) [Blog post]. Retrieved from <http://kinlane.com/2010/06/10/cloud-platform-as-a-service-paas/>
- Lesson 1: Basic Features of Web Authoring Software. (n.d.). Retrieved from <https://www.washington.edu/accesscomputing/webd2/student/unit7/module1/lesson1.html>
- Lestari, D.M., Hardianto, D., & Hidayanto, A.N. (2014). Analysis of User Experience Quality on Responsive Web Design from its Informative Perspective. *International Journal of Software Engineering and Its Applications*, 8(5), 53-62.
- Litwin, B. (2017). *CMSs and Website Builders: What are they and how do they compare?*. Retrieved from <http://www.buzzle.com>
- Luzardo, G., Avendaño, A., Chiluita, K., & Verdezoto, N. (n.d.). *A Web Content Management System to Create Accessible Web Sites*. Unpublished manuscript.
- Martinez-Caro, J., Aledo-Hernandez, A., Guillen-Perez, A., Sanchez-Iborra, R., & Cano, M. (2018). A Comparative Study of Web Content Management Systems. *Information*, 9(27), 1-15.
- Minton, L. (2013). *Designing Experiments* [Video file]. Retrieved from <https://www.youtube.com/watch?v=GCGuXLdhQbs>
- Mishra, A., & Dubey, D. (2013). A Comparative Study of Different Software Development Life Cycle Models in Different Scenarios. *International Journal of Advance Research in Computer Science and Management Studies*, 1(5), 64-69.

- Mutula, S. (2008). Local content development projects in Africa. *SA Jnl Libs & Info Sci*, 74(2), 105-115.
- Naithani, S., & Kaushik, A. (2016). Evaluation and Comparison of the Conceptual Frameworks of CMS. *International Journal on Computer Science and Engineering*, 8(9), 354-361.
- Nasit, Y. (2018). *Install Joomla CMS On Online Web Server* [Video file]. Retrieved from <https://www.youtube.com/watch?v=ciJ16heqH-E>
- Nawarathne, N.P.R. (2012). Overhead of FTPS and FTP over IPsec in IPv6 networks. *International Journal of Scientific & Engineering Research*, 3(11).
- Nayak, P. (2016). Exploring Platform as a Service [Blog post]. Retrieved from <http://cloudsearch.blogspot.com/2016/01/exploring-platform-as-service.html>
- Nazir, M. (2012). Cloud Computing: Overview & Current Research Challenges. *IOSR Journal of Computer Engineering*, 8(1), 14-22.
- Nordahl, S. (2016). *Animations Effect on Reading Comprehension in Web-based User Interfaces* (Master's thesis, KTH Royal Institute of Technology, Stockholm, Sweden). Retrieved from <http://www.diva-portal.org/smash/get/diva2:947353/FULLTEXT01.pdf>
- Novak, M., & Švogor, I. (2016). Current Usage of Component Based Principles for Developing Web Applications with Frameworks: A Literature Review. *Interdisciplinary Description of Complex Systems*, 14(2), 253-276.
- Olusola, M., & Sunday, I. (2013). Evaluation of Content Management Systems Performance. *IOSR Journal of Computer Engineering*, 9(4), 62-69.
- Organizing Your Social Sciences Research Paper: Types of Research Designs. (2018). Retrieved from <http://libguides.usc.edu/writingguide/researchdesigns>

- Randolph, B. (2018, January 2). How Does Mobile-First Indexing Work, and How Does It Impact SEO? [Blog post]. Retrieved from <https://moz.com/blog/mobile-first-indexing-seo>
- Random Sampling. (n.d.). Retrieved from <http://www.businessdictionary.com/definition/random-sampling.html>
- Request: CMS System. (n.d.). Retrieved from <https://support.wix.com/en/article/request-cms-system>
- Robles, P. (2018). Google's mobile-first indexing is finally rolling out: Here's what you need to know [Blog post]. Retrieved from <https://www.econsultancy.com/blog/69908-google-s-mobile-first-indexing-is-finally-rolling-out-here-s-what-you-need-to-know>
- Rohilla, N. (2017). Web Content Management System (WCMS): A Need. *International Journal of Engineering and Computer Science*, 6(6), 21729-21731.
- Salas-Zárate, M.P., Alor-Hernández, G., Valencia-García, R., Rodríguez-Mazahua, L., Rodríguez-González, A., & Cuadrado, J.L.L. (2015). Analyzing best practices on Web development frameworks: The lift approach. *Science of Computer Programming*, 102, 1–19.
- Sami, M. (2012). Software Development Life Cycle Models and Methodologies [Blog post]. Retrieved from <https://melsatar.blog/2012/03/15/software-development-life-cycle-models-and-methodologies/>
- Schäferhoff, N. (n.d.). *WordPress vs. Joomla: Which is the Right CMS for you?*. [White paper]. Retrieved May 1, 2018, from <https://wpengine.com/wp-content/uploads/2017/02/WP-WP-WordpressVsJoomla-v02-PUB.pdf>
- SDLC - Iterative Model. (n.d.). Retrieved from https://www.tutorialspoint.com/sdlc/sdlc_iterative_model.htm

- SDLC - Overview. (n.d.). Retrieved from https://www.tutorialspoint.com/sdlc/sdlc_overview.htm
- Shah, R. V. (2012). *Building a Web Content Management System* (Masters thesis). San Diego State University.
- Sharma, M.K. (2017). A study of SDLC to develop well engineered software. *International Journal of Advanced Research in Computer Science*, 8(3), 520- 523.
- Sharmila, S., & Ramadevi, E. (2014). Analysis of Performance Testing on Web Applications. *International Journal of Advanced Research in Computer and Communication Engineering*, 3(3), 5258-5260.
- Sharwood, S. (2018). *WordPress is now 30 per cent of the web, daylight second*. Retrieved from https://www.theregister.co.uk/2018/03/05/wordpress_is_now_30_per_cent_of_the_web/
- Sheokand, R.N., & Vinay (2015). A Comparative Study of Content Management Systems: Joomla, Drupal and WordPress. *International Journal of Advance Research in Science and Engineering*, 4(1), 176-183.
- Shirazi, F., & Iqbal, A. (2017). Community clouds within M-commerce: a privacy by design perspective. *Journal of Cloud Computing: Advances, Systems and Applications*, 6(22), 1-12.
- Singh, N., Giri, M., & Mathew, S. (2015). Responsive Website: A Transformation in Web Designing. *International Journal of Engineering Technology, Management and Applied Sciences*, 3(1), 182-190.
- Singh, S.P., & Maini, R. (2011). *Review of Domain Name System* Paper presented at *National Workshop-Cum-Conference on Recent Trends in Mathematics and Computing*.

- Software Testing - Documentation. (n.d.). Retrieved from https://www.tutorialspoint.com/software_testing/software_testing_documentation.htm
- Sommerville, I. (2011). *Software Engineering* (9th ed.). Boston, MA: Addison-Wesley.
- Sridaran, R., Padmavathi, G., & Iyakutti, K. (2009). A Survey of Design Pattern Based Web Applications. *Journal of Object Technology*, 8(2), 61-70.
- Srivastav, M. K., & Nath, A. (2016). Web Content Management System. *International Journal of Innovative Research in Advanced Engineering*, 3(3), 51-56.
- Stanley, C. (2012). *Principles of Experimental Design* [Video file]. Retrieved from <https://www.youtube.com/watch?v=kssXJWAiz7U>
- Tomsho, G., Tittel, E., & Johnson, D. (2007). *Guide to Networking Essentials* (5th ed.). Boston, MA: Thomson Course Technology.
- Types of Website Builder. (n.d.). Retrieved from <https://digital.com/website-builders/>
- United Republic of Tanzania (2016). *National Information and communications technology policy*. Retrieved from <http://www.mwtc.go.tz/uploads/publications/en1490101734-National%20ICT%20Policy%202016.pdf>
- ur Rehman, K., & Khan, M.N.A. (2013). The Foremost Guidelines for Achieving Higher Ranking in Search Results through Search Engine Optimization. *International Journal of Advanced Science and Technology*, 52, 101-110.
- van Beuninge, J., de Ruyter, K., Wetzels, M., & Streukens, S. (2009). Customer Self-Efficacy in Technology-Based Self-Service: Assessing Between- and Within-Person Differences. *Journal of Service Research*, 11(4), 407-428.
- Vissers, J. (2017). 5 Website Builder Features Your Small Business Needs [Blog post]. Retrieved from <https://www.merchantmaverick.com/5-website-builder-features-your-small-business-needs/>

- Watson, M., & Shove, E. (2008). Product, Competence, Project and Practice: DIY and the dynamics of craft consumption. *Journal of Consumer Culture*, 8(1), 69–89.
- Web App Framework vs. CMS vs. Website Builder. (n.d.). Retrieved from <https://www.learneroo.com/modules/9/nodes/617>
- Webb, O. (2016). Features of Web Authoring Software [Blog post]. Retrieved from <https://quizlet.com/76259012/features-of-web-authoring-software-flash-cards/>
- Website builders: features you should look out for. (2015). Retrieved from <https://internet.com/website-building/website-builders-features-you-should-look-out-for/>
- What is Website Builder Software: Analysis of Features, Benefits and Pricing. (2018). Retrieved from <https://financesonline.com/website-builder-software-analysis-features-benefits-pricing/>
- Williams, C.C. (2008). Re-thinking the motives of do-it-yourself (DIY) consumers. *The International Review of Retail, Distribution and Consumer Research*, 18(3), 311–323.
- Wojcik, M. (2015). *Capstone Blog: My Project Proposal*. Retrieved from <http://michaelwojcik.org/blog/2015/08/capstone-blog-my-project-proposal>
- Wolf, M., & McQuitty, S. (2011). Understanding the do-it-yourself consumer: DIY motivations and outcomes. *Academy of Marketing Science*, 1, 154–170.
- Yadav, P., & Barwal, P.N. (2014). Designing Responsive Websites Using HTML and CSS. *International Journal of Scientific & Technology Research*, 3(11), 152- 155.
- Yang, Y., Kim, Y., Lim, Y., & Lim, H. (2016). A Comparison of Open-Source CMS and Analysis of Security Vulnerability. *International Journal of Computers*, 10, 82-86.

APPENDICES

Appendix I: Equivalent Textual Representation for Figure 13

Length: 10,059 Characters

[illegible]

qJLBE0aRnrv70EZMX4seB2Nd+6SVOVQZjcfSa9n7wC2FTFK0HxbLWdWdxQYsbYWHWOQV5vmc53dl5117R4nVR+LNNYEXFIR6T2m3nSaUtrvrsS1+roWTjTNN
YWjaSPIHJqJGvZdTtTGRh+2j72clpLb8jEdu/2BSfC8ain0aSHWuW00dbpHAjsJUEillqRaSWIm5HJzQsz92bU9oUafVXSzvjajTHJACxdc5GFgzEuB0DC3Xm2
Nio0vJfQnlsY+FU7ji3I8gZ2eS63sIsr1HpW3UBERAxzztZUWrakdE0n75X0MvnPwoxmHE52nc8tlb1te0a+kHDuVxNaPfp7xntWmwKnL6o5dwBJ9QXtVVF
2kLrszJaWQ9QPrKb8n0sbCkEfpzmjzqVQUY Y22igtDjQbukcPSW2ixprZ0zR1ucG+1aqRv5Ga9unu/12LxfiEcXjG56B71oNodqYYYyR11/AZXBxcegW4daqF
toZ6g6uLWnc1p07zvKlIUOfwkrXhKOSBHBZ7nzCy70nhbaTYif3Y4tPrJVGRA4dfk27dFJhsG47NVKsfrtDtrA8nRzA43sbHK47y0je07+BvfftoFwG8srgR0
j/wHl8y0WJSwnmul+adW+ZWHstXnGhs4AZmHcR9Y6+HtuakWDjBtBI0nFD5H2NLIa0SAZbm+lhq9GiwqjZyR5vykLL8HOcT5mtKo04mXYTL YDZOb8
dT+eQfwL8tUp+EtOf03fW1BreWXYTL YfFgP8qA/rR9YT40iFbsZ7JWe9BrGT849jfa9evLr3+KtZncOSBIA06PzXL7ceors7ZitH93f3EH60GOJ12E65dgNWN9
PL6JK8nYZUjFBKPOHe5B68unLrFdTsJfIO1jvcvllw3tC04nhy6fCota3lke/Wg2oqj0nzr1iqXu0BK07Zrrf4PDzXvPBrreYJofgrBHG0nxso9gW1hdmN3arSNls
o/tts2nacrGLaF28k9DBXPX/+oLEqK+KIXke1vadfNvWjrNsaMaZnO7G6eshfPeM7aTzOOuLOPE8557Sdy0Tp5n6l8ju9XUqx9D10P0z9zy00W9iWjKkb2m46Q
qlZVzMOj3jvPsK3uEbXPaQJN3lDQ9449yUi2RiN9DqsZ1VYrS0eItkAI11Fxbcr1L1tLQojbCvG7NYHgd3eDovGoijeCHN0dbNkWB2UggODTI0aAsRuCi7+I
wKlNfKlI42zXsL2LjmJnJzaL6BeFa91PXy0wmfJfEw5WOCguSodm0G69/UoLk2Aqc88gbfK1xcL5ub5hm86nqqXwRVtnVM7A9+QRNeWkNyFznOYCCG
3BAJ4uN8x0CtpZ1oREQFqcZ2apKog1FPFKWiwc9oLgLS3bHwFtkQqerE+FP15BzPoTSgebnZQLwmYNBhscVJTU7RHU0pHGR8kxis0AOzWillPNsRzjx
V6qsfDxhpdSw1LrTy2d1MlFr+m2PzoKcJY7yX+i73Ll7rDU761rQQSWSObre2uneCucTjyrBeXONhdxJOu/fw3qe29znM9a1FTKZX+oDqWygEdgBmed
NBc36AFjYUy13dG5TFY6niivPUMJY+J73PsfvTNA0t07M65Nm/VYamH6ZpfGZSLNabODQZHi/Sxq5qtn54msdmbeQZmsddj7dbTuPUVIsf2cmhbUftQIY
Z2cyZjuaDfmShyHAmxHDS TGNnWyOhnqJ+TggY1hcDmfIWeO2MDVz3009asRBailOJZl0teOkWK1jHvgkBGjmmul9XvmbVUTqxrxlsWQCJskQNmvaOof
C5psTe1wRpr2qv8RAewP4i3mKipfQbQnKMP1BIACQ9wAuflWF7A6cfUswYvMNOXYf1r/3lCcluxzDw3d/wDVbEuHvFj2q1lILceqB+EZ+0h+sr3ZJU9LD
+IAfY5Qou614uq2bPLCZrVD5DT3NP7r172tqBvgB/QefY5VmarqXHWnqSkWZHp1HuJp97WDRko3F/b0+pZbNvb6d3nePa1VS2rsTv3D616DEHdLvP/V
KRbLPCGzJE4frD9bV6i818PH1b+sh9FUGxR/lvH6R967DFn/jH+k5WkXAZwjU/ly+kw/L2bt/Tn5cne1h/iVN/d3lu77lPumfKHe1p9oSXp8cqU73j9KIH6iurto
qJ2/kd2wf8ASqZ+H/Q9BnuXYVo6l/Rb9QSKXEMSO8Dimp6BafOLLxrxmm5M8hKxpNhyZcNcxAOV3fuKqcVXzW+a3sK5+EG40G/peN2vlJSJzi20JDHXNh
Y3PQOKqfFK99TL3t3ua3oHv6VsdokP2TUnnmndmcRYa8SepYWGRhrS89g96m6R6Q0rxd2rfUfshStkBJ35GuvlLyGA232BW82XwQSMJIFUY2vgDrBjS8k
Mab7nutp29RWdhkVwaZ1PM1r545WvDZImy2BuHizwRmaL44dlQqL VIFPElywyukA5rZm2PG43Dtc1FTRNcM0fm9yss0M8mOqpJclHFYy19wMjRYM
b5bnuvu9147VhrWSSTr8nGJS1zPk5Xaxy07K4EXtxPDVlQbOPyM y3hjjzXHT5ruB6gp9C0yDeAeNzZvisIBDxud7VtQvKLGnO7UdPeaJmpuJFW7Kkv
MrJ3Mfvu13Vw3e1Y02CYf7ZHyZ3NtraxLaLab+gWWQdiMnlH1e5Bi0g4n1K+hY3grdI3Eo2CwaY5S699eaLZbXGYG2+2l7HgbzVG+BGmVFU+V73B9M0O
QQBLE2vskZvrQ2PeFeSzzqiliAiilGhR6KOaNOUrGvjeLOY4Atl6wVklgrHHfAxsSkuppZKYn5fHJEONcQ4dma3QFD8U8BtaRopp3i97ueK8wDvar+RB8b
S07oeVid48b3sNtrmYMS026rhTqvgxEcnZSse+F0TCLNa5gBY3eToBY8vR/AAR4KaTFJ9LR1J+EMPDn/wBoO0PzadBHSvXZStmZGU7pLiFr4XaslglOVzQd
z2kgacLdaullcFHIIInBrKWUMB5VsWmV5J5rxcHosACbaN517WC4gZKKcyvMdNCAWF4YXEEHKwhmU2aC0AuFrEG4Jcl/snikhrhSsY6OCBkrsgB1a2Nw
c54td5doLnp67L1xQVkuFUG9kb43BskMkbWtsCHF7RyYcGkOaRpfgcKtr1wTz+c1QqXVkuTOCbuExfmNiA0tt42o0UBeyrbrbuHZfRTJaOcQRRPfGY53U5jg
hsAiUvLUWnFuNrtB6bdBUAQ3gNDQpp48aCfl4kmwlt7FnfdJvle1XF/4ed8sUtdKwffvVUVx8hpu946i4fqyrhNHh+LZ6I9yivjy7oN6fUhr29I839f9hHD4vxUf
oN9y6OwqA74Yj+rb7kHx/8OZ830R7kFUz5not9y+u3YDSnfTQH9Uz3LqdnKp8AJKf9jH9lB8jCf2fN9Ee5dvhLN9m+YL60dsVqnQ0dMf1Mf2V4P2Mw46mgp
D+oi+yg+UTVs6G+v6kEzOged3vX1Z8ScN/IKT9rhF9ldTsNhn5BS/sY/cg+VDUR9A87veuRKzyfW736nOwOf/kFN+zaup8H2F/kNP6AQfLkN8n2pyjOg+ftU
DvBxhR/uMPmI9hXQ+DPCfYGL/N70HzKypaN1x3+8L2u8Gnzke5fSbvBzB0+BM7nyj1h68T4JMHP9z/41R/MQfMmKVieW24A+uyyqdoLGNvo5zWnsL
H2q1fDB4NaWloxU0MBjMTxywzyPvG7QOs9x3Oy7uDj0Koad122B1G5BosXsfPUS0zPGPDbEhmNc/LYNPN1JAOgN7KT1kpbGIZW11yhr/AHx2Ytvm5K
ws7TTKMXfyTlsL4mHY5Nuw14IEeaMayrSHKskb/amkfvIOhbbKSHb9Fh4Vj4lqgYmW5CCB8gu7NKZGyNle9z785zuq1uBWsZbeRslTG0xwZy0Axc54lu7
K9rI72kLLkG+oy666LTTU1a9ldHO14igLmksyNzBmbm6WdYjhoBeW1+PNLoYzsslPLFA+JtufHdgHKMkHEOuCnb2Ta2uMVO9nKtelayCnizE2YGtM88t
9X0Ju0E+XpuRuArTYx239g86Orystcbyp14MsBbW4jFDIvPhjD5ZWncWtGVoPa9zfWrdxPwM4VL4ckbB33jclFJmA7gsq+dWDrLwjC6msk5OnhfK7iGDQd
b3HmtHaQfB9V4A4fwVbI36cTx/uuYrG2K2bbh9JHSH/KZC5xfly5nOCXE5bm2+287KEW8E3g+kw/PPUOaZ5Whga03bGy+Ygu+U4kC9BIfrqx0RAREQER
EBERARYFdh7pN1RNf9Dk9PSYVo6jZCd27Fa8djoPqiCDp4Sni2YnTLzLS8ZLOJCPfCbXa62ur1gD2A62svmivpZ6SfKpmvgniI4OBB0exw3joIX0dJsNVHdj
NcO0xn2NC0uOeCOSrDRUYpU5SDdueNjLi77G4KcrovCDUM0ncr15c4yvkbeKV7ehz2bu4A9a8aXbOSBpbSwRwAm5zPdLr0hrzYHjeynj/AAPK4g4dtOD/8
AYF5n/wAP7vzj/wAt/wB5KKjxDenyPdLi90kr97nG5PUOgdQ3Kd+D/wAE9RWpbNWNbDTDXK4FssvzWt3safKPCOIIE+BWopnF9PipjclNphm7jytx3LcjY
HFvz9N+w/7qCkXSmZGxscQ1jGhrWgWDWgWDQOgBeyrcbDYv+fpP80P5i5GxOMfn1/wDhm/bQWOim4m4z+fxf4Zn2lz8T8a/Ph/wrPegsVFXxSxv89/8
rGuPirjvDG2/wCFj9yCkxvCHZfH/wA8s/w0f2V1Oz0OH54i/wANH9bBZKKr/i9f+doD/7dn8tPuDtH+dKY9sDf5aCyEYbfcXaT840Z7YR/KXP3J2l/L6l/q/8At
LIRVv9zdpvyvgP6Dv5S4NDtP8AlOH9HF/8tBZKktvgu0/47Dj3P+wnI7T/AlzD23SfZQWHWUzJGOjkaHMe0tc04c1wsQR0EFfMvhF8Hc+GyOlja6SkceaXJ
jB3Ml6lbg7ce3RWvk2n8rD8Aie5cPbtMQQRhrdCDylj1IKHwTGH07+UjDXh2j4n6skFiLEdOpseHnBmMO2lI2JoIsa8RmXO4BswaGG4ZEHEAC+vijptfV
ZWNeCbFJ5TLyNDCXb2wPcxhPlZDcNPZYdSwx4HcWHCD9r/RB1dt1COM+8GskizcnLO1keTNqbhl8wrv5vbbvE6+umq6gyOvLPM4NDWN7msYxu4cAB
7blSt/gdxc/Jh/aj3KUBf7GYthxL4qKhfbkgzSSvc+vxvW2dZg14C54oJt4Kti/ufTEy/wDmZ8rpeLaAOEDxDbm54knhZThQNmlBQcaOg/bPWRHIGoFkO6LuqHj
+AoJoiJuHVwFwE1JTMbxc2qe

Appendix II: Project Timeline

ACTIVITY										
Project idea generation										
Concept Note writing										
Approval of project idea										
Preliminary Analysis										
Literature Review and Proposal writing										
System Design										
System Building and Testing										
Report Writing										
	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	July	Aug
	2017		2018							

Time

Appendix III: Project Budget

Item	Amount (Tsh)
Stationeries	400,000/=
Responsiveness-testing devices (mobile phones, computer monitor and a tablet)	1,800,000 /=-
Travel costs	200,000/=
Internet costs	200,000/=
Miscellaneous	500,000 /=-
TOTAL	3,100,000/=

