ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

ONLINE AMHARIC-ENGLISH, ENGLISH-AMHARIC MULTIMEDIA DICTIONARY

By

Ahmed Aragaw

A Project paper submitted to
the School of Graduate Studies of Addis Ababa University in partial fulfillment of the requirements for the
Degree of Master of Science in Computer Science

July 2008

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

FACULTY OF INFORMATICS

DEPARTMENT OF COMPUTER SCIENCE

ONLINE AMHARIC-ENGLISH, ENGLISH-AMHARIC MULTIMEDIA DICTIONARY

By

Ahmed Aragaw

Name and Signature of Members of the Examining Board

1. Dr. Dejene Ejigu, Advisor _____
2. Dr. Dida Midekso _____

ACKNOWLEDGEMENT

First of all, I would like to express my deepest gratitude to my advisor Dr. Dejene Ejigu for his encouragement and advice throughout this project.

I am thankful to Tamirat Tesfaye for his fruitful and valuable assistance in reading this document and give me his suggestion and recommendation.

Finally, I would like to thank my family for their continuous support, moral and initiative through out the project.

TABLE OF CONTENTS

List of Figures

Acronyms

Unicode	Unique number (code point) - A standard for representing characters as integers.
IIS	Internet Information Service
HCL	Hardware Compatibility List
ASP.NET	Active server Page.NET - A Microsoft Corporation web development tool
WEFT	Web Embedding Font Tool
eot	embedded font type

ABSTRACT

Dictionary is the list of words with their definitions. An online English-Amharic, Amharic-English multimedia dictionary is a dictionary that provides an online translation service from English to Amharic and vice versa. As the name implies, it is a multimedia dictionary that displays the meaning of the word with multimedia content. Multimedia is an application that uses multiple modalities to their advantages including text, image, drawing/graphics, animation, video and sound. Displaying the meaning of the word with different multimedia content will help the users to understand the meaning of the word very easily. The sound/voice support helps the user to know how the word is pronounced correctly; picture also describes the meaning of the word better than text i.e. *"a picture worth more than 1000 words"*.

In this project, we propose and implement an online English-Amharic, Amharic-English Multimedia Dictionary. The multimedia dictionary provides an online dictionary service for both Amharic to English and English to Amharic word translation. It displays the meaning of the word with multimedia content that can describe more about the word and its meaning.

To display Amharic text online, a WEFT tool is used. WEFT is a Microsoft's utility for generating embeddable web fonts. It creates "Embedded Open Type" files for inclusion on the web site. These files usually use the extension ".eot". Using .eot file, it is possible to embed the Amharic Unicode font in the system. The embedded Amharic font helps Amharic words to be displayed in any computer whether the computer has an Amharic font or not. To write an Amharic text, the Amharic typing JavaScript code is developed that maps all Amharic Unicode values to their phonetic keyboard combination. Using the script users can write any Amharic word. The Amharic typing JavaScript code works like any Amharic phonetic keyboard combination. Using WEFT and Amharic Typing script, Amharic text can be displayed in any computer and users can write Amharic text.

The online dictionary has different searching facilities and displays the result using appropriate multimedia content. In addition to displaying the meaning of a word with

multimedia content, the online dictionary supports users' interaction. It allows the users to post comments, requests and even they can add new words in the dictionary. They can also give comment on the meaning of the word, post additional meaning or post a better meaning.

The online dictionary also has a separate administration part that helps the system administrator to manage dictionary functionalities that includes management of dictionary data, users, word category, security, conjugation, additional meanings and assign privilege to the users.

For multimedia data, different tools are used for capturing/recording, processing and displaying. For an audio content, cool edit 2000 is used for capturing and processing. Microphone is the hardware tool that helps for audio recording. JavaScript popup audio player is used to play the audio content. The JavaScript popup audio player window is a client side program and makes the audio to play with out refreshing or submitting the form to the server.

For picture content, the main sources of picture files are Clip Art CDs and Internet. The hardware to capture pictures files are scanner and Digital camera. For image processing, Adobe Photoshop is used for cropping the picture in to equal size and to make some modification.

For video content, an ASPNetVideo component is used. The ASPNetVideo Component Suite allows to add video files and streaming media into ASP.Net web forms. For processing of Video files, Adobe premier is used.

The development Environment that is used in this system is ASP.NET, IIS is the web server and SQL server 2005 is used for database Implementation.

1. Introduction

1.1 Overview

"A dictionary is a list of words with their definitions, a list of characters with their glyphs, or a list of words with corresponding words in other languages. ..."[1]

There are different kinds of dictionaries available online and offline (printed version). Some of the dictionaries deal with the mapping of words from one language to other language. There are also dictionaries that work with in the same language like an Oxford Advanced Learners Dictionary that gives the meaning of an English word with another simplified word expression. There are many countries that have their own dictionary to translate foreign language to their local language and vice versa. It helps them to learn and understand the foreign language in an easy way. Therefore, dictionary plays the biggest role in translating one language to the others; it is one means of understanding foreign language

.

Online dictionary is a process that facilitates translation service using Internet. It is typically managed searching environment with a navigation menu and icons giving access to automated tools and content pages. An online dictionary is a web based systems that doesn't need any software installation and that can be accessed easily with the internet browser software. The only thing an online dictionary needs is an internet connection and a computer. Therefore, as long as we can get the translation of a word online any time any where, there is no need to buy different dictionaries or ask somebody to translate a word.

Multimedia consists of applications that use multiple modalities to their advantage, including text, images, drawings (graphics), animation, video, sound (including speech) and most likely, interactivity of some kind [2].

One means of getting translation of English to Amharic word is using the print versions of Amharic dictionary.

The print version of the dictionary has the following drawbacks

- It is time consuming - locating the required word takes a lot of time and the user has to pass different stages i.e. first he/she has to locate the first letter of the word and then to locate the second, the third till the last letter of the word.
- Users should have the dictionary at hand; the users should have to carry the dictionary with them. Many people don't feel comfortable time and place to have the dictionary with them.
- user should have a means to get the dictionary, buy or lend
- The dictionary is not available any time any where; availability of the dictionary is limited.
- May be damaged and unable to find the required word.
- Some of the page may not be available because of different reason.

Using the printed version of the dictionary has been only means of getting word translation, but now it is not convenient means especially in today's rashly world and due to the above mentioned drawbacks.

The other means of getting word translation is using online dictionaries. Online dictionaries solve some of the drawbacks of the printed version dictionaries.

Online dictionaries provide the following advantages

- Fast searching facility
- It is available any time any where
- Users don't need to buy or lend
- Free from damage unless the system is down.
- Different means of displaying the meaning of the word.
- Update from time to time

One of the limitation of online dictionary is one has to have a computer and an internet connection other wise it is unable to do the translation of the word.

In the online dictionary, most of the words in the dictionary will be displayed using multimedia content because the meaning of words can be easily described using multimedia. For example: a sound support for pronunciation, it will help the user to

know how to pronounce the word correctly. A picture also describes the meaning of the word in a better way - *"a picture can worth more than a thousand words"*.

1.2 Problem Statement

There are few Amharic dictionaries available online for the translation from English to Amharic or from Amharic to English. The method of displaying the result of a search word is only using text. Since Amharic has different words that have the same type of writing but different sound and meaning, expressing the meaning of such kind of words using text is difficult and users can't understand and distinguish them easily. Such kind of problem can be solved if the dictionary has a voice support that will help the user how to pronounce these words.

There are also Amharic words that are unique to Ethiopia and expressing such words using text is difficult but using multimedia content for example picture can help the users to understand what it means very easily. Non Amharic speakers who want to use the online dictionary can understand the word and its meaning very well if the dictionary has multimedia content.

Therefore, an online multimedia dictionary solves the problem of the exiting text based dictionaries and helps the user to understand the word and its meaning very easily. The online multimedia dictionary will have an audio/voice support, picture and video in addition to text.

1.3 Objective of the Project

1.3.1 General Objective

The objective of this project is to analyze, design and develop enhanced online English-Amharic, Amharic-English multimedia dictionary.

1.3.2 Specific Objectives

- different ways of searching and displaying mechanism
- to make the system interactive to the users.
- to enhance the dictionary by adding multimedia content

- develop an easy administration part
- implement security facility for the system
- optimum utilization of computer resources

1.4 Significance of the project

Since Amharic is the working language of the Federal government and some states in Ethiopia and English is becoming the defacto international language, an online English-Amharic, Amharic-English Multimedia Dictionary will benefit both Ethiopians and Foreigners (Non Amharic Speakers). The dictionary will help users who want to get the meaning of a specific word from English to Amharic as well as from Amharic to English.

The multimedia feature of the dictionary helps to describe the meaning of the words effectively. There are words in Amharic that are unique to Ethiopians and describing such kind of words using text is not sufficient to understand what it means. Similarly some English words can be easily described using multimedia contents.

In addition to the above benefits, word conjugation in the dictionary can be used as a means to learn the language and, therefore, any foreigner who will use this online dictionary can learn Amharic. This contributes to the nation tourist attraction by helping them communicate with the local people.

The outcome of this project is that, any Ethiopian or foreigners can make use of this online multimedia dictionary to get the translation of words from Amharic to English or from English to Amharic.

1.5 Scope of the Project

The scope of this project will include the following activities related to the recording, updating, retrieving and deleting records online.

- managing online English-Amharic, Amharic-English Data
- Upload multimedia contents
- Users management & assign privileges

- Accept user request, comment, better meaning as well as additional meaning of the existing words.
- manage user requests
- Search facilities
- Conjugation words on how to conjugate some Amharic words - display the variety of a word in using a sentence
- Security mechanism of the system
- additional manning to the word

For this project, a paper version MERIT English-Amharic Dictionary is used as the source of our dictionary data because it has as many words as 44,000. Word categorization and structuring will be adapted from this dictionary.

Due to budget and time limitation, only representative words will be encoded to test the functionalities of the system. The insertion of all dictionary data will be left for data encoders upon permission from the authors of the MERIT dictionary.

1.6 Document Organization

This project report is outlined as follows: Chapter 2 discusses related works on English-Amharic, Amharic-English dictionary and Amharic based website on the method of displaying and using Amharic text. In Chapter 3, system analysis of the project is discussed. Chapter 4 discusses the system design of the system, and the proposed system architecture. The tools/technologies used to implement the prototype of the project is discussed in chapter 5. Chapter 6 presents the conclusion, recommendation and future works.

2. Literature Review

2.1. Amharic Based Dictionaries

Most of Amharic based Dictionaries are doing the translation of English word to Amharic and there are only few that perform the translation in both ways i.e. from English to Amharic and vice versa. Their method of displaying the meaning of the word is different but almost all dictionaries display the meaning using text.

2.1.1 Method of displaying Amharic words

Since Amharic is a language spoken only in Ethiopia, to use Amharic text on computers, there must be local based software tools that help to display and write Amharic text. There are Amharic software like PowerGeez, Visual Geez and others that can do the above mentioned functionalities. But when we come to online systems, the server (the resource provider) and the clients (the computer that request the service of the server) are far apart. Since the client computer uses the fonts installed or available in the local machine, a computer which don't have installed Amharic software or fonts can not display Amharic contents. Because of this, some of the Amharic based dictionaries use images of Amharic text that will be displayed as an image instead of text. This kind of approach is very inefficient because it will increase the download time of the page. Saving all dictionary Amharic words in an image format needs too much computer resource like hard disk space and memory space. Such kind of system also not suitable for Amharic to English words translation. Another important point is that user interaction with the system using Amharic script is impossible.

Ethiopian Dictionary [4] is one of Amharic based dictionary that translates English words to Amharic using the above mentioned method. Ethiopian Dictionary does the translation only in one way i.e. from English to Amharic because the system is unable to user Amharic fonts. The translation from Amharic to English is impossible since it requires the use to type Amharic words in order to get its meaning. It uses pronunciation of phonetic symbols to help the user how to pronounce the Amharic word using an English alphabet

For Example, for an English word Student - ተማሪ *TeMaRee* will be displayed. Here, the Amharic word is image format instead of Amharic letters. The pronunciation information for the Amharic word is ***TeMaRee.***

Since Amharic has sounds that are not found in English, in order to help the reader correctly pronounce these words, there is a chart that will tell the user how to pronounce such words.

The system accepts English word and translates it in the following ways, if the word has more than one meaning, the navigation button helps to see the next available meaning of the word. Figure 1 shows user's interface for the result of the search in Ethiopian Dictionary.

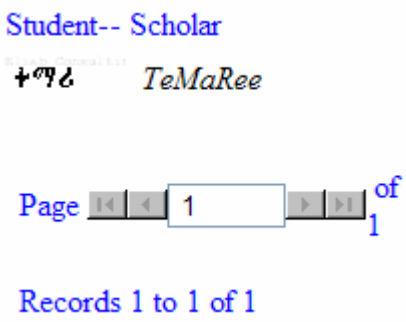

Figure 1: The result of the search and how to pronounce it

The other and efficient method of using or displaying Amharic word is by embedding Unicode Amharic fonts in the webpage. Microsoft developed a Web Embedding Font Tool (WEFT) that helps to use any Unicode fonts to display the required text. There are many countries that have their own language and doesn't support by the system or need its own software to display it. Using this tool WEFT, it is possible to use any Unicode fonts to use or display the required text. Most of Amharic based website nowadays are using this tool to display Amharic text in their website.

Web Embedding Fonts Tool, or **WEFT**, is Microsoft's utility for generating embeddable web fonts. WEFT is used by web developers to create 'font objects' that are linked to their web pages so that users using Microsoft's Internet Explorer

web browser will see the pages displayed in the font style contained within the font object.

WEFT scans the HTML document file(s), the TrueType font file(s), and some additional parameters. It adjusts the HTML files and creates "Embedded Open Type" files for inclusion on the web site. These files usually use the extension ".eot".

Using the ".eot" file it is possible to create a style sheet and apply to Amharic Unicode text to display Amharic text. If the web based system or the website uses an ".eot" file, the system will use the embedded font instead of the fonts that are found in the local machine. Therefore, by using WEFT, any Amharic based system can display Amharic text whether the local machine has Amharic software or not. It will not refer the fonts found in the local machine.

One of the dictionaries that does the translation of English-Amharic, Amharic-English is AmharicDictionary.com, **Online Dictionary of the official language of Ethiopia** [3].

The search has the following interface.

Figure 2: An interface that accepts users input

AmharicDictionary.com is one of Amharic based dictionaries that uses WEFT to display Amharic words. It is an efficient and better way than the previous one.

The result of the search is table based as shown below. The result of both search type is the same style, it uses the same sequence of column when the translation is done in both ways. Figure 3 shown below is an Amharic-English translation and figure 4 is English-Amharic translation, here it doesn't show whether the user inputs Amharic or English.

Amharic	Pron.		English	
ደረሰ	däräsä	ግ	arrive	W
ደረሰ	däräsä	ግ	write books	W
ደረሰ	däräsä	ግ	be delivered	W
ደረሰ	däräsä	ግ	to write	W

Figure 3: The result of Amharic-English translation

Amharic	Pron.		English	
አስተማሪ	astämari	ሰ	teacher	W
አስተማሪ	astämari	ሰ	instructor	W
ማስተማር	mastämar	ግ	teach	W

Figure 4: The result of English-Amharic translation

AmharicDictionary.com uses a better style than others; it also includes the pronunciation information for Amharic words that helps the users how to read correctly. There is no pronunciation information for English words. It also displays the type of word in both language i.e. English and Amharic. Some of the limitations of the dictionary are

- it uses only text to display the meaning of the word.
- it includes pronunciation information but there is no information how to relate the pronunciation information with the Amharic letters. No syllable information for the sound of Amharic.
- method of displaying the result is tabular form with the same fashion whether the user searches an Amharic or English word. It is difficult to distinguish the searched word and its meaning.

2.1.2 Method to write Amharic words

One of the methods to display Amharic text on the web is using WEFT tool, by embedding the Unicode fonts in the website. But for Amharic-English, English - Amharic Dictionary, displaying Amharic fonts is not enough because for Amharic to English translation the users have to write Amharic words and WEFT doesn't

help to write Amharic words. Therefore, there must be another means that makes the user to write/type Amharic words.

One of the methods to write Amharic text is using the installed Amharic software in the local machine. Using this method for example the web administrator can write Amharic words that will be uploaded or inserted to the remote computer or server but for the online dictionaries that involve user interaction, this kind of method is not suitable because the online dictionary will be accessed from different corner of the world and the client computer may not have an installed Amharic software.

The other method that is suitable for online dictionaries or any web system is to create a program/software that can be embedded with the system. The software or the program helps the users to write Amharic text. All Amharic alphabets and numbers are assigned a Unicode value and the program creates the mapping of Amharic Unicode values with the phonetic keyboard combination. It is time taking task and requires knowing all Unicode value and letters. This is almost creating the Amharic software that provides the facility to write Amharic text.

The Amharic typing program is a client side JavaScript code that will run on the client machine and it is preferable, if it follows the phonetic Amharic keyboard combination. The phonetic Amharic keyboard combination follows the sound of Amharic and it is easy to users to use it. Therefore, having its own Amharic program or script is the only way that makes the users to write Amharic words. It makes the system to be interactive and the dictionary to operate in both directions i.e. from Amharic to English and from English to Amharic.

2.2. Development tools for Amharic online system

One of the most persistent problems in multilingual technology has been exchanging documents between applications and operating systems. To make documents more readable across platforms and machines, **the Unicode specification** was created and has been implemented in many systems [10]

2.2.1 Unicode

Computers store data as numbers, even textual data. An **encoding** system, such as **ASCII**, assigns a number to each letter, number or character. Operating systems include programs and fonts which convert these numbers to letters visible on the screen and computer monitor.

Unicode, also known as **UTF-8 or the "Universal Alphabet"** is an ordered set of over a million characters covering the majority of writing systems in the world. Unlike older systems, Unicode allows multiple writing systems to co-exist in one data file. Systems which recognize Unicode can consistently read and process data from many languages

Unicode Fonts

In order to display Amharic text, Amharic Unicode fonts should be embedded in the system. There are few Amharic Unicode fonts developed, among them Visual Gee'z Unicode and Power Gee'z Unicode are well known. Using one of the Unicode fonts, it is possible to create ".eot" file using WEFT tool and then it can be used in a style sheet.

There are several ways you can type or import Unicode text, but each page must include an encoding meta tag specifying the utf-8 Unicode encoding, so that browsers render the text correctly. See the code below:

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
...
</head>
```

For Amharic character, the Unicode value is assigned from &#4608 - &#4988, there are 380 Amharic characters including the numbers and each character has its own unique Unicode number

ETHIOPIC UNICODE CHARACTERS

Character Name	Character	Entity	Hex Entity
ETHIOPIC SYLLABLE HA		ሀ	&#x1200
ETHIOPIC SYLLABLE HU		ሁ	&#x1201
ETHIOPIC SYLLABLE HI		ሂ	&#x1202
ETHIOPIC SYLLABLE HAA		ሃ	&#x1203
ETHIOPIC SYLLABLE HEE		ሄ	&#x1204
ETHIOPIC SYLLABLE HE		ህ	&#x1205
ETHIOPIC SYLLABLE HO		ሆ	&#x1206
.....			
.....			
ETHIOPIC NUMBER TEN		፲	&#x1372
ETHIOPIC NUMBER TWENTY		፳	&#x1373
ETHIOPIC NUMBER THIRTY		፴	&#x1374
ETHIOPIC NUMBER FORTY		፵	&#x1375
ETHIOPIC NUMBER FIFTY		፶	&#x1376
ETHIOPIC NUMBER SIXTY		፷	&#x1377
ETHIOPIC NUMBER SEVENTY		፸	&#x1378
ETHIOPIC NUMBER EIGHTY		፹	&#x1379
ETHIOPIC NUMBER NINETY		፺	&#x137A
ETHIOPIC NUMBER HUNDRED		፻	&#x137B
ETHIOPIC NO. TEN THOUSAND		፼	&#x137C

Figure 5. Sample Amharic Unicode Characters & their value

Note: Complete list of Amharic Unicode value are found at [10]

To write Amharic text in the system, the JavaScript code should map all Amharic Unicode value (Amharic Alphabet) with phonetic Amharic keyboard combination.

2.2.2 Ethiopic Script and Syllable

The Ethiopic script developed from the script of Ethiopia's classical language, Ge'ez, which was derived from the Sabaean/Minean script. The script used to write

Ge'ez has been in use since at least the 4th century AD. At first the script represented only consonants. Vowel indication was added in around 350 AD when Christianity arrived in Ethiopia. Whether there is any connection between these two events is not known [9]

Notable Features of Ethiopic Script

- Written horizontally from left to right.
- Each symbol represents a syllable consisting of a consonant plus a vowel. The basic signs are modified in a number of different ways to indicate the various vowels.
- In inscriptions, words were separated with a vertical line. Elsewhere, two dots similar to a colon are used to separate words, though in languages such as Amharic blank spaces are generally used instead.
- A full stop or period is four dots (::) and a comma is two dots with horizontal lines over and between them.

The sounds of Amharic

The sound of Amharic helps to include pronunciation information to the dictionary, and there is no standard of writing Amharic sounds and many of the dictionaries are using their own Amharic sound combination.

Once the sound of Amharic is defined, help page should be defined to help the users how to pronounce Amharic words otherwise it is difficult to use pronunciation information that is attached with the word.

EthioTran[9] defines the following Amharic sounds.

Consonants

ሀ	ለ	ሐ	መ	ሠ	ረ	ሰ	ሸ	ቀ	ቈ	በ	ተ	ቸ	ኀ	ኈ	ነ	ኘ	አ
h	l	ḥ	m	ś	r	s	š	q	qu	b	t	č	ḫ	ḫu	n	ñ	ʾ
[h]	[l]	[h]	[m]	[s]	[r]	[s]	[ʃ]	[k']	[kʷ]	[b]	[t]	[ʧ]	[h]	[hʷ]	[n]	[ɲ]	[ʔ]

ከ	ኸ	ወ	ዐ	ዘ	ዠ	የ	ደ	ጀ	ገ	ጐ	ጠ	ጨ	ጰ	ጸ	ፀ	ፈ	ፐ
k	h	w	ʿ	z	ž	y	d	ǧ	g	gu	ṭ	č	ṗ	ṣ	ẓ	f	p
[k]	[h]	[w]	[ʔ]	[z]	[ʒ]	[j]	[d]	[ʤ]	[g]	[gʷ]	[tʼ]	[ʧʼ]	[pʼ]	[ts]	[ts]	[f]	[p]

Vowels

ä	u	i	a	e	ə	o
[ɜ]	[u]	[i]	[a, ʌ]	[e, ɛ]	[ə]	o, ɔ]

	hoy		läwe		ḥäwt		may		śäwt		rəʾs		sat		š		qaf		qaf	
ሀ	hä	ለ	lä	ሐ	ḥä	መ	mä	ሠ	śä	ረ	rä	ሰ	sä	ሸ	šä	ቀ	qä	ቈ	quä	
ሁ	hu	ሉ	lu	ሑ	ḥu	መ	mu	ሡ	śu	ሩ	ru	ሱ	su	ሹ	šu	ቁ	qu			
ሂ	hi	ሊ	li	ሒ	ḥi	ሚ	mi	ሢ	śi	ሪ	ri	ሲ	si	ሺ	ši	ቂ	qi	ቍ	qui	
ሃ	ha	ላ	la	ሓ	ḥa	ማ	ma	ሣ	śa	ራ	ra	ሳ	sa	ሻ	ša	ቃ	qa	ቋ	qua	
ሄ	he	ሌ	le	ሔ	ḥe	ሜ	me	ሤ	śe	ሬ	re	ሴ	se	ሼ	še	ቄ	qe	ቌ	que	
ህ	hə/ø	ል	lə/ø	ሕ	ḥə/ø	ም	mə/ø	ሥ	śə/ø	ር	rə/ø	ስ	sə/ø	ሽ	šə/ø	ቅ	qə/ø	ቍ	quə	
ሆ	ho	ሎ	lo	ሖ	ḥo	ሞ	mo	ሦ	śo	ሮ	ro	ሶ	so	ሾ	šo	ቆ	qo			

	bet		täwe		č		ḫärm		nähas		ñ		ʾälf		kaf	
በ	bä	ተ	tä	ቸ	čä	ኀ	ḫä	ኈ	ḫuä	ነ	nä	ኘ	ñä	አ	ʾä	ከ	kä	ኰ	kuä	
ቡ	bu	ቱ	tu	ቹ	ču	ኁ	ḫu			ኑ	nu	ኙ	ñu	ኡ	ʾu	ኩ	ku			
ቢ	bi	ቲ	ti	ቺ	či	ኂ	ḫi	ኌ	ḫui	ኒ	ni	ኚ	ñi	ኢ	ʾi	ኪ	ki	ኲ	kui	
ባ	ba	ታ	ta	ቻ	ča	ኃ	ḫa	ኋ	ḫua	ና	na	ኛ	ña	ኣ	ʾa	ካ	ka	ኳ	kua	
ቤ	be	ቴ	te	ቼ	če	ኄ	ḫe	ኊ	ḫue	ኔ	ne	ኜ	ñe	ኤ	ʾe	ኬ	ke	ኴ	kue	
ብ	bə/ø	ት	tə/ø	ች	čə/ø	ኅ	ḫə/ø	ኍ	ḫuə	ን	nə/ø	ኝ	ñə/ø	እ	ʾə/ø	ክ	kə/ø	ኵ	kuə	
ቦ	bo	ቶ	to	ቾ	čo	ኆ	ḫo			ኖ	no	ኞ	ño	ኦ	ʾo	ኮ	ko			

Figure 6: The sounds of Amharic

In [10], the sound of Amharic sound is defined in the following ways

HA
HU
HI
HAA
HEE
HE
HO

2.2.3 Database Implementation of UniCode Values

For database implementation of Unicode values, there must be a conversion function that converts the Amharic text into its equivalent Unicode value and makes the database to understand it.

There are several English - Amharic dictionary and Learn Amharic software but its content and their implementation style is hidden unless it is purchased online or from the available department store. They put only the description about the software.

In general, most online dictionaries:

- have the functionality of translating from English to Amharic only, not from Amharic - English
    - the famous book of **English-Amharic dictionary** [5] by Amsalu Aklilu and G. P. Mosback that is available in the market for long time, is also available online. It does the translation only in one way i.e. from English to Amharic and also
- displays the meaning of the word using text, no multimedia content at all.
- don't support Amharic scripts

3. System Analysis

3.1 Current System

One means of getting translation of English to Amharic word is using the print versions of Amharic dictionary.

Using the printed version of the dictionary has been only means of getting word translation, but now it is not convenient means especially in today's rashly world and due to the above mentioned drawbacks.

The other means of getting word translation is using online dictionaries. Online dictionaries solve some of the drawbacks of the printed version dictionaries.

One of the limitation of online dictionaries is one has to have a computer and an internet connection other wise it is unable to do the translation of the word.

Despite the fact that online dictionaries are better means of getting the translation of the word than the printed version of dictionaries, most of the online Amharic dictionaries have the following draw backs

- do the translation from English to Amharic only, not from Amharic to English.
- Display the meaning of the word only using text. For Amharic words that are unique to Ethiopia and for those words that are the same in writing but different in meaning, text is not sufficient to describe what it means. Therefore, text based meaning of the word is difficult to understand especially for non-Ethiopians or non-Amharic speakers.
- no multimedia support

Even if the existing online dictionaries have some drawbacks, it solves the problem existed in the printed version of dictionaries. The online dictionary is convenient to many people since it is accessible any time any where, many people would like to use online dictionary.

This project tries to solve the above mentioned problems of an online dictionary and to make more convenient to the users.

3.2 Proposed System

3.2.1 Overview

The proposed system of an online Amharic-English, English Amharic multimedia dictionary will be available at any time any where and display the meaning of the word with multimedia content.

The proposed system will

1. do the translation of the word from English to Amharic and vice versa.
2. have different means of searching the word.
3. uses different multimedia contents to display the meaning of the word.
4. display the meaning of the word with an attached audio for both Amharic and English words to helps the users how to pronounce the word correctly.
5. available online.
6. display Amharic text with out installing Amharic software or downloading an Amharic font; the system has its own embedded Amharic font that makes Amharic words to be displayed in any computer.
7. have an Amharic typing software/script that helps the users to write Amharic words. Users can write Amharic with out installing Amharic software.
8. has an easy interface to make a search
9. interact with the dictionary users to post requests the word to be added, suggestion/comment on the meaning of the word , post a new word , better meaning or an additional meaning.
10. have an administration page to manage dictionary data, users, request and security of the system.
11. have an administration page to manage major dictionary functionalities that include view, add, edit, delete, publish dictionary data, manage additional meaning, manage users, manage privilege, manage conjugation, mange security, manage word category and manage users request.

3.2.2 Functional Requirements

Functional requirements describe the interactions between the system and the users independent of its implementation. The functional requirements of the proposed system are listed as follows

- The system should provide searching facility
- The system should have the facility to navigate from Amharic to English or from English to Amharic search type.
- The system should provide the facility to write both Amharic and English word.
- The system should display the result of the search with multimedia content.
- The system should provide the facility to write users request and comment
- The system should provide validation mechanism for users input.
- The system should notify the users if the word is not available in the database and an appropriate suggestion.
- The system should provide an appropriate error message.
- The system should authenticate the user and assign privilege according to the assigned rights/roles.
- The system should deny access if the users supplies wrong username or password.
- The system should provide the facility to change password
- The system should provide the facility to create, edit and delete user account.
- The system should give the facility to view detail information, add, edit, and delete dictionary data.
- The system should check all the supplied user input - field validation.
- The system should check validity of the type and size of multimedia content.
- The system should register the time, date and by whom the data is registered.
- The system should provide the facility to view, add, modify, delete additional meaning

- The system should provide the facility to add conjugation to Amharic words.
- The system should provide the facility to publish posted data.
- The system should provide the facility to view, add, edit and delete the word category.
- The system should have the facility to view, edit, and delete users request and comment.
- The system should have the facility to control the published and unpublished data.
- The system should display the list of data posted by the specific users.
- The system should have the facility to view, add, edit and delete security page.
- The system should the facility to assign privilege to the users.
- The system should display the administration page menu according to the assigned user's right.

Therefore, in general the system shall,

- have online search facilities
- have facility to post users request and comment
- authenticate and authorize users according to the assigned users' right.
- enable users to manage and validate dictionary date.
- enable administrator to administer posted data
- enable administrator to manage the users.
- have facility to response to the users request

3.2.3 Non-functional Requirements

Nonfunctional requirements describe user-visible aspects of the system that are not directly related with the functional behavior of the system that are listed above. Non functional requirements of the proposed systems are listed as follows

User interface and Human factors

The system interface shall be able to run on any web browser preferably supporting both JavaScript and VB script. The system shall have a user

friendly menu driven interface that is easy to navigate with. The greatest degree of user system interface consistency and standard shall be provided for all user interfaces.

Documentation

The activities and outputs of each system development stage in the project workflow shall be properly documented for the successful completion of the project. These documents produced at the end of each stage shall be organized and compiled together at the end of the project for future reference, system maintenance, and system support.

Hardware/Software consideration

The system is expected to be implemented using ASP.net, therefore it has to be loaded on a web server with **IIS** service and support **.Net Framework**. And at the user side the computer shall have a web browser preferably supporting JavaScript and VB script.

The software requirement mainly impacts the hardware, especially the web server. Therefore, the server shall comply with the HCL of the Windows 2000 operating system technology.

Performance characteristics

The system shall perform operations with in a minimum amount of time. The system shall be interactive and the user shall get the expected result with in few seconds or the system must enable users to do something else while fetching result for their request to hide response latency.

Security Issues

Administration part of the system is accessible only for the authorized Dictionary Administration staff according to their assigned right. An Unauthorized user should not access the administration page by any means. Only few pages of the system are accessible to the users that help to do the search facility and to interact with the Administrator. Some of the resources and activity shall be restricted and allowed only for authorised users. Therefore

the system shall have user identification method and it should grant or deny access to users as per the privilege of the prospective user.

Training

After the completion of the development of this system, training shall be conducted system administrator and data encoders on how to use the system effectively and efficiently. The training shall be given to the system administrator, administration staffs on the method of maintaining and managing information.

Reliability

The system shall be uploaded on a reliable machine and there shall be a frequent and full backup mechanism to avoid any information loss and inconsistency.

3.3 Analysis Model

The aim of analysis model is to produce a model of the system that is correct, complete, consistent, unambiguous, realistic, and verifiable. In this section the system is described by showing its subsystems' functionality by use cases, its static behavior by class diagram and activity diagram, and its dynamic behavior through sequence diagram.

Identified Actors

Actors	Description
DictionaryUser	A person who use the dictionary
DataEncoder	A person who is authorized to encode data
Administrator	Authorized person to over took the system

3.3.1 Use Case Diagram

Use case diagram is the representation of the functionality of the system. This section presents functionality of the system in terms of actors and use cases. A use case describes a function provided by the system that yields a visible result for an actor. An actor describes any entity that interacts with the system (Dictionary User, DataEncoder or Administrator)

The identified use cases of the system are the following

No	Use Case	Description
1	SearchDictionary	This use case used to search a dictionary term
2	ManagePostRequest	This use case is used to manage the posting of the user request.
3	MaintainUsers	This use case is used to create, modify and delete user's accounts.
4	MantainDictionaryData	This use case is used to create, edit, delete and publish dictionary data. this use case also deals with managing additional meaning and conjugation of dictionary data
5	ManageLogin	This use case is used to the login activity.(Login to the system and change password)
6	ManageSecurityPage	This use case is used to manage security page that serves as the list of privileges assigned to users.
7	ManageAccessPrevilages	This use case is used to manage access privileges

The description of each use cases are listed below

Use Case Name:	**SearchDictionary**
Actors:	Dictionary User
Description:	This use case is used search dictionary words.
Trigger:	When the user wants to search dictionary word
Preconditions:	The *Dictionary Search Page* is displayed
Normal Flow:	1. The user clicks on *"English-Amharic*" / *"Amharic-English*" type of search [Select *English* or *Amharic* menu to see list of dictionary words]. 2. The user enters the word and presses the Search button [double click on the word from the list box] 3. The system validates the data and display the result. 4. If system found the requested word, it display the result plus user interaction menu (Better meaning, Additional meaning and Comment) else the system display a message that the word is not found plus user interaction menu (Request the word to be added, Add the word and Comment) **4.1 Post user request when search result is found.** 1. The user selects one of the user's interaction menu. 2. The system populates *Better meaning Form* or *Additional meaning Form* or *Comment Form* based on user's choice. 3. The user fill the form and presses the *Submit* button 4. The system validates the data and displays confirmation message **Post conditions**: The user request is posted **4.2 Post user request when there is no search result** 1. The user selects one of the user's interaction menu. 2. The system populates *Request the word to be added Form* or *Add new word Form* or *Comment Form* based on user's choice. 3. The user fill the form and presses the *Submit* button 4. The system validates the data and displays confirmation message **Post conditions**: The user request is posted
Post conditions:	search result
Exit Condition:	users select one of users menu or make another search

Use Case Name:	**MaintainPostRequest**
Actors:	Administrator
Description:	This use case is used to view, modify or delete user request.
Trigger:	When the administrator wants to manage user request
Preconditions:	1. The user is successfully logged in 2. The *Dictionary Admin Main Page* is displayed
Normal Flow:	1. The user chooses *"Users Request"* menu item *from Admin Main Menu* 2. The user clicks on *"view user request*" link 3. The system displays user request information 4. The administrator evaluates the user request and presses accept or deny button. **4.1 Modify the request when the administrator press accept button** 1. The system populates *Add new data Form or Additional meaning Form or better meaning Form* based on the user request. 2. Administrator performs the required modification, additional information and presses the *Submit* button 3. The system validates the data and displays confirmation message **Post conditions**: The user request is added to the dictionary knowledge base/Dictionary data table **4.2 Delete the user request when the Administrator press deny button.** 1. The system displays confirmation message **Post conditions**: The user request is deleted permanently
Post conditions:	The user request is screened.
Includes:	Login
Special condition:	If the user request is comment, the administrator will act accordingly.

Use Case Name:	**MaintainUsers**
Actors:	Administrator
Description:	This use case is used to create, modify or delete user accounts.
Trigger:	When the user wants to manage user accounts
Preconditions:	1. The user is successfully logged in 2. The *Dictionary Admin Main Page* is displayed
Normal Flow:	1. The user clicks *"Users"* menu item *from Admin Main Menu* 2. The user clicks on *"Add new user*" link 3. The system displays Add new user Form 4. The user enters full name and account information such as username, password, confirm password, status and presses the *Save* button. 5. The system validates the data and displays confirmation message **2.1 Modify the record of existing User** 1. The user selects an account from the list of available users. 2. The system populates detail user information such as full name, username, type (user/administrator) in the *Edit User Form* 3. The user performs the required modification and presses the *Save* button 4. The system validates the data and displays confirmation message **Post condition**: The record of user account information is modified **2.2 Delete the record of existing User Permanently** 1. The user selects an account from the list of available users. 2. The user presses the *Delete* button 3. The system displays confirmation message **Post conditions**: The record of user account information is deleted permanently
Post condition:	The new user account is saved into the user account file
Includes:	Login
Priority:	High
Frequency of Use:	Once in a week
Business Rules:	1. The administrator can activate or deactivate users account, and reset the password. However, s/he can't view user's password 2. Password cannot be less than 4 characters

Use Case Name:	**MaintainDictionaryData**
Actors:	Data Encoder, Administrator
Description:	This use case is used to create, modify or delete dictionary data.
Trigger:	When the user wants to manage dictionary data.
Preconditions:	1. The user is successfully logged in 2. The *Dictionary Admin Main Page* is displayed
Normal Flow:	1. The user chooses *"English-Amharic" or "Amharic-English"* menu item *from Admin Main Menu* 2. The user clicks on *"Add new Data*" link 3. The system displays Add New Data *Form* 4. The user enters all the required information and presses the *Save* button. 5. The system validates the data and displays confirmation message **2.1 Modify the record of existing data** 1. The user selects a data from the list of dictionary words. 2. The user click on *"Edit data"* button 3. The system populates all posted information in the *Edit Data Form* 4. The user performs the required modification and presses the *Save* button [if the user is administrator, publish button will be displayed and the administrator can publish the data] 5. The system validates the data and displays confirmation message **Post condition**: Dictionary data information is modified **2.2 Delete the record of existing data Permanently** 1. The user selects a data from the list. 2. The user presses the *Delete* button 3. The system displays confirmation message **Post condition**: The record of user account information is deleted permanently

....

	2.3 Add Additional meaning for the existing data 1. The user selects a data from the list. 2. The user click on *View detail* button 3. The system populates detail data information in the Data view detail *Form* 4. The user press on add additional meaning button. 5. The system display "Add additional meaning" form 6. The user fill all the required information and presses *Submit* button 7. The system validates the data and displays confirmation message **Post condition**: Additional meaning is saved. **2.4 Add conjugation to the existing data** 1. The user selects a data from the list. 2. The user click on *View detail* button 3. The system populates detail data information in the Data view detail Form 4. The user press on add conjugation button. 5. The system display "Add Conjugation" form 6. The user fill all the required information and presses *Submit* button 7. The system validates the data and displays confirmation message **Post condition**: Conjugation information is saved.
Post conditions:	The new data is saved into the Dictionary database.
Includes:	Login

Use case Name:	**ManageLogin**
Actors:	User
Description:	This use case is used to manage login users.
Trigger:	When the user wants to login into the system When the user wants to change his/her password
Preconditions:	1. The user is successfully logged in or 2. The *Login Form* is displayed
Normal Flow:	1. The user chooses *Change Password* menu item *from Dictionary Admin menu* 2. The system displays *Change Password Form* populated with user id/username 3. The user enters old password, new password, and password confirmation information and presses the Save button. 4. The system validates the data and saves the changes. **Login to the System** 1. The user enters the url of the Dictionary Admin page 2. The system displays the *Login Form* 3. The user enters the user name and password and presses the *login* button 4. The system validates the data and redirects the user to system functionality set to the user. 5. The Administration main menu is dynamically created according to the assigned access right of the user. **Post condition**: The user is successfully logged in
Post condition:	The password of the user is changes
Includes:	Login

Use case Name:	**ManageSecurityPage**
Actors:	Administrator
Description:	This use case is used to manage security page.
Trigger:	When the user wants to manage security pages that makes the users to navigate to the other pages of the system.
Preconditions:	1. The user is successfully logged in 2. The Dictionary Admin *main Page* is displayed
Normal Flow	1. The user clicks ***Security Page*** menu from the administration main menu. 2. The system displays *list of security page.* 3. The user click on "Add new page" 4. The system displays Add new page Form 5. The users fill title of the page that will be displayed as privilege list and the name of file (.aspx file) and submit the form. 6. The system validates the data and saves. **2.1 Modify the security page** 1. The user selects the page from the list. 2. The system populates detail page information the *Edit Security Page Form* 3. The user performs the required modification and presses the *Save* button 4. The system validates the data and displays confirmation message **Post condition**: The security page information is modified **2.2 Delete the record of existing page Permanently** 1. The user selects the page from the list. 2. The user presses the *Delete* button 3. The system displays confirmation message **Post conditions**: The page is deleted permanently
Post condition:	The security page (access privilege) is saved.
Includes:	Login

Use case Name:	**ManageAccessPrivilege**
Actors:	Administrator
Description:	This use case is used to manage access privileges.
Trigger:	When the user wants to manage access privileges to permit whole or part of an system functionalities
Preconditions:	1. The user is successfully logged in 2. The Dictionary Admin *main Page* is displayed
Normal Flow:	1. The user chooses ***Privilege*** menu from the administration main menu. 2. The system displays *list of users and the list of access privilege.* 3. The user selects a username form the list and select the access right to the user and presses the submit button 4. The system saves the access *privilege*
Post condition:	The new access privilege is saved to the selected user account.
Includes:	Login

Use case Diagram

Use case diagram describes the functionality of the system from the user's point of view.

The following diagram shows the use case diagram of the system.

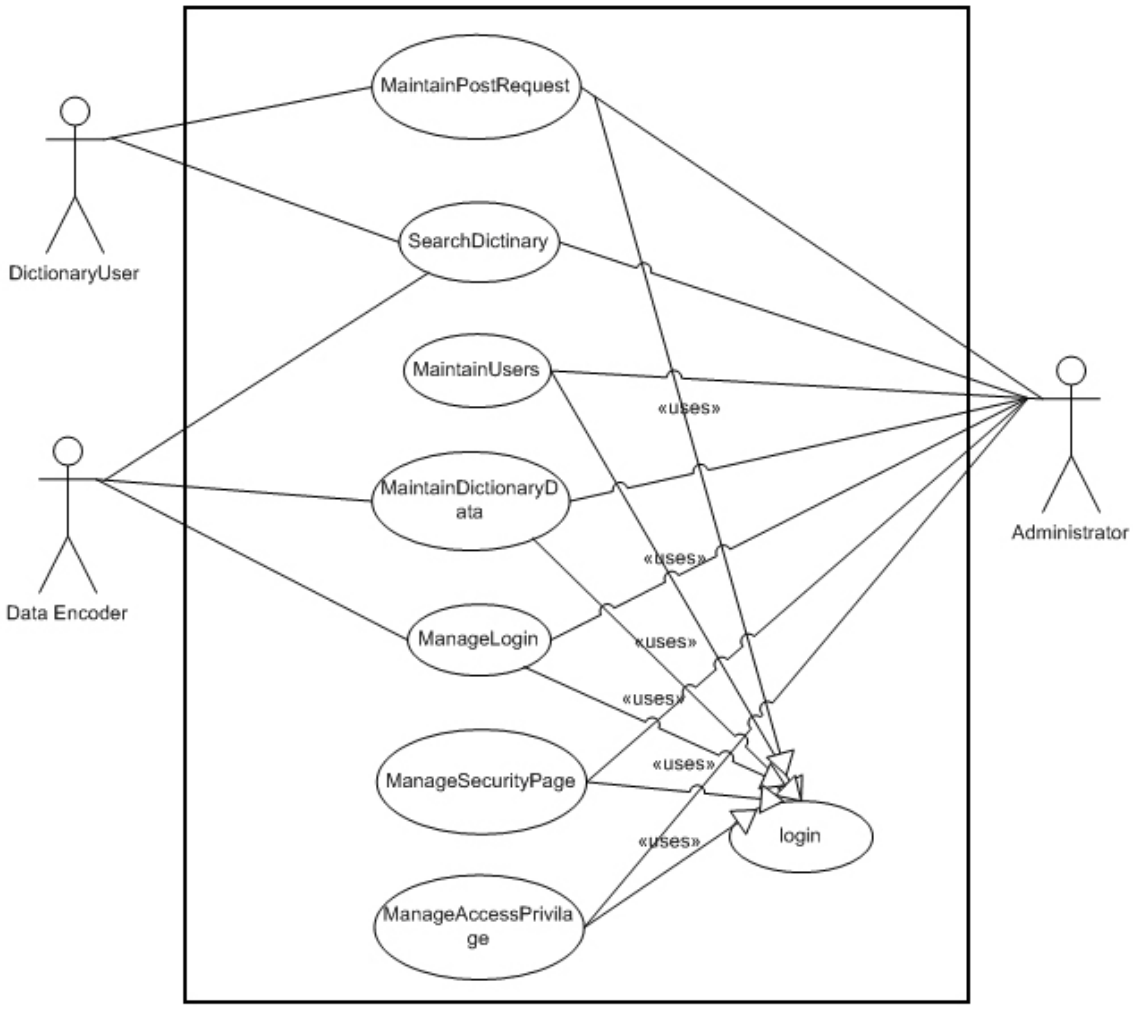

Figure 7: Use case diagram of the system

3.3.2 Sequence Diagram

Sequence diagram describe behavior of the system as a sequence of messages exchanged among a *set of objects*. It is used to formalize the behavior of the system and to visualize the communication among objects of the system. Figure 8-14 depict sequence diagram for the various use cases.

.

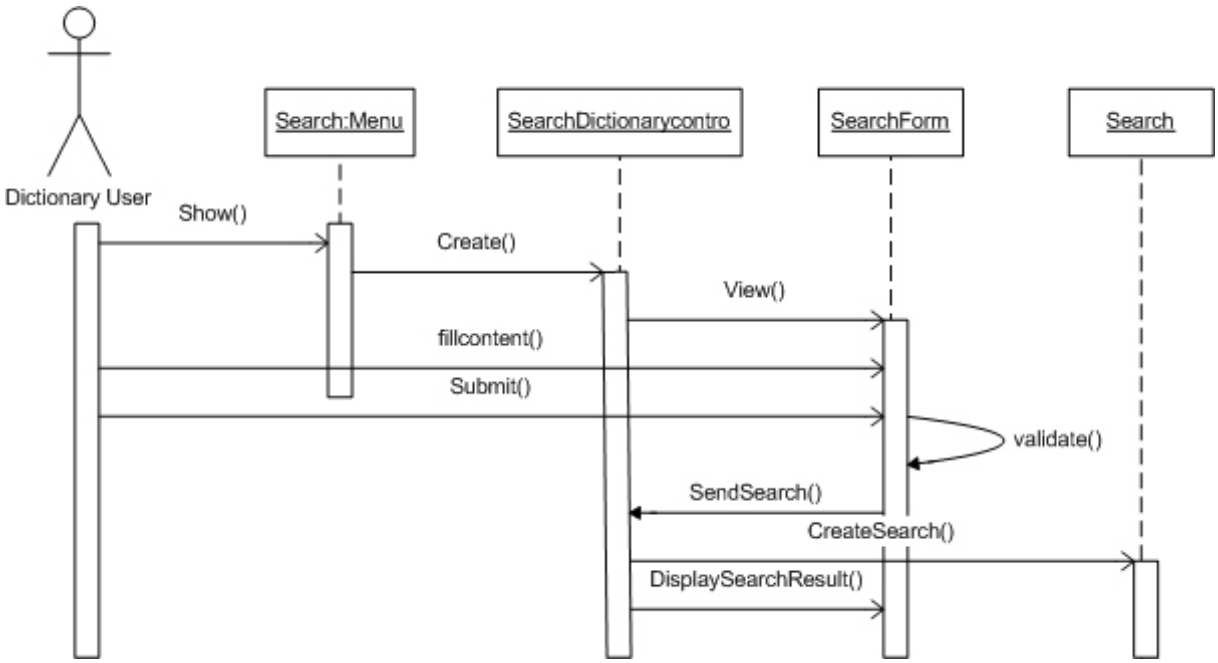

Figure 8: Sequence Diagram for SearchDictionary use case

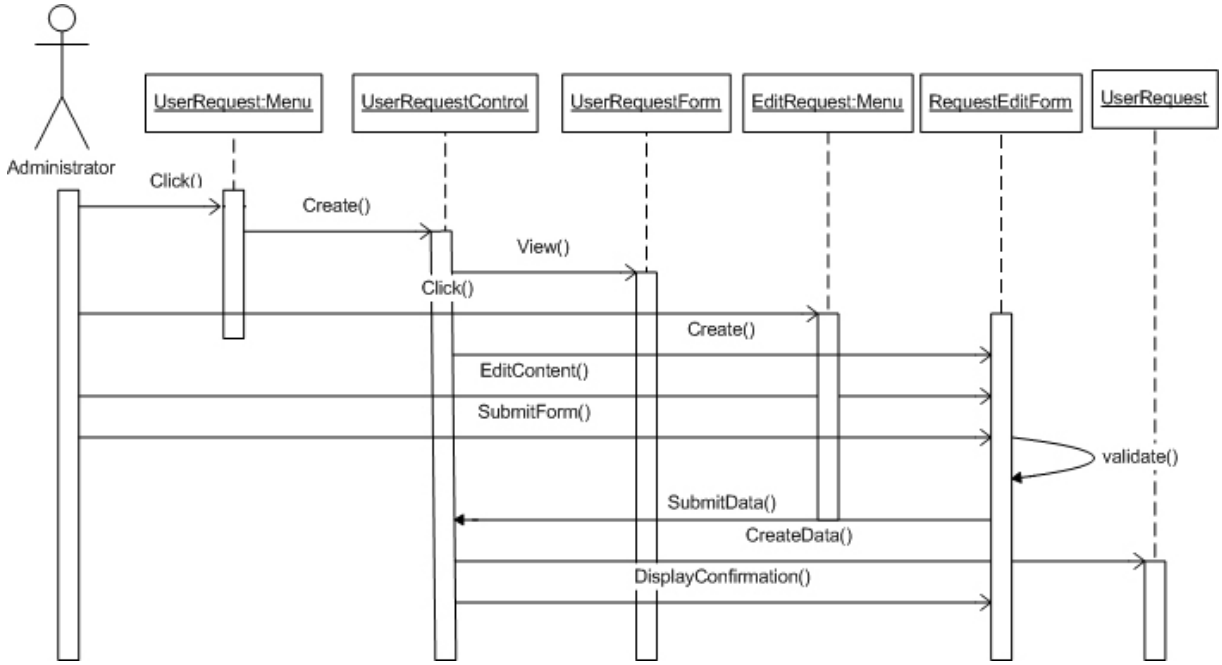

Figure 9: Sequence diagram for ManagePostRequest use case

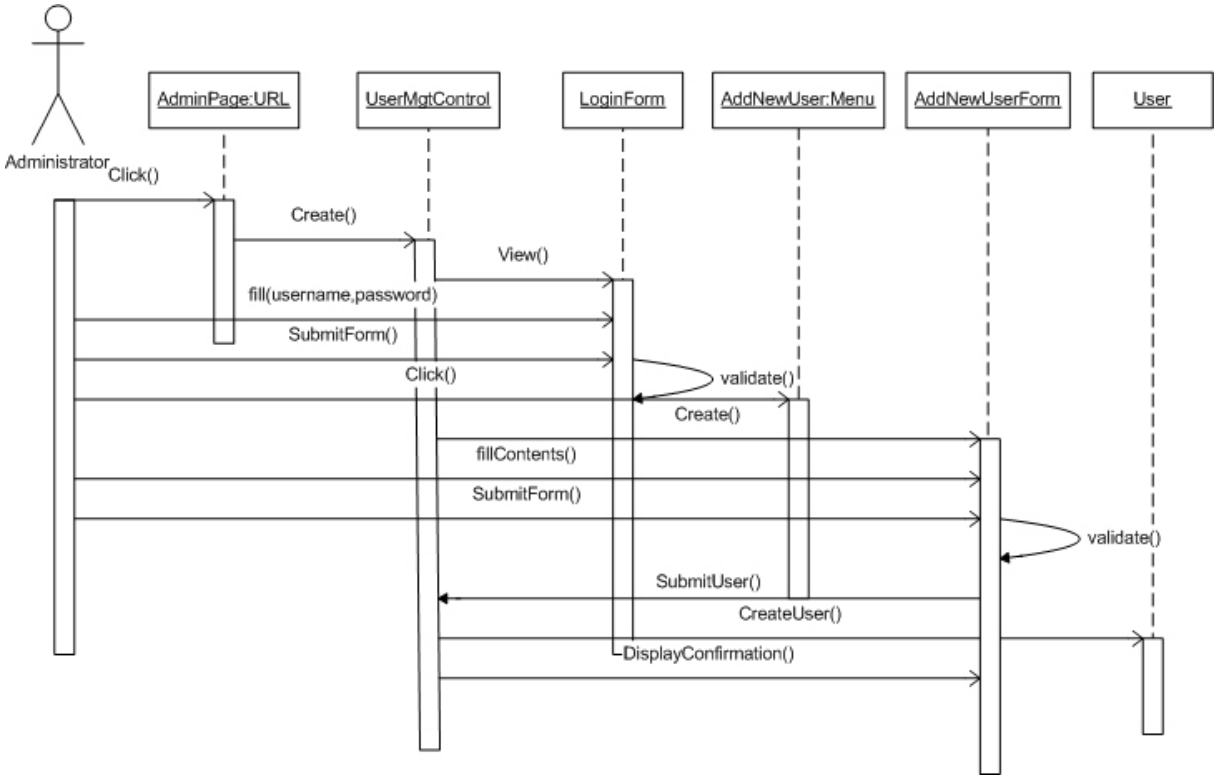

Figure 10: Sequence diagram for MaintainUsers use case

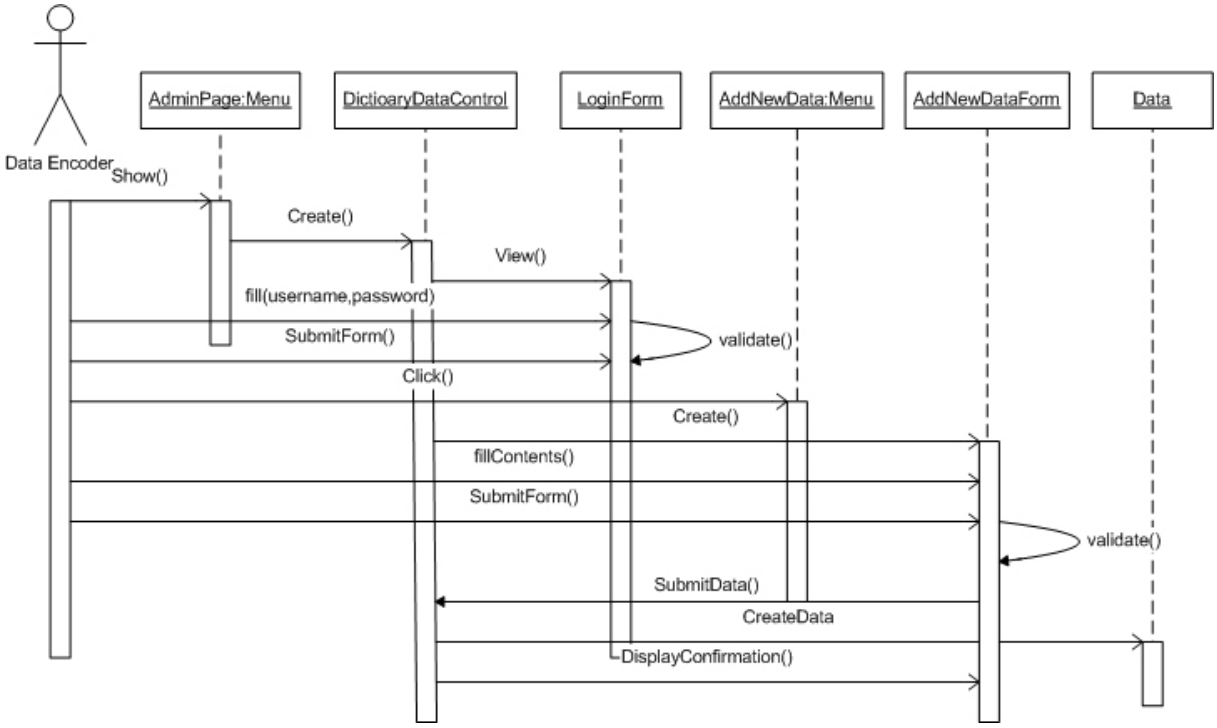

Figure 11: Sequence diagram for MaintainDictionaryData use case

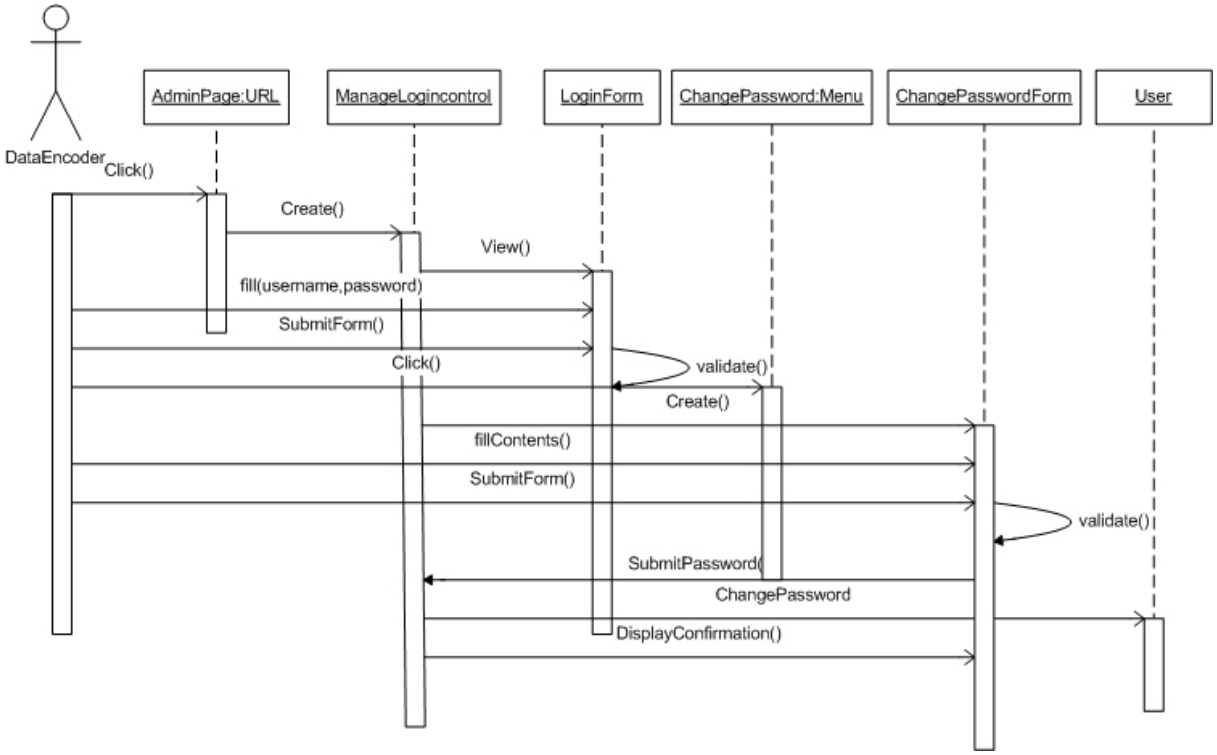

Figure 12: Sequence diagram for ManageLogin use case

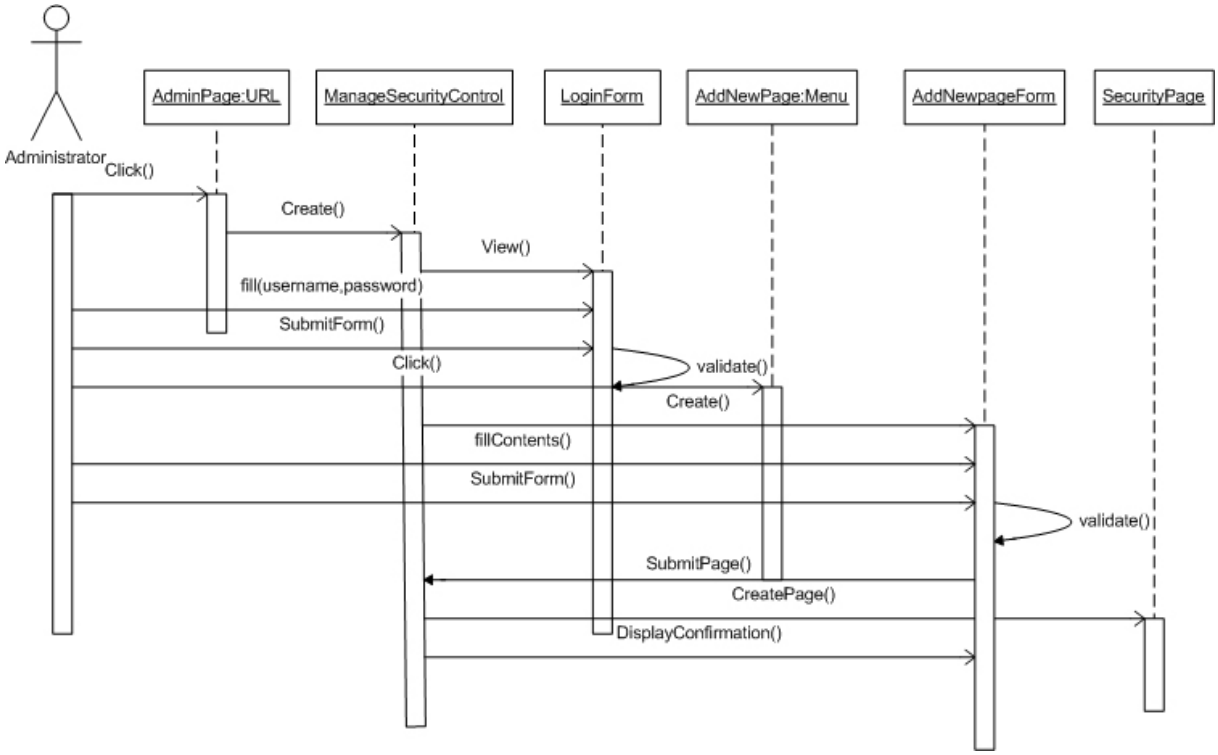

Figure 13: Sequence diagram for ManageSecurityPage use case

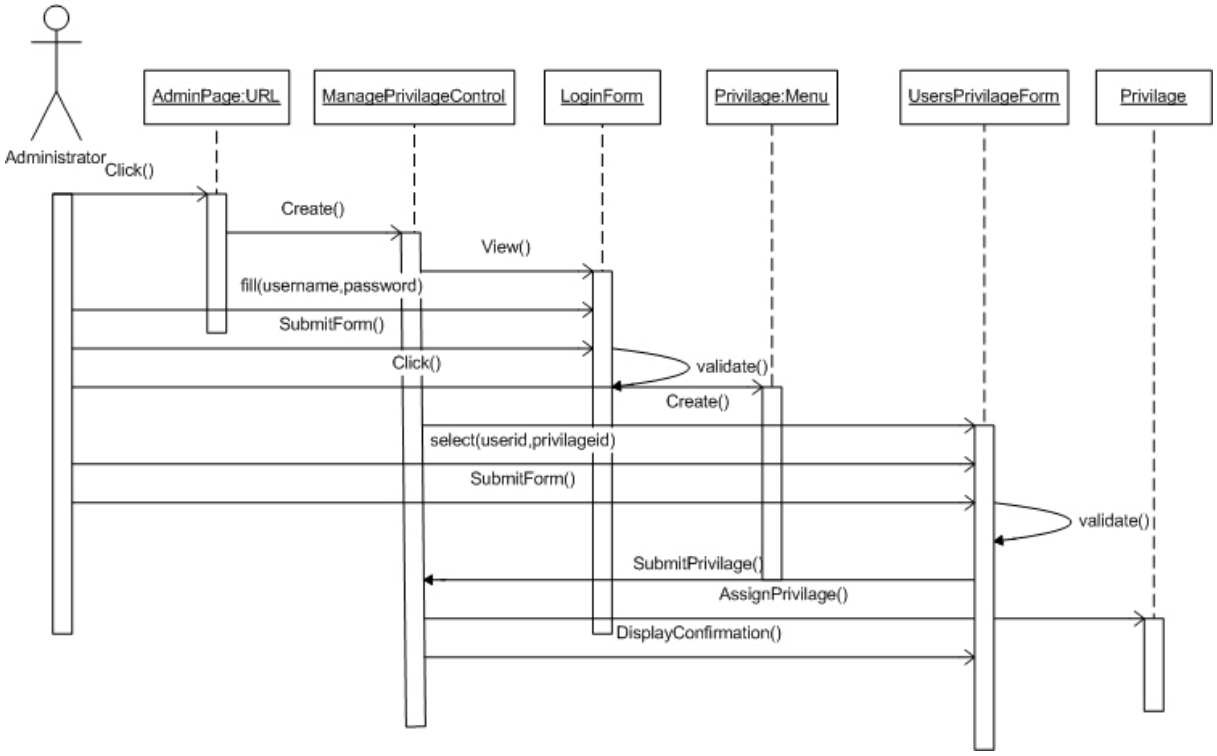

Figure 14: Sequence diagram for ManagerAccessPrevilages use case

3.3.3 Class Diagram

Class diagrams describe the structure of the system in terms of objects, classes, attributes, operations, and their associations. A class is an abstraction in object-oriented programming languages. Like abstract data types, a class encapsulates both attributes and operations. Unlike abstract data types, classes can be defined in terms of other classes by using generalization. Object is an instance of a class. The association denotes the number of links each object can have with another object in the system.

The following class diagram describes the structure of the `DictionarySystem`

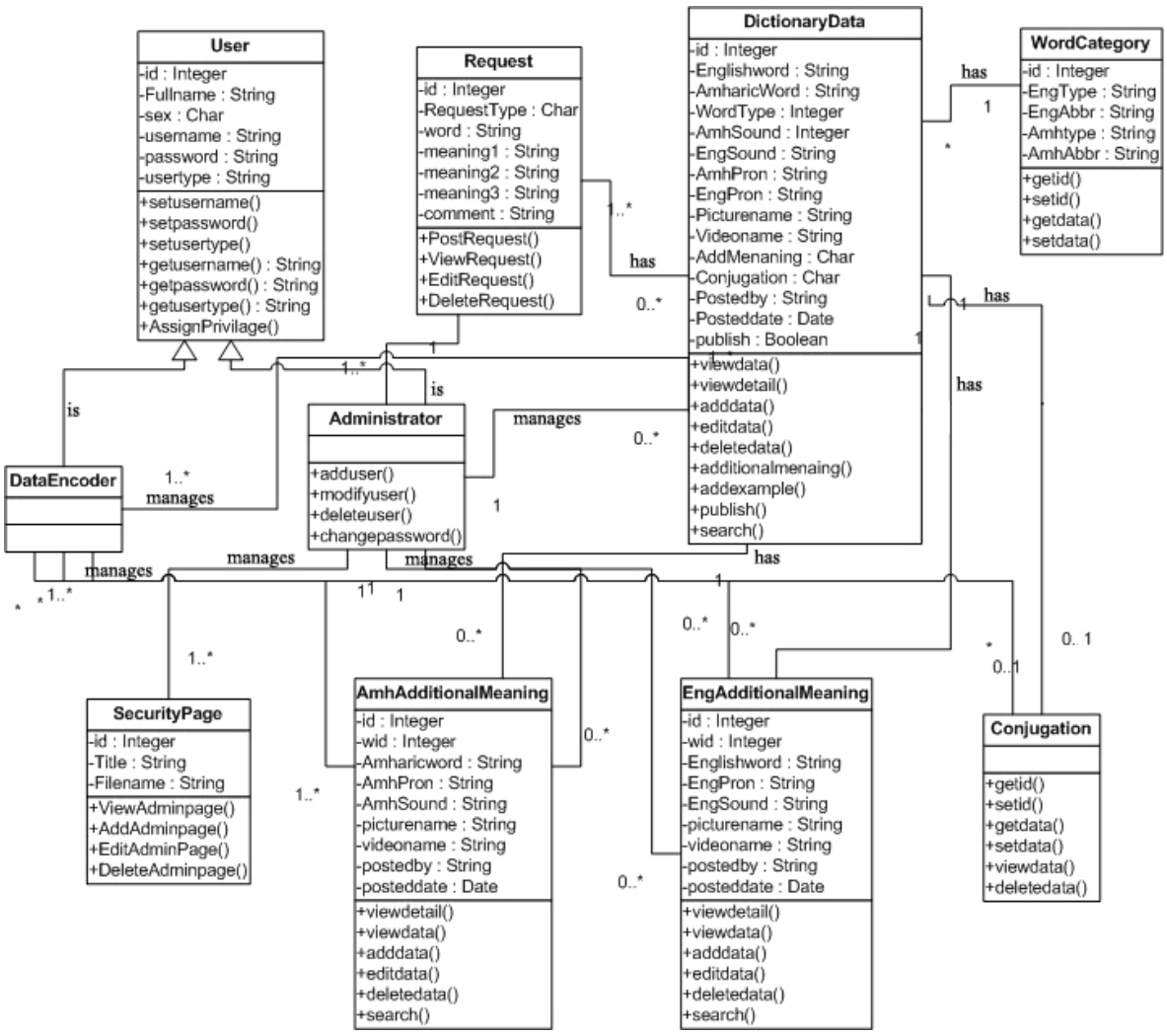

Figure 15: Class diagram of the system

3.3.4 Activity Diagram

An activity diagram describes a system in terms of activities. Activities are states that represent the execution of a set of operations. The completion of these operations triggers a transition to another activity. It is a flow diagram used to represent the data flow or the control flow through a system

Figure 16 is an activity diagram representing activities related to searching `DictioonaryWords.` Rounded rectangles represent activities; arrows represent transitions between activities; thick bars represent the forks of the control flow.

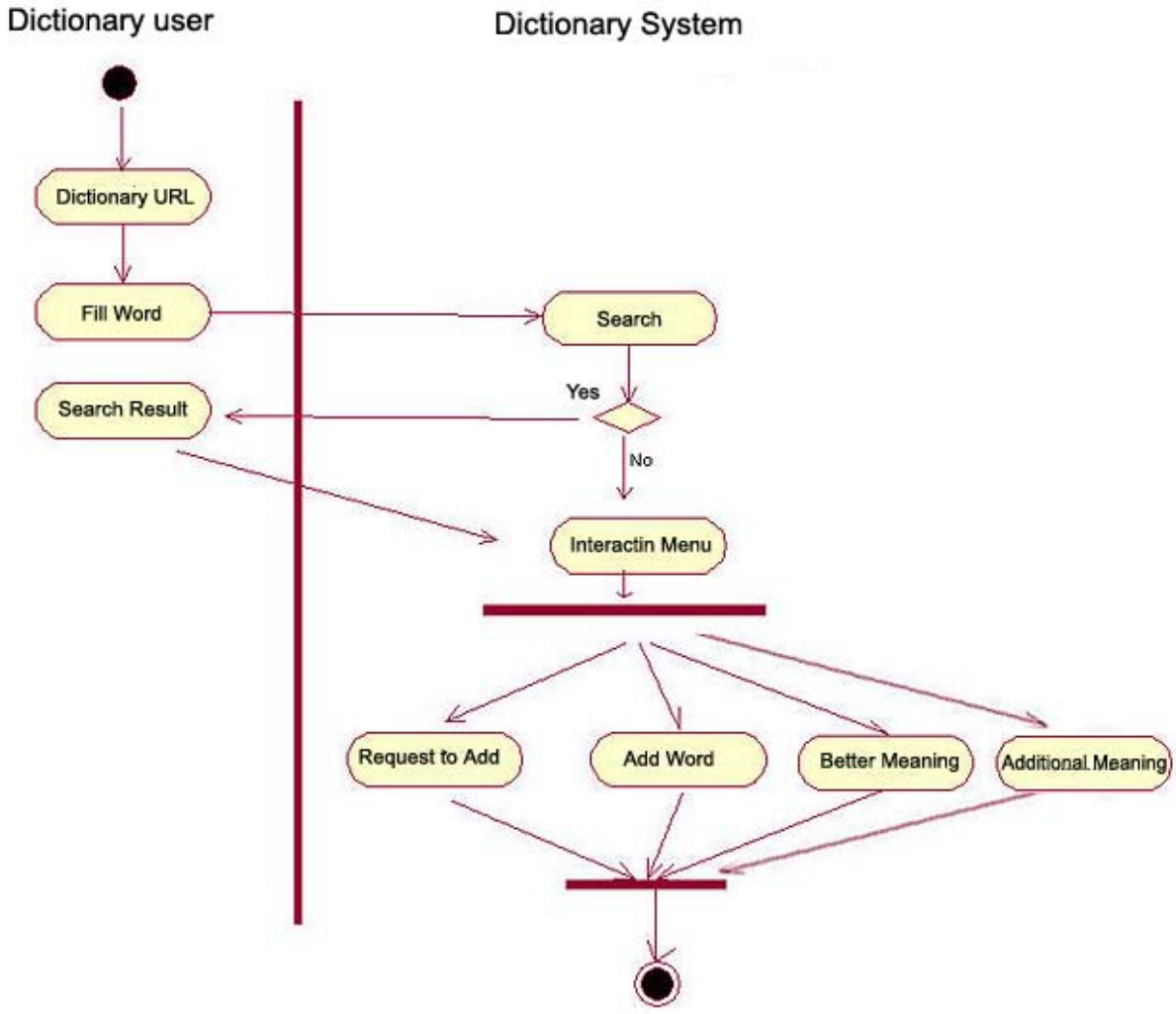

Figure 16: Activity diagram for searching dictionary words

4. System Design

In this section the systems is described by defining the design goals of the project, by decomposing the system into smaller subsystems that can be easily realized (proposed system architecture) and by selecting strategies for building the system, such as the hardware/software platform on which the system will run, the persistent data management strategy, the global control flow, the access control policy, and the handling of boundary conditions. The result of system design is a clear description of each of these strategies, subsystem decomposition, and a deployment diagram representing the hardware/software mapping of the system.

4.1 Design Goals Criteria

The system is expected to run on a web server since it is a system to be accessed online from different end of the world. Therefore, the design goals criteria stated in this section consider this fact to determine the performance, dependability, cost, maintenance and end user criteria.

Performance

Response time: Since the allowed execution time (response time) on any web server is limited coding should follow simple, efficient and fast algorithms. Especially looping statements shall be avoided (if possible) or minimized. For example instead of using loop for retrieving a specific record form a table use SQL statement.

Throughput: As described above the system mainly runs at the web server, therefore the number of tasks it may perform belongs to not only to this system but also to other request coming from other clients.

Memory: Memory is required mainly during a client request and when the server is preparing the response, after that there is no such memory requirement to be dedicated to the system.

Dependability

Robustness: All user input shall be verified and checked for it correctness and completeness before it is passed as a parameter for further execution at the server side, therefore the system can be protected from failures that may occur from invalid user inputs.

Reliability: The system shall be tested after and during development process to verify that the specified services are available on different client platforms. And also it should give consistent and correct output for various type of input it is given.

Availability: The system shall be deployed on a web server that is available 24hrs. Therefore ample care shall be taken when selecting the deployment server.

Fault Tolerance: Error handlers and RecordSet control objects shall be used to handle and tolerate fault that may arise during execution. For example, end of RecordSet shall be checked during traversing.

Security: In order to protect the system from malicious attack the system shall be well protected and secured. Therefore,

1. The Database shall be password protected
2. User name and password shall be used to identify authorized user from unauthorized users
3. User's password shall be stored in the Database in encrypted manner.
4. Each secured page shall be controlled by a session variables so that a secured page will expire if not used for a longer period of time (often 20 minutes). And also this will help to prevent unauthorized access to secured page through URL.

Maintenance

Modifiability: In order to make the system flexible and easy to modify its functionality some of its feature shall be modifiable.

Adaptability: The system shall be implemented by avoiding constant information therefore some features shall be set any time.

Portability: The system shall be developed to run on .NET environment therefore it is not portable to run on any environment which is not .NET enabled.

Readability: Coding shall incorporate significant and clear comments describing each part of the code what it does and what its purpose is.

Traceability of Requirement: The system shall be developed according to the system analysis and the design specification and it shall not incorporate any thing beyond. Otherwise newly incorporated functionality shall be documented.

End User

Usability: The system shall be developed to be easy for user understanding. Especially in developing the user interface it is better to keep the user dictionary in mind.

4.2 Proposed Software Architecture

4.2.1 Overview

English-Amharic, Amharic-English Multimedia Dictionary is an online system that gives dictionary searching facilities and collects requests from different Dictionary users. Therefore the system shall maintain a central data store which can be maintained and accessed from different locations by a number of users. Thus the system mainly follows a special type of repository architecture called client/server architecture.

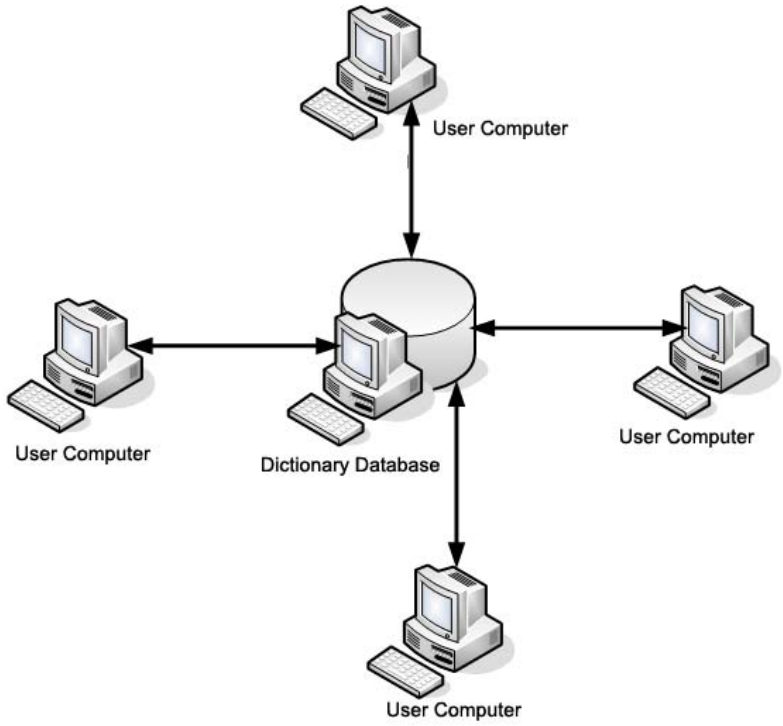

Figure 19: Client Server Architecture

In Online Dictionary, users from different end of the world (as client) will access the service readily available on the server. The services are requested from remote location via the Internet by accessing any of the dynamic page that contains remote procedure and which will be executed remotely by the server. Once the server is done with the execution of the remote page it will return the result to the client (user) who requests the service.

4.2.2 System Architecture

The proposed software architecture is a client-server architecture on which all the operations are performed by the server side. The clients (workstations of actors) request service from the server. The proposed system is decomposed in to four main subsystems, namely, the Search subsystem and the Dictionary subsystem, the Users subsystem and the Security subsystem.

Each subsystem has its own function and there is also communication among them to exchange information. The system is decomposed according to the specialized task it performs to ensure high coherence. The level of coupling is relatively low because the interactions between subsystems are only exchange of data. The major tasks of the dictionary shall be handled by these four subsystems.

Search Subsystem

Search subsystem deals with the automated and online search facilities of the system. It also deals with new information capturing from the users to meet the interactivity of the system. It involves the interaction of the dictionary users for the building of dictionary knowledge base. However the information to be captured by this subsystem is not available online until the administrator examine and publish it and will remains in the system for a given period of time.

Therefore, the search subsystem provide the facilities

- search both Amharic to English and vice versa
- interact with the system - the dictionary users can post a request for a new word to be added, add a new word and its meaning, provide additional or better meaning(s) for the existing words and post comment.
- manage users requests - view, edits and delete user requests by the system administrator.

Dictionary Subsystem

Dictionary is responsible to manage and maintain information for both English-Amharic data and Amharic-English data. It deals with maintaining requirements of the system that is required to add, edit, display, delete and publish dictionary data.

Therefore dictionary subsystem provides the facilities

- maintain dictionary data - the system allow the users to view, add, edit, delete, publish dictionary words
- manage the dictionary data by providing a controlling mechanism for controlling the publication of the newly coined term to the public users
- manage word category - view, add, edit and delete word category/type - i.e. noun, pronoun, adjective etc.

Users Subsystem

Users subsystem is responsible to manage and maintain information about users. Administration users in the system are expected to be authenticated and authorized to access all or part of a system according to the assigned privilege.

Therefore, Users subsystem provides the facilities

- modify users account - access and modify
- maintain users account - view, create, edit, delete user account by the system administrator.

Security Subsystem

Security subsystem deals with the security of the system. It involves creating the list of security pages. Once the list of security page is created, the administrator can assign privilege to the users according to their role.

Therefore Security subsystem facilitates to

- maintain security pages - a way to view, create, edit and delete security page and assign privilege to the users.

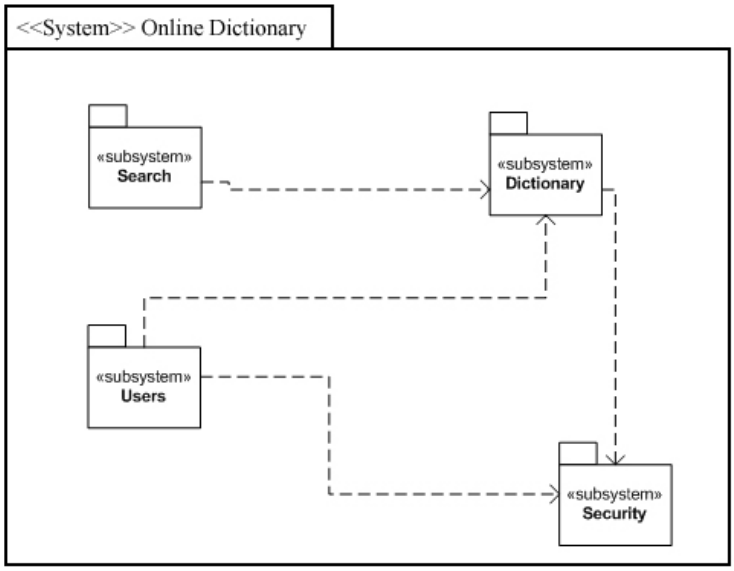

Figure 19: Subsystem decomposition for the Dictionary System
(Folders represent subsystems, dashed lines represent dependencies)

4.2.3 Hardware/Software Mapping

As described earlier in section 4.2.1, client-server architecture is selected to the Online Dictionary system. Therefore, mainly the system will operate at the server side while the output of the system process directed to the server.

Thus, at the client side a web browser that is capable of interpreting basic HTML tags can be used. And the system as well as the database will reside on the web server which is IIS and .NET framework enabled.

The user will operate at the client side and he/she will request a service from the server by accessing a page that may discharge her/his need. And the server will process the server script implemented in ASP.NET that connects to a SQL Server database and produce a set of HTML tags as a response to the client. In due process, the server could access the database to produce the required information to the user/client.

The system will deploy a relational database system, Microsoft SQL that is found to be ideal for client server architecture and capable of handling concurrent record access.

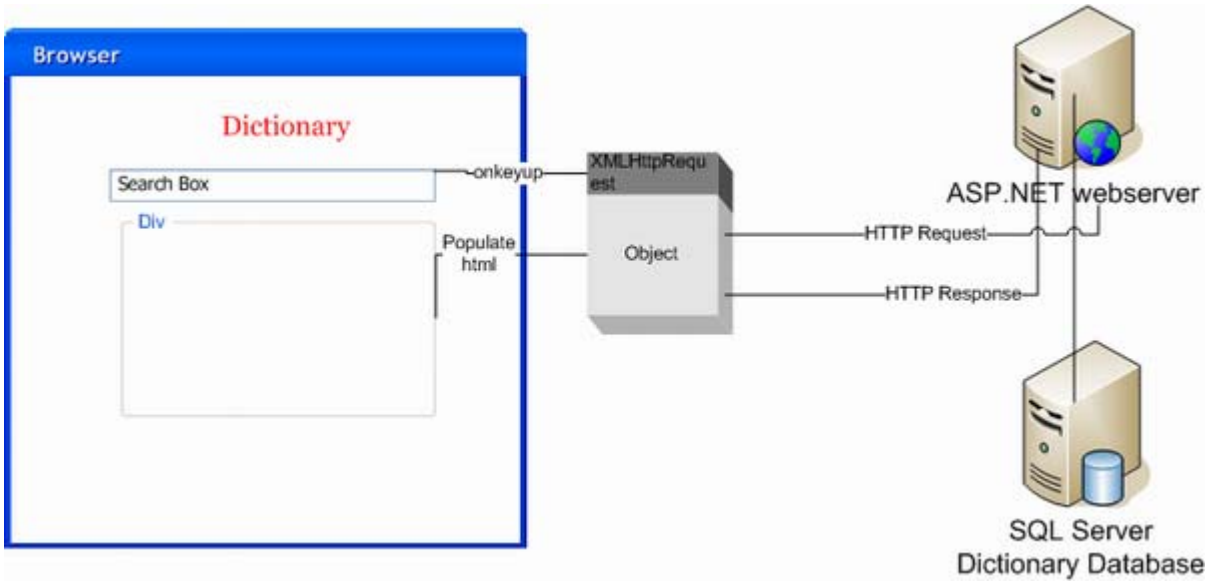

Figure 20: Dictionary System Structure

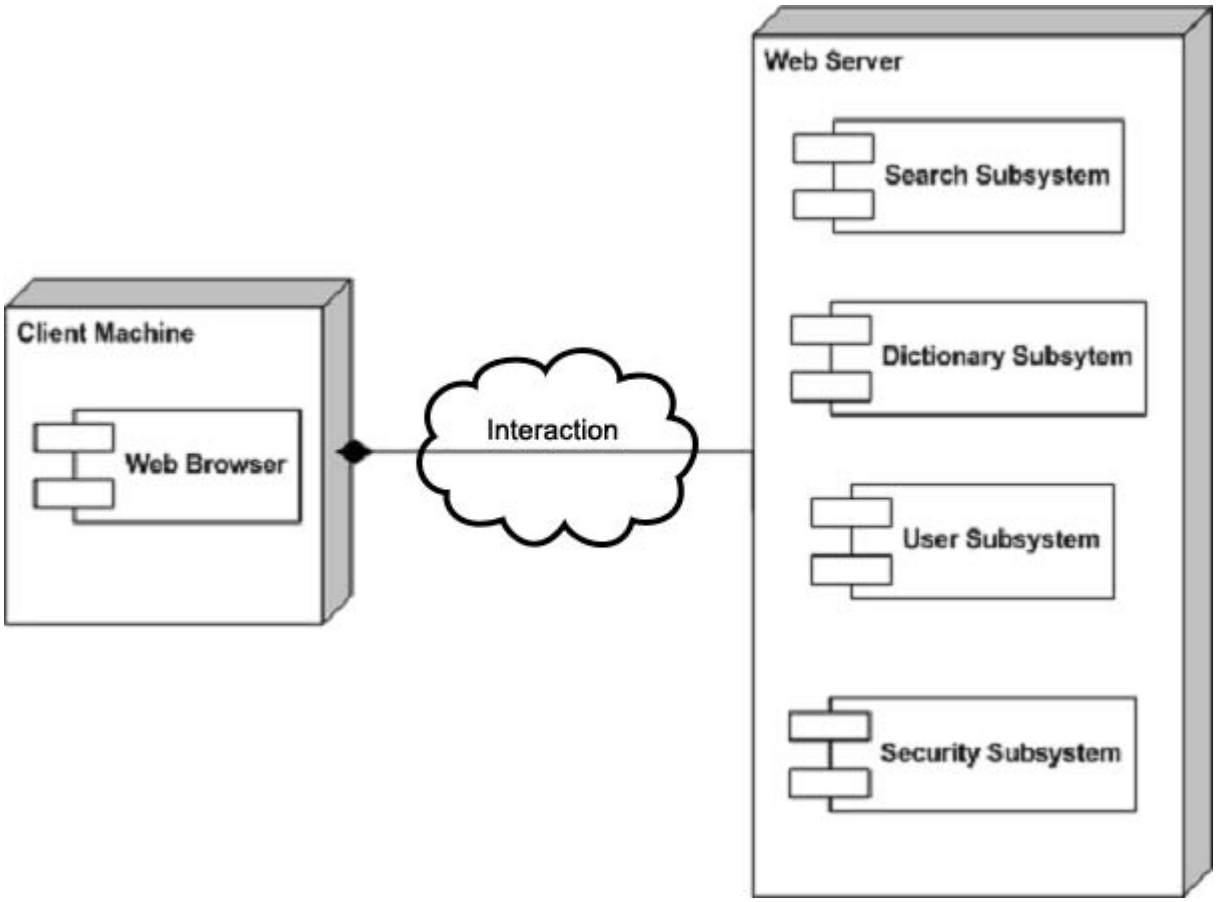

Figure 21: Dictionary deployment diagram

4.2.4 Persistent Data Management

This part of the document depicts the persistent data management rooted from the original class diagram that was produced in analysis phase. In due process, relationships, which are logical but irrelevant to the system are eliminated and those relationships that are important are added.

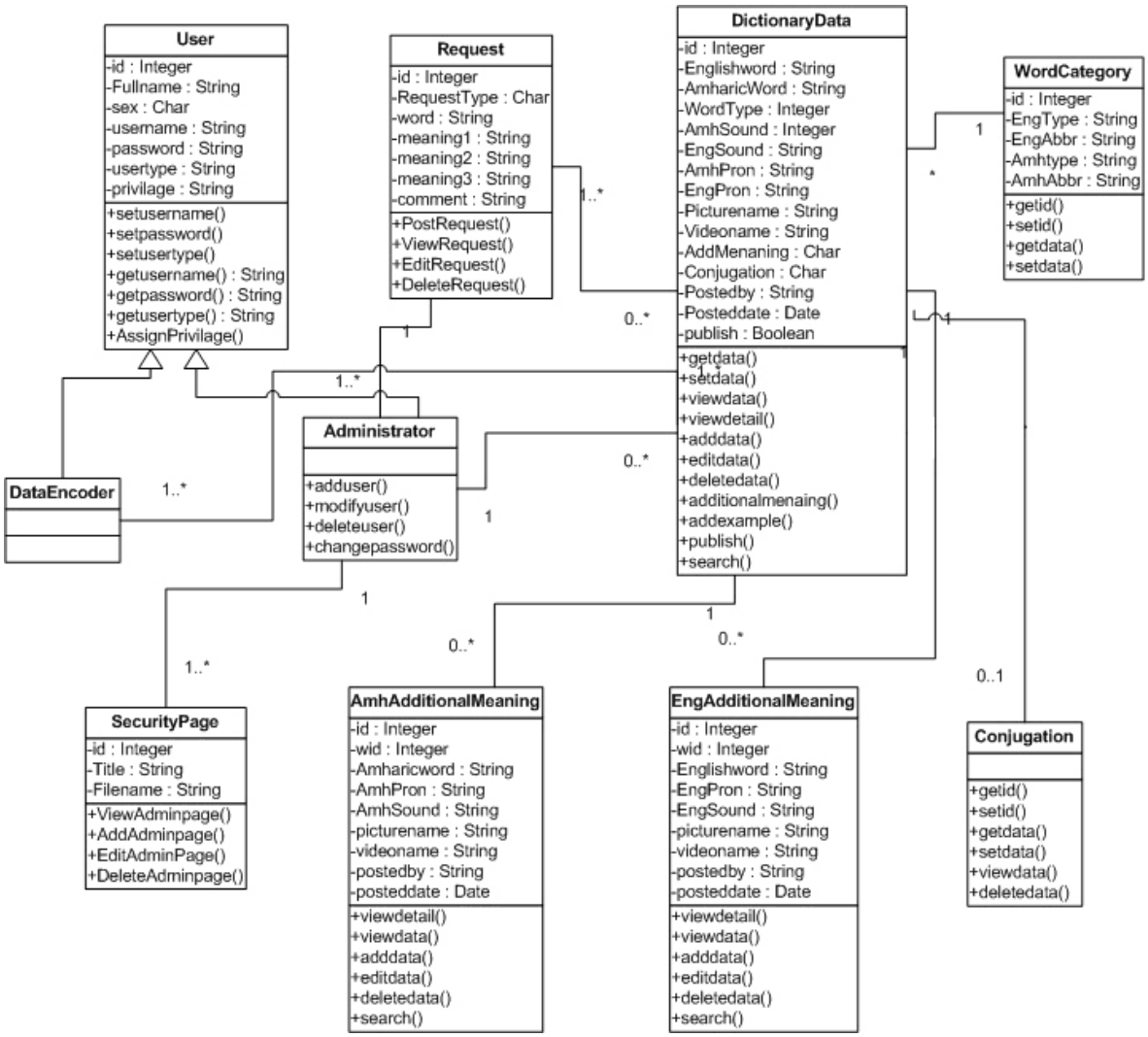

Figure 22: Revised class diagram

Mapping

As described in the “Hardware/Software Mapping” section of this document, Microsoft SQL server is preferred for implementing the database of the system. Therefore, though the design has followed the Object Oriented approach, it is required to shift to relational database model for implementation. Thus, mapping of Object Oriented Model to Relational one is required.

The following table shows result of the mapping process and its effect on the classes and relationships between classes:

Object Model Element	Data Model Element	Results
Attribute	Column	Attributes of classes transform to columns of a table using the same names as the attributes. Attribute types are transformed to appropriate DBMS data types with exception to types that use a domain.
Class	Table	Classes tagged persistent are transformed into tables in the schema of the data model.
Composite Aggregation	Identifying Relationship	Each composite aggregation transforms to an identifying relationship between the appropriate tables in the data model.
Generalization (Inheritance)	Separate Table	Each subclass transforms to a separate table.
Multiplicity	Cardinality	Multiplicity uses the same UML notation as cardinality, however if no multiplicity is designated Data Modeler assigns the table a cardinality of 1.
Operation	Not transformed	

Class Mapping

The classes of the dictionary system mapped in to relational tables in the following ways.

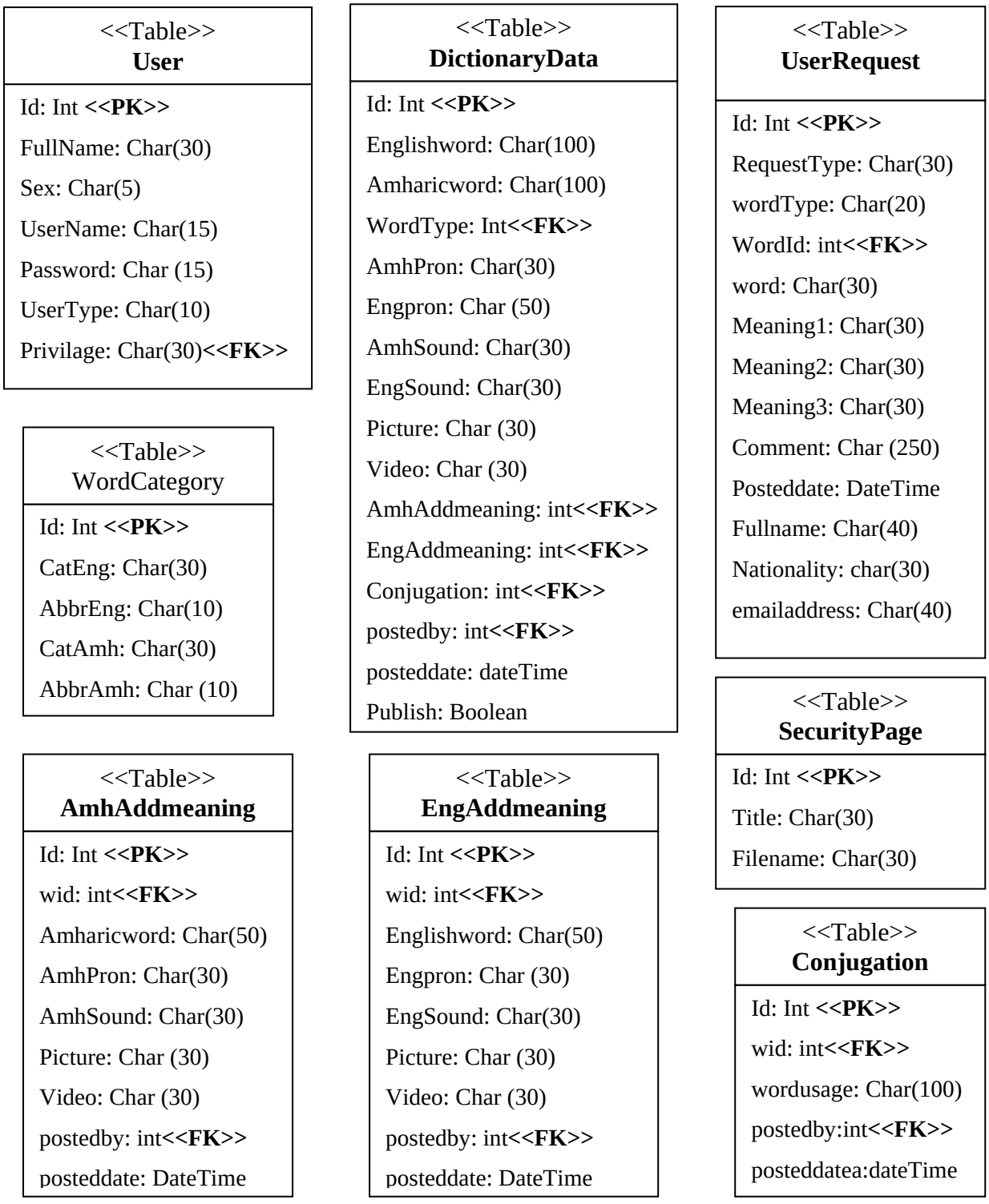

Figure 23: Mapping classes in to Relational tables

Association

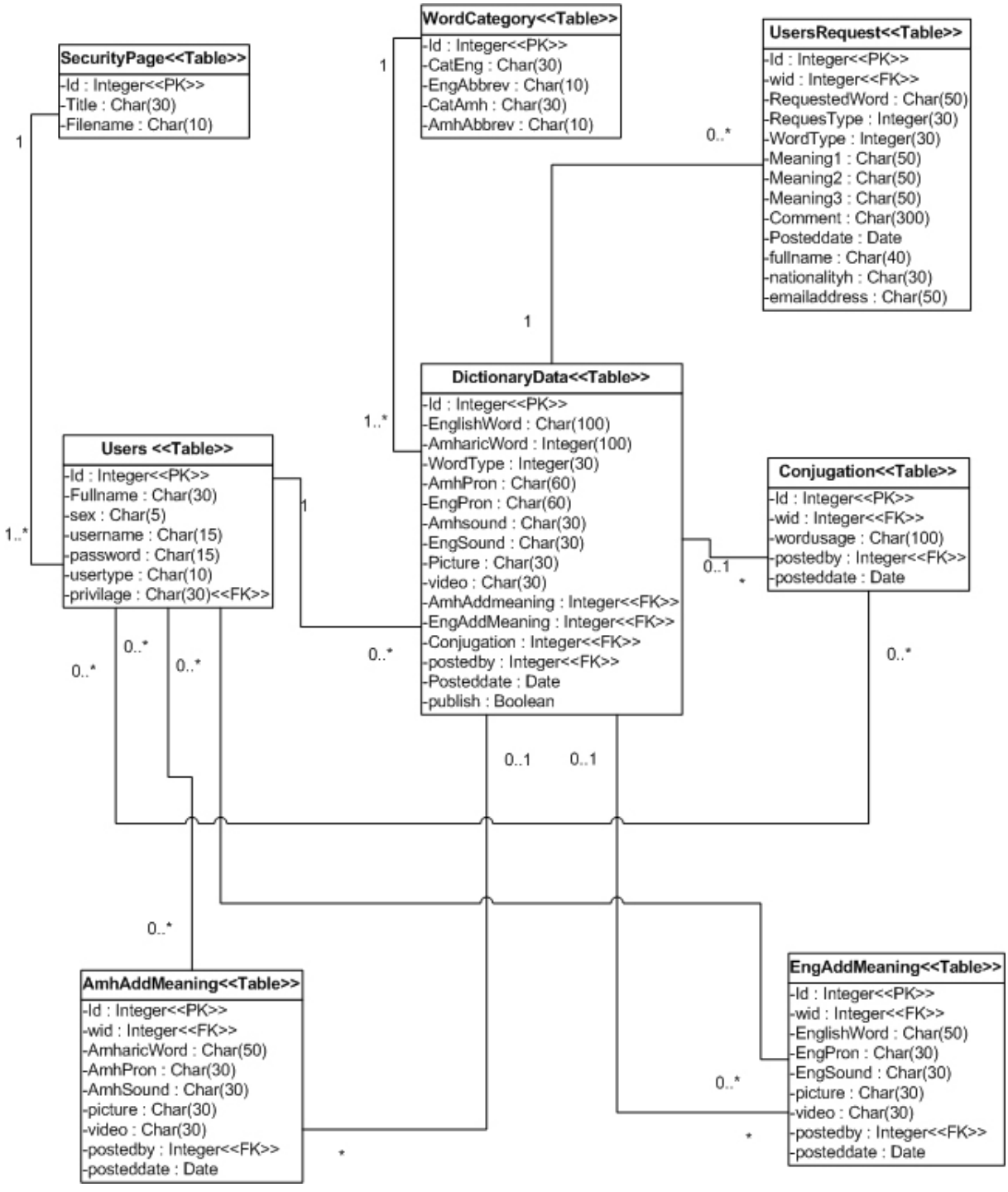

Figure 24: Association among relational tables

4.2.5 Access Control & Security

This section presents descriptions of access control for the system

Actor / Class	Administrator	Data Encoder	Dictionary Users
administrator	SetFullName() SetID() SetSex() GetName() GetID() GetSex() SetUserName() SetPassword() SetUserType() SetPrivilage() GetUserName() GetPassword() GetUserType() GetPrivilage() Login() ModifyAccount() AddUser() DeleteUser() ViewUser()		
DataEncoder	SetFullName() SetID() SetSex() GetName() GetID() GetSex() SetUserName() SetPassword() SetUserType() SetPerivilage() GetUserName() GetPassword() GetUserType() GetPrivilage()		

	ModifyAccount() ModifyAccount() AddUser() DeleteUser() ViewUser()		
Data	ViewData() ViewDetail() AddNewData() EditData() DeleteData() AdditionalMeaning() bettermeaning() Add Cojugation() Publish()	ViewData() ViewDetail() AddNewData() EditData() DeleteData() AdditionalMeaning() bettermeaning() Add Cojugation()	Search()
UsersRequest	ViewRequest() EditRequest() DeleteRequest()		PostRequest()
SecurityPage	ViewAdminPage() AddAdminPage() EditAdminPage() DeleteAdminPage()		
Users	ViewUsers() AddUser() EditUser() DeleteUser() AssignPrivilage() ChangePassword()	changepassword()	
Conjugation	ViewData() ViewDetail() AddNewConjugation() EditData() DeleteData	ViewData() ViewDetail() AddConjugation() EditData() DeleteData()	
WordCategory	Viewdata() addnewcategory() editCategory() deletecategory()		

4.2.6 Global Control Flow

In this section the control flow of the system which is the sequencing of actions in a system is presented. The system mainly uses two type of control flow mechanism namely procedure-driven and event-driven. These two types of control flow and their application in online Dictionary is given in *section 4.2.6.1* and *section 4.2.6.2* and pseudo code is used to describe the flow control of major parts of the system that requires flow control. After that encapsulation of control flow is depicted in *section 2.6.3.*

4.2.6.1 Procedure-driven control

A dynamic page requested by the client from the server uses procedure-driven control flow. In this case the page may collaborate different functions and procedures to populate data from the database or affect record in the database. Pseudo code for some of important procedure-driven control is given below.

Authentication

```
Public connect As New Data.SqlClient.SqlConnection("Data
Source=.\SQLEXPRESS;AttachDbFilename=|DataDirectory|\Dict
ionary.mdf;Integrated Security=True;User Instance=True")
    Public SQL As String
    Public rst As SqlDataReader
    Protected Sub btnlogin_Click(ByVal sender As Object,
ByVal e As System.EventArgs) Handles btnlogin.Click

SQL = "SELECT * FROM users WHERE (uname='" &
txtusername.Text.ToLower & "' AND pwd='" &
txtpassword.Text & "') "

  Dim cmd As New Data.SqlClient.SqlCommand(SQL, connect)

        connect.Open()
        cmd.ExecuteNonQuery()
        rst = cmd.ExecuteReader()
        rst.Read()

        If rst.HasRows Then
            Session("fullname") = rst("Fullname")
            Session("username") = rst("uname")
            Session("uid") = rst("id")
            Session("privilage") =
            (Left(rst("privilage"),
            Len(Trim(rst("privilage"))) - 1))
            Response.Redirect("AdminPage.aspx")
        Else
```

```
                lbluserlogin.Text = "Wrong username or
        password"
                End If
```

Assigning Privilege

```
If Session("fullname") = "" Then
            Response.Redirect("login.aspx")
        End If

        SQL = "SELECT * FROM AdminPages where id IN (" &
        Session("privilage") & ") order by pagetitle"

        Dim cmd As New Data.SqlClient.SqlCommand(SQL, connect)
        Session.Timeout = 10
        connect.Open()
        'cmd1.ExecuteNonQuery()
        rst = cmd.ExecuteReader()

        If rst.HasRows Then
            Dim tRow As New TableRow
            Dim tCell As New TableCell

            tCell.Text = "<ul class='sidemenu'>"
            While rst.Read
                tCell.Text += "<li><a href=" & rst("pagename")
                & ">" & rst("pagetitle") & "</a> </li>"
            End While
            tCell.Text += "<li><a href='logout.aspx'>Log
             Out</a></li>"
            tCell.Text += "</ul>"

            tRow.Cells.Add(tCell)

            tblleftmenu.Rows.Add(tRow)


        End If
```

Client respond

```
      If RdoType.SelectedItem.Value = "Amharic" Then

      ' Do something

      Else

      'Do another thing

      End If
```

4.2.6.2 Event-driven control

In order to save communication cost and minimize server's overload, checking for completeness of required fields and verifying their validity (for example email address shall be in the form *text@text.text* character) has to be done at client side. And such operation uses event-driven control flow. In this case a code will waits for an external event and when the event occurs, it will be dispatched to the appropriate object based on information associated with the event.

```
/* … */

Sub SubmitButton_Click()
        For Each CompulsoryControl in Page
          If (CompulsoryControl.Value = "") then
                strMsg = "CompulsoryControl.Name & "Can not be empty"
                Exit Sub
          End If
        Next
End Sub

/* …   */
```

4.2.7 Boundary Condition

In this section, boundary conditions of online Dictionary, that is, description on how the system started and maintained, handles major failure is given. How the system is started

As described so far, the online dictionary is a web application that should be available always to discharge its services to client requests. Therefore, it is not expected to be started and shutdown every now and then. Yet, there could be a need of starting the system after maintenance or failure recovery. But, even in such cases the Administrator is expected to simply start the server and make sure the IIS is up and running. In addition to this the administrator shall make sure the system directory is available in the appropriate location or it should check if the virtual directory is running.

But in case of deploying the system on commercial server, the administrator is expected to only to manage the system remotely, otherwise assuring availability of IIS will is responsibility of the web hosing company administrator.

4.2.7.1 How the dynamic pages starts-up

Though there is no need to start up the server every now and then, every dynamic page that requires accessing the databases start up in a uniform way by including the connection string variable declared in the web.config file and open procedure that is capable of the connecting to the database and opening it for use. The use of the include file will make the system easy to maintain when there is need of deploying the system for different clients at different web servers. A typical content of a connection string file and its mechanism of including is shown below.

```
<connectionStrings>

<add name="DictionConn" connectionString="Data
Source=.\SQLEXPRESS;AttachDbFilename=|DataDirectory|\Dictionary
.mdf;Integrated Security=True;User Instance=True"
providerName="System.Data.SqlClient"/>

</connectionStrings>
```

A typical way of including the database connection string at dynamic page is by creating sqlDataSource control and assigning the connectingString value to the value declared in the web.Config file or declaring connection string

```
<asp:SqlDataSource ID="ds1" runat="server"
ConnectionString="<%$ ConnectionStrings:DictionConn %>"

or

Public StrConn As String = "Data
Source=.\SQLEXPRESS;AttachDbFilename=|DataDirectory|\Dictionary
.mdf;Integrated Security=True;User Instance=True"
```

4.2.7.2 How the system is maintained

Whenever there is a need of adding additional functionality, there also could be a need of adding new component to the system residing on the web server. This can be done by accessing the web server remotely using FTP username and password to transfer the new or modified component into the system directory.

Use Case Name:	`TransferComponent`
Participating Actors:	**`Administrator`**
Description:	A use case that is used to transfer new or modified component on to the web server
Pre Condition:	The new or modified component shall be available on the local computer
Flow of Events:	1. The **`Administrator`** opens the uploading software such as WSFTP or CUTEFTP 2. Clicks on the `connect command button`; name of the button may vary depending on the type of uploading software being used 3. The software responds by presenting a form to be filled with ftp username, ftp password, IP address and port number 4. **`Administrator`** will fill the form and submits 5. The software will authenticate and if verified it will create connection to the web server and it will display the system directory and local file structure. [Otherwise the system will respond connection failed] 6. **`Administrator`** will select the new or modified component from the displayed local file structure and copy it in to the appropriate location in the system directory
Post Condition:	Component is transferred to web server

4.3 System Specification/Services

The system is decomposed into four main subsystems as shown in the system decomposition section of this document. In this section the detailed service of these subsystems and their preliminary interface is presented.

Search Subsystem

A subsystem that deals with

- searching of Dictionary words From Amharic to English as well as from English to Amharic
- posting request for new word to be added by the system administrator
- posting a new word and its meaning
- posting additional meaning
- posting better meaning
- post comments

Figure 25: Dictionary user interface

Search for the word can be done in two ways, the first option is the user can write the required word in the text box and click on submit button, and the second option is using the list box displayed on the left side of the user interface. The user can see the list of available English or Amharic words by clicking on English or Amharic button and then she/he can double click the required word to see the meaning of the word.

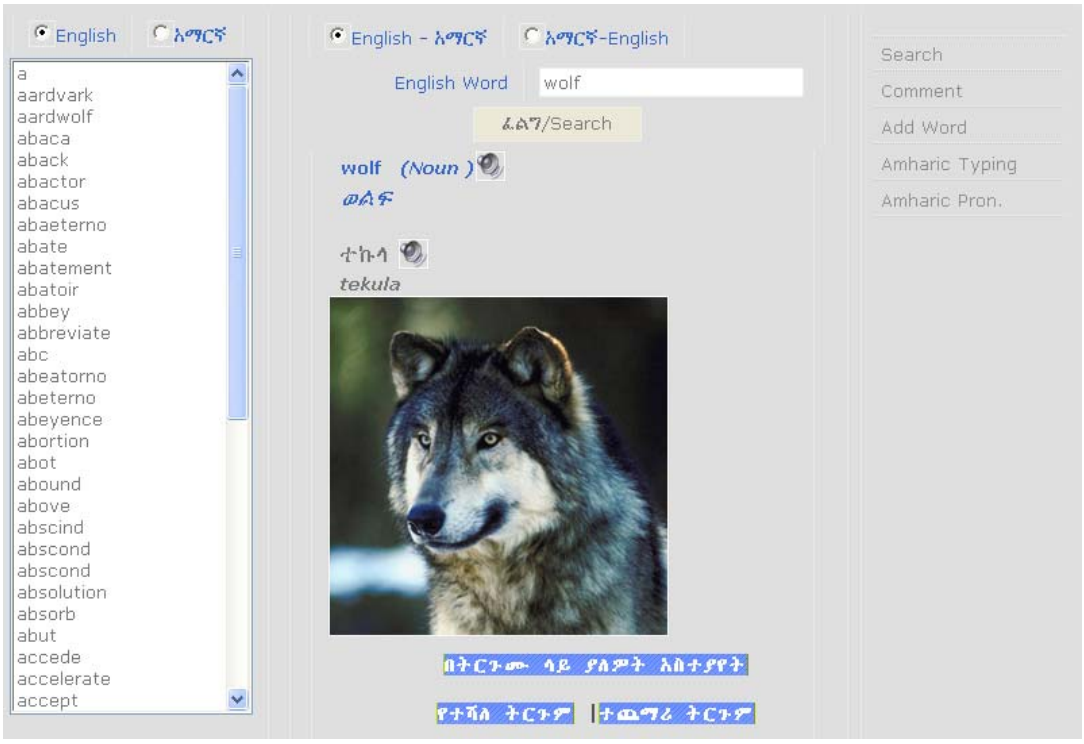

Figure 26: English-Amharic search result with text, sound and picture multimedia content

The same result will be displayed if the user double clicks on the list of the word displayed on the list box.

If the word has more than one meaning, or the search result has more than one result, it will be displayed sequentially and the user can scroll the vertical scroll bar to see additional meaning

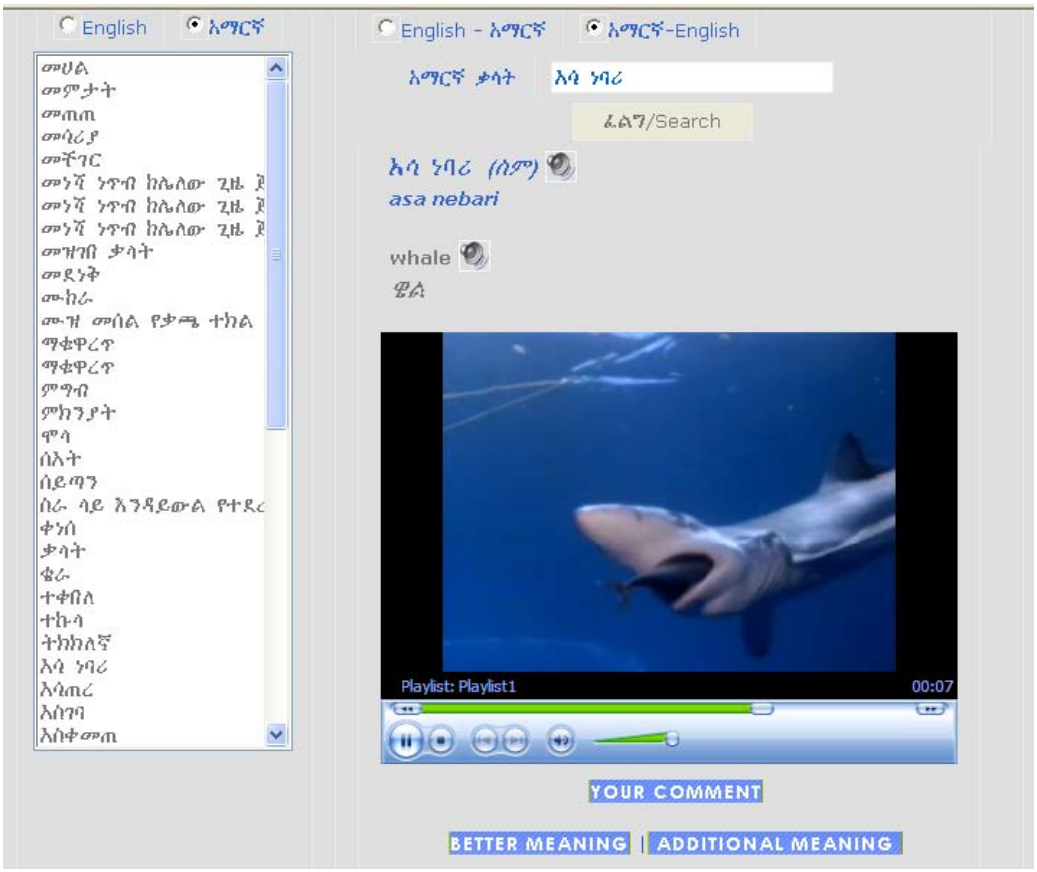

Figure 27: Amharic-English Search result with text, sound and video multimedia content

Figure 28: Displaying additional meaning of the word

Figure 29: Displaying conjugation for English-Amharic word

Figure 30: Playing an audio file using the popup media player

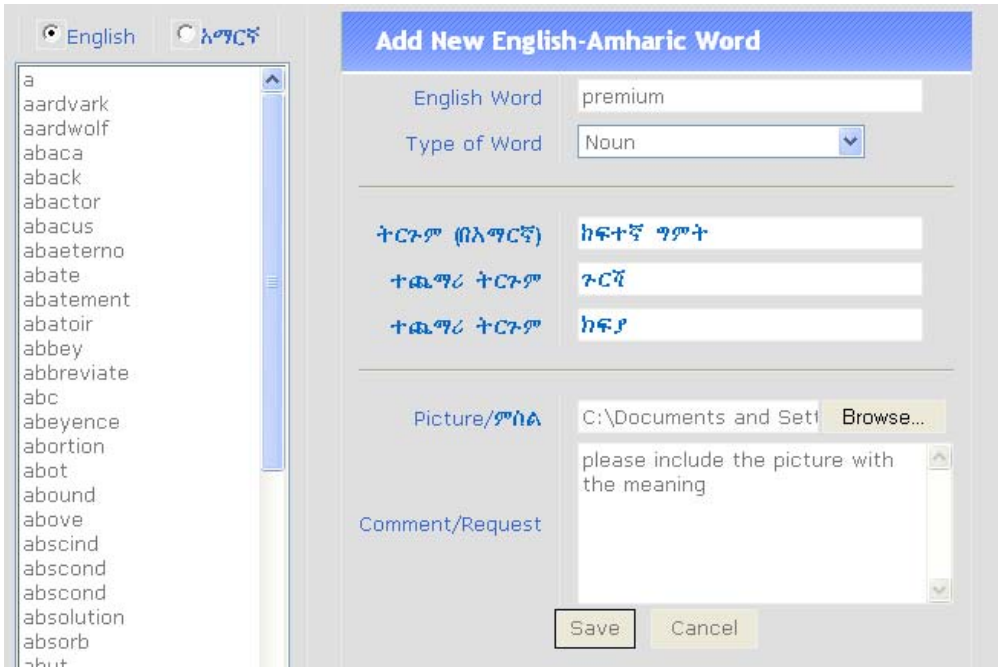

Figure 31: An interface to request to add new word and its meaning

Note: the request to add new word will be posted automatically when the user clicks on request the word to be added link.

Figure 32: An interface to post comment

Dictionary Subsystem

A subsystem that deals with

- managing of Dictionary words i.e. view list, view detail, add, edit, and delete dictionary word.
- posting additional meaning
- posting Conjugation
- publish data
- managing user request i.e. view, edit, delete user requests.

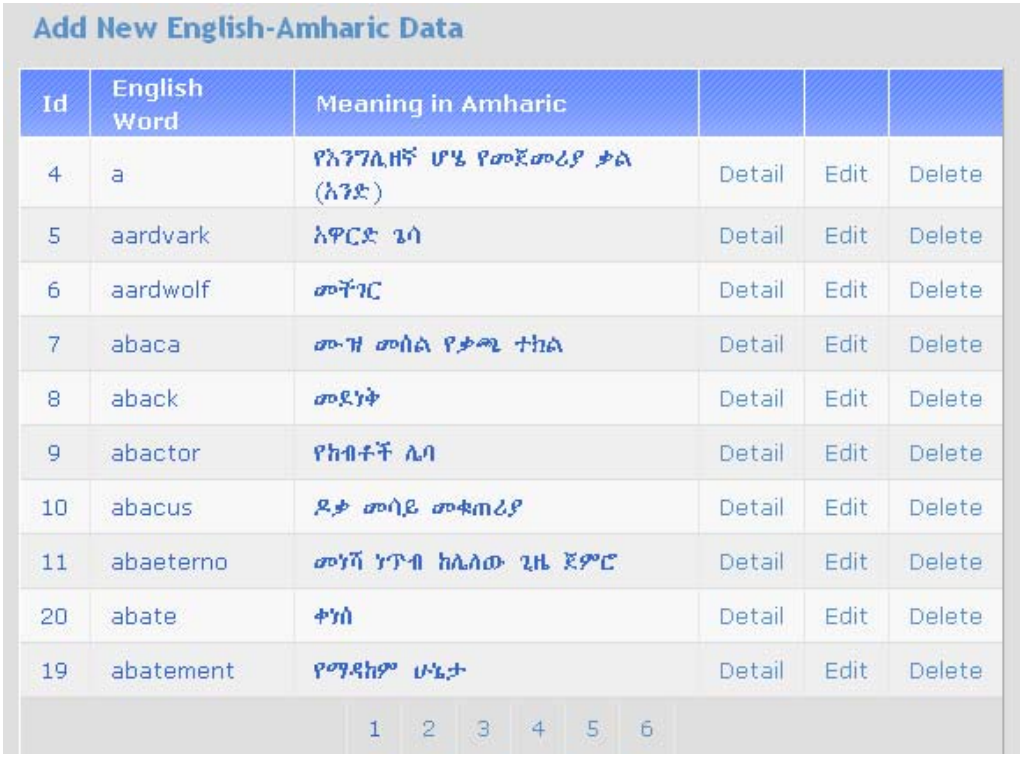

Figure 33:An interface to view all posted dictionary words

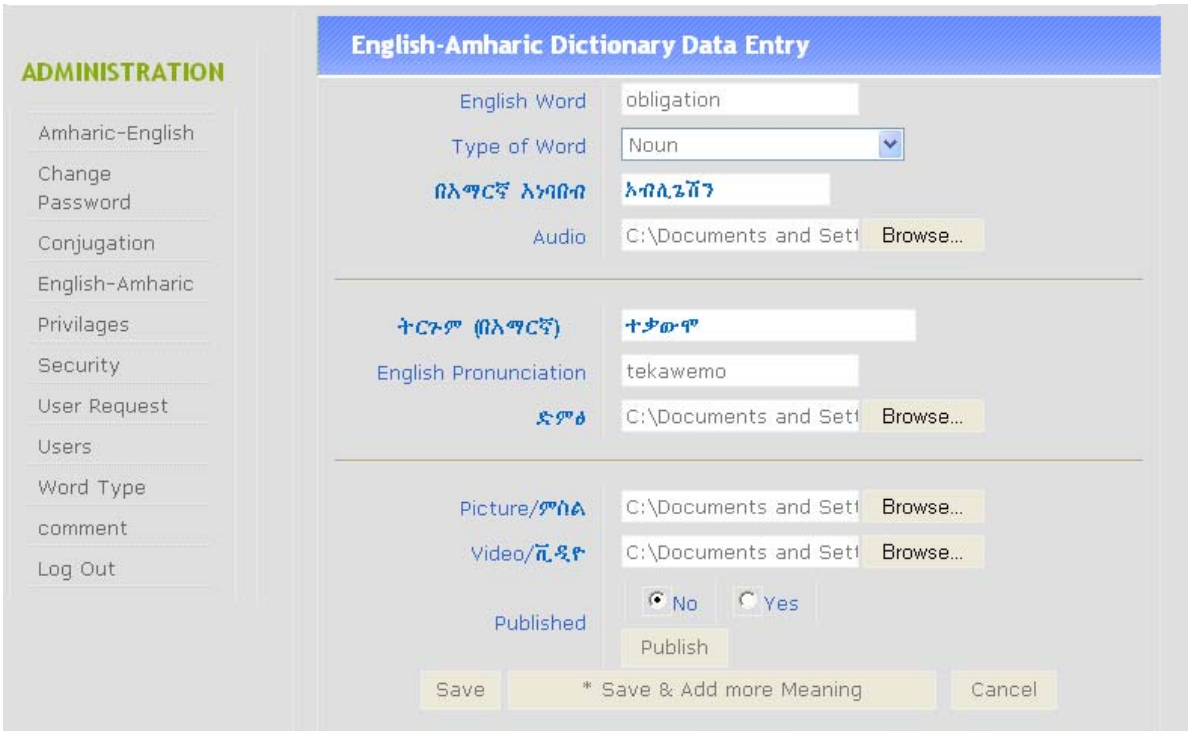

Figure 34: An interface to add new dictionary data

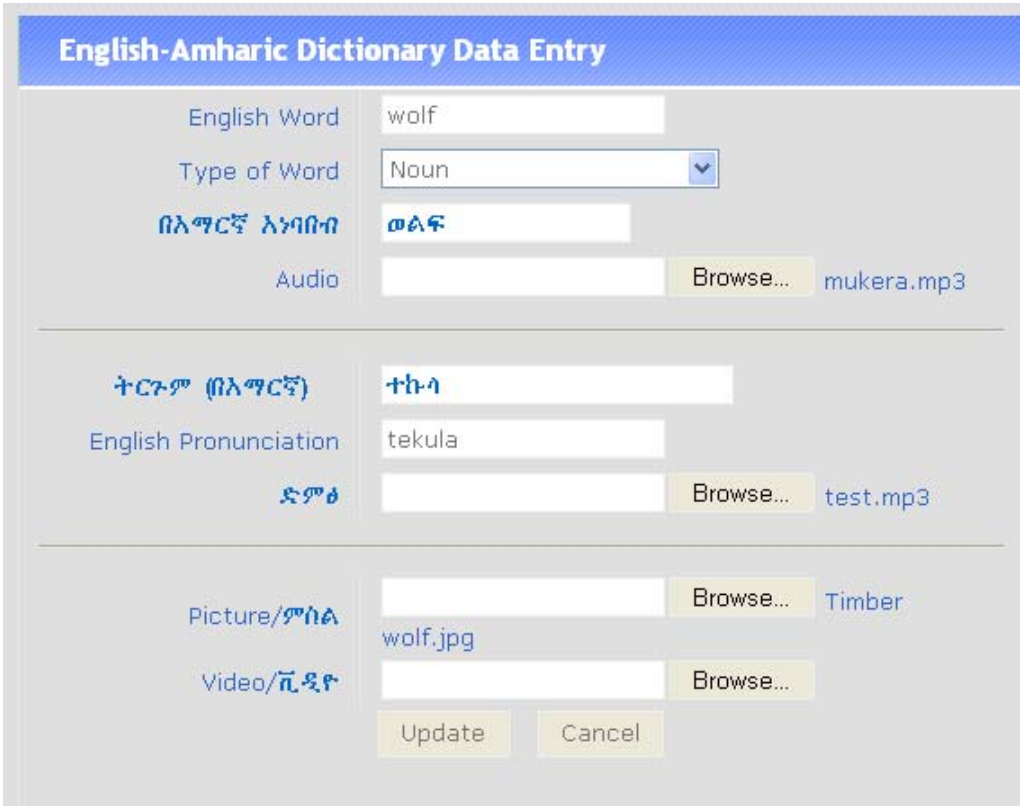

Figure 35: An interface to edit dictionary data

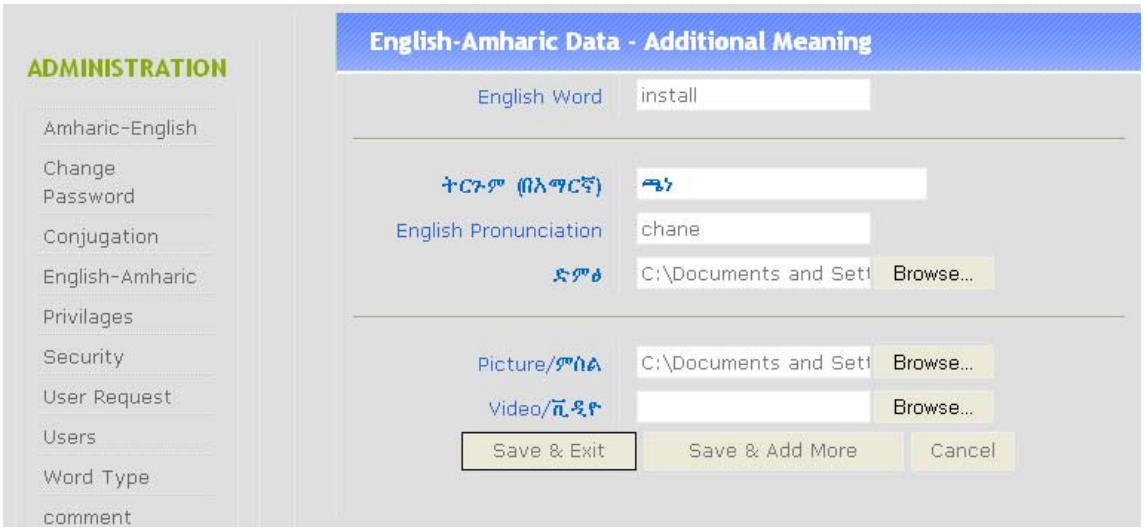

Figure 36: An interface to add additional meaning

Figure 37: An interface to add conjugation

Figure 38: An interface to manage word category

Figure 39: An interface to add new word category

Figure 40: An interface to manage user's request

Figure 41: An interface to view user's request

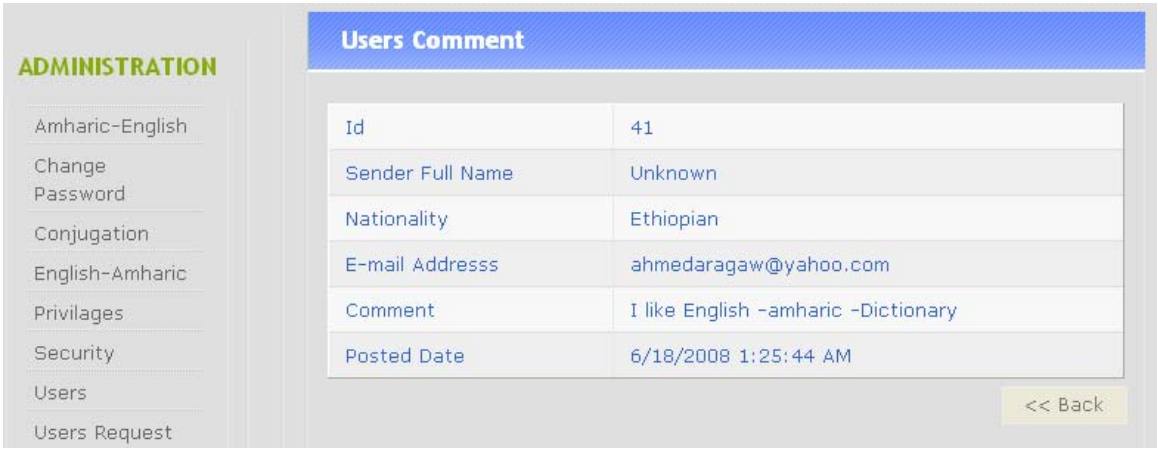

Figure 42: An interface to view user's comment

Users Subsystem

A subsystem that deals with

- managing of user account i.e. view, create, edit, and delete user account
- changing of password

Figure 43: An interface to login page

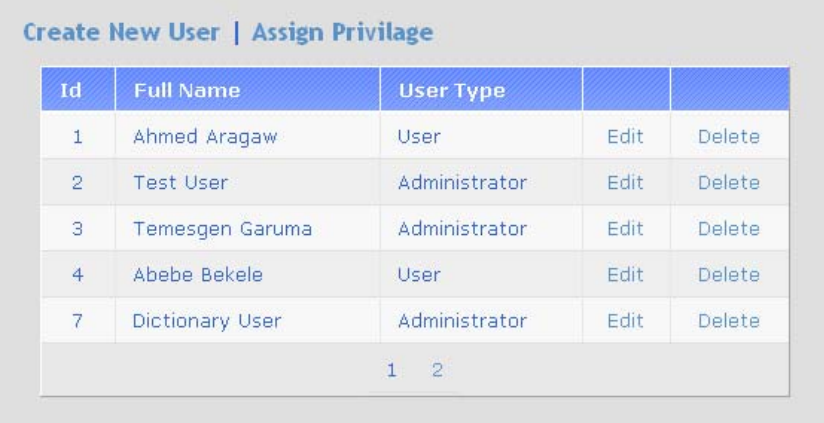

Figure 44: An interface to view all registered users

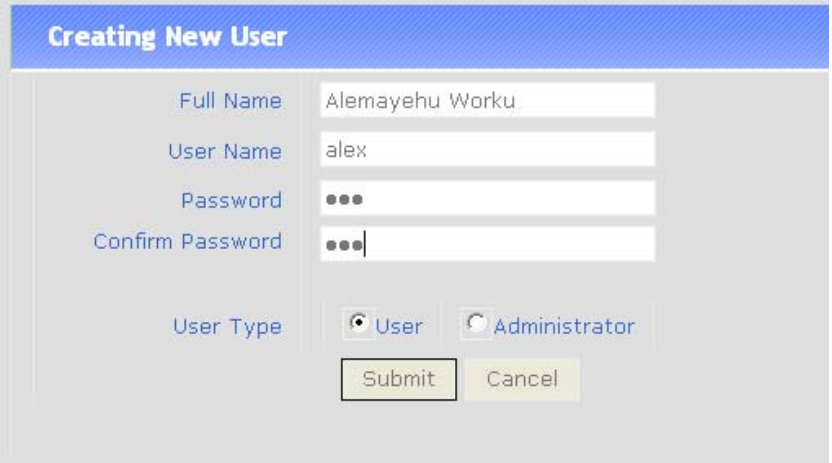

Figure 45: An interface to create new user

Figure 46: An interface to change user password

Security Subsystem

A subsystem that deals with

- security mechanism of the system that include creating the list of administration page that should be secured
- assign privilege to the users

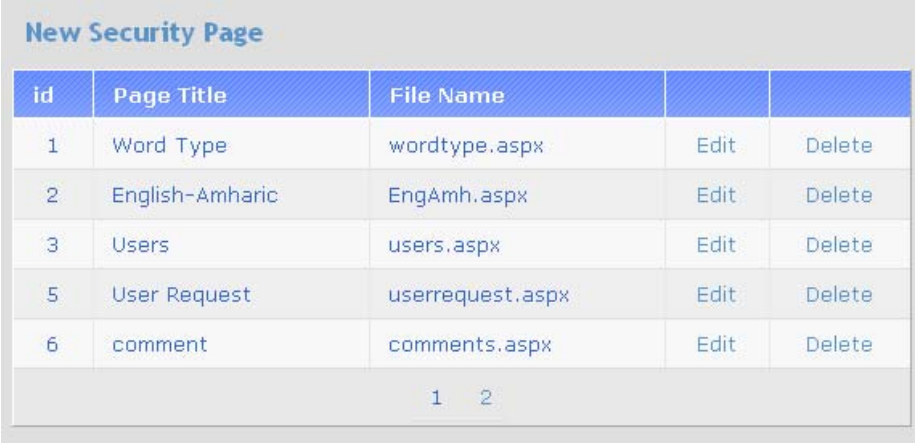

Figure 47: An interface to manage security page

Figure 48: An interface to add new security page

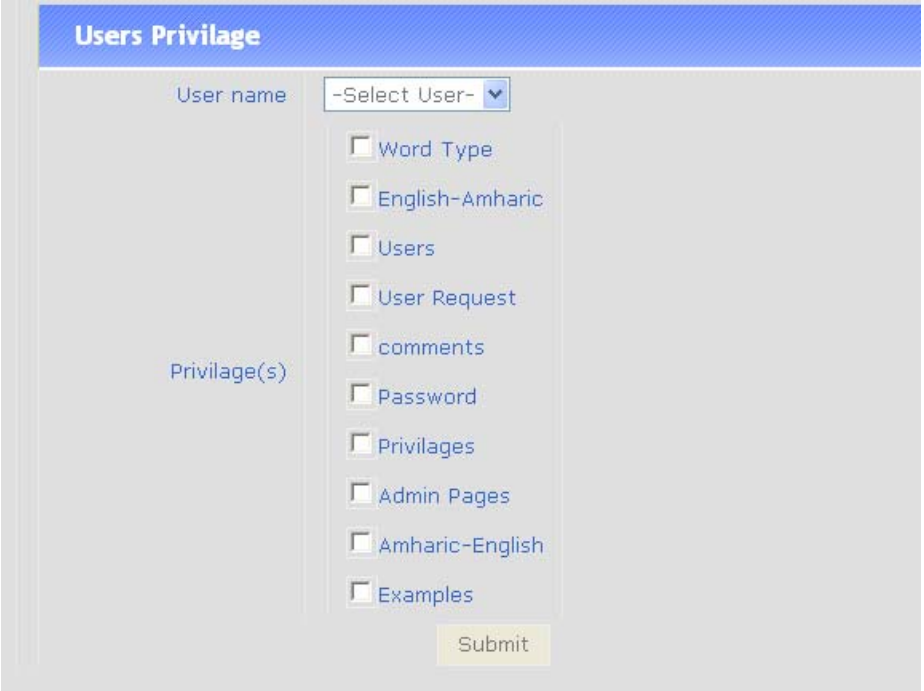

Figure 49: An interface to assign privilege

5. Software Development

5.1 Programming Tool

Web Embedding Font Tool (WEFT)

Web Embedding Fonts Tool, or **WEFT**, is Microsoft's utility for generating embeddable web fonts. WEFT is used by webmasters to create 'font objects' that are linked to their web pages so that users using Microsoft's Internet Explorer web browser will see the pages displayed in the font style contained within the font object. WEFT scans the HTML document file(s), the TrueType font file(s), and some additional parameters. It adjusts the HTML files and creates "Embedded OpenType" files for inclusion on the web site. These files usually use the extension ".eot".

WEFT can embed most fonts, but it will not embed fonts that have been designated as 'no embedding' fonts by their designers. WEFT may reject other fonts because problems have been identified. Embedded fonts are widely used to generate non-English language websites. [8]. Since Online English-Amharic, Amharic-English Multimedia Dictionary uses Amharic fonts to display the Amharic words as well as to write Amharic word when the user wants the translation from Amharic to English, therefore WEFT plays the biggest role to embed the Amharic font to the system and any users can use the online dictionary whether they have Amharic software or not in their Computer.

In this system there is a style sheet called amharic.css that uses an .eot file created using the WEFT software. To create an .eot file, Visual geez UniCode is selected as one of Amharic UniCode Amaharic font.

Here is how the style is created using an .eot file

```
@font-face {
    font-family: Visual Geez Unicode;
    font-style:  normal;
    font-weight: normal;
    src: url(VISUALG0.eot);
}
a{text-decoration:none}
.Amharic_Text
{
color: #0066CC;
```

```
font-family: Visual Geez Unicode;
font-size: 10pt;

}

....
```

The above amharic.css file is attached to each of the page in the system using the following code.

```
<link href="styles/Dictionary.css" type="text/css" rel="stylesheet"
/>
```

JavaScript

JavaScript is a client side script functions that cause elements on the page to respond to user actions. One of the advantage of JavaScript over Server side script or code is, it is working on the client side and it doesn't request the server to respond therefore it reduces the download time and make the system to respond to the users action very fast. Therefore, in the dictionary system, there are two JavaScript code that performs the major job of the system. The first one is a PC Typing JavaScript code that makes the users to write Amharic and the other one is the popup media player window that makes the audio files to play with out requesting the server to respond to the user action.

Amharic Typing Script

Since this project involves the translation from Amharic to English and from English to Amharic, Embedding the UniCode version of Amharic font is not enough because the user should be able to write Amharic and search for English translation. The online dictionary will be access from different corner of the world and users do not have Amharic software installed in their computer could not write Amharic text, and the project will not serve its purpose with out having its own Amharic pc typing script that will be embedded or attached with the system and will be activated when the users tries to write Amharic text on the search text box that are designed to accept Amharic words.

The Amharic typing JavaScript code works like any Amharic phonetic keyboard combination that is easy to the users to adapt it.

In this project, the second step next to Embedding Amharic font to the online dictionary system was creating Amharic typing code. Therefore the script tries to map all the Amharic Unicode to the physical keyboard layout with the already familiar Amharic key combination. Any user that uses PowerGeez or VisualGeez Amharic software can easily write and get the required result. For the new or the one who is not familiar with the Amharic software, a help is designed that will show how to write Amharic word and its key combination.

The source code for this PC Amharic typing JavaScript code is attached with document. See Annex I.

Popup Audio Player

Popup Audio player is a JavaScript code that makes audio files to play in the client side. Here when the users search for any dictionary word, an audio file will be attached to both Amharic and English word how to pronounce it. The files are already stored in the Audio folder of the system and the filename are found in the database. The attachment is done dynamically when the file is displayed by the system using ASP.Net VBScript Code.

```
Sample code for attaching an audio file.

tCell3.Text += "<a onclick='javascript:PlayerOpen(this.href); return
false;' href='Audio\" & rst1("AmhSound") & "' target=_self> <span
class='audio'><img src='Images/audio.gif'></span></a></br>"
```

And the javascript code will be executed when the user click on an audio image displayed next to the English or Amharic file.

The javascript code to play an audio file looks like this.

```
var UniqueID = 314 // Make each link open in a new window.
var newWinOffset = 300 // Position of first pop-up
function PlayerOpen(soundfilepath) {

PlayWin = window.open('',UniqueID,'width=305,height=42,top=' +
newWinOffset
+',left=350,resizable=0,scrollbars=0,titlebar=0,toolbar=0,menubar=0,s
tatus=0,directories=0,personalbar=0');
PlayWin.focus();
var winContent = "<HTML><HEAD><link href='~/styles/Dictionary.css'
type='text/css' rel='stylesheet' /><TITLE>AE,EA MM
Dictionary</TITLE></HEAD><BODY bgcolor='#CCCCCC'>";
Dictinary</P></Center>";
var winContent = "<OBJECT>";
winContent += "<param name='SRC' value='" +  soundfilepath + "'>";
winContent += "<param name='AUTOPLAY' VALUE='true'>";
winContent += "<param name='CONTROLLER' VALUE='true'>";
winContent += "<param name='BGCOLOR' VALUE='#CCCCCC'>";
winContent += "<EMBED SRC='" + soundfilepath + "' AUTOSTART='TRUE'
LOOP='FALSE' WIDTH='305' HEIGHT='42' CONTROLLER='TRUE'
BGCOLOR='#CCCCCC'></EMBED>";
winContent += "</OBJECT>";

winContent += "<FORM><DIV align='center'><INPUT type='button'
value='Close this window'
onclick='javascript:window.close();'></DIV></FORM>";

winContent += "</BODY></HTML>";
PlayWin.document.open(winContent);
PlayWin.document.write(winContent);
PlayWin.document.close(); // "Finalizes" new window
UniqueID = UniqueID + 1
return false;

}
```

5.2. Development Environment

The development Environment that is used in this system is ASP.NET. ASP.NET provides a unified Web development model that includes the services necessary to build enterprise-class Web applications. ASP.NET is part of the .NET Framework and enables you to take full advantage of the features of the common language runtime, such as type safety, inheritance, language interoperability, and versioning [9]

Since ASP.NET has too many web controls that help to create dynamic Web applications and it is selected as the Development Environment.

ASP.NET applications typically run using IIS (Internet Information Services) as a Web server. Internet Information Service (IIS) serves as the main Web Server in the system.

For a database platform implementation, SQL Server 2005 is selected since it provides simplicity of use and integration with ASP. Net.

Method of inserting Unicode Amharic values to the database

To insert a Unicode Amharic text to SQL Server 2005 database needs a conversion mechanics. The Amharic text should be converted to its equivalent Unicode value. The conversion function makes the server to know the value inserted in the database is a Unicode value or the SQL statement explicitly specifies to insert Amharic text as a Unicode value otherwise all the data entered to the database will be garbage value. Using the conversion function, Amharic text will be saved as a Unicode value.

Sample code how to insert Amharic text to the database

```
SQL = "INSERT into Dictionary (Englishword, wordType, AmhPron,
EngSound, Amharicword,EngPron, AmhSound, Picturename,
Videoname,postedby,posteddate) " & _
"VALUES ('" + txtenglish.Text + "'," + cmbEngType.SelectedValue +
",N'" + Replace(txtAmhpronouncation.Text, "'", "''") + "','" &
FileAudioEnglish.FileName & "',N'" + Replace(txtAmharic.Text, " '",
"''") + "','" & txtEngpronouncation.Text & "','" &
FileAudioAmharic.FileName & "','" & FilePictureUpload.FileName &
```

```
"','" & FileVideoUpload.FileName & "','" & Session("fullname") & "','" & Now() & "')"
```

Appending an N (National character) in the amharic text values will tell the database to save the value as unicode value.

5.3. Multimedia Content Tools

5.3.1 Picture

Source

The main sources of picture file in the system are Clip art CDs, Internet, scan picture and capturing using a digital camera

Editing/Processing

Once the picture is captured using the above mentioned tools, Adobe Photoshop is editing software. For the purpose of uniformity, all the pictures will be cropped in to 250px by 250 px and the resolution will be 72 pixel per inch to make the pictures displayed/downloaded very quickly.

Adobe Photoshop is a professional picture editing software and the Data Encoder or Administrator may have limited knowledge about the software and therefore training should be conducted how they use it but for this system only few of its features will be applied especially how to crop the picture and set the resolution of the picture.

This is how the users crop and set the resolution.

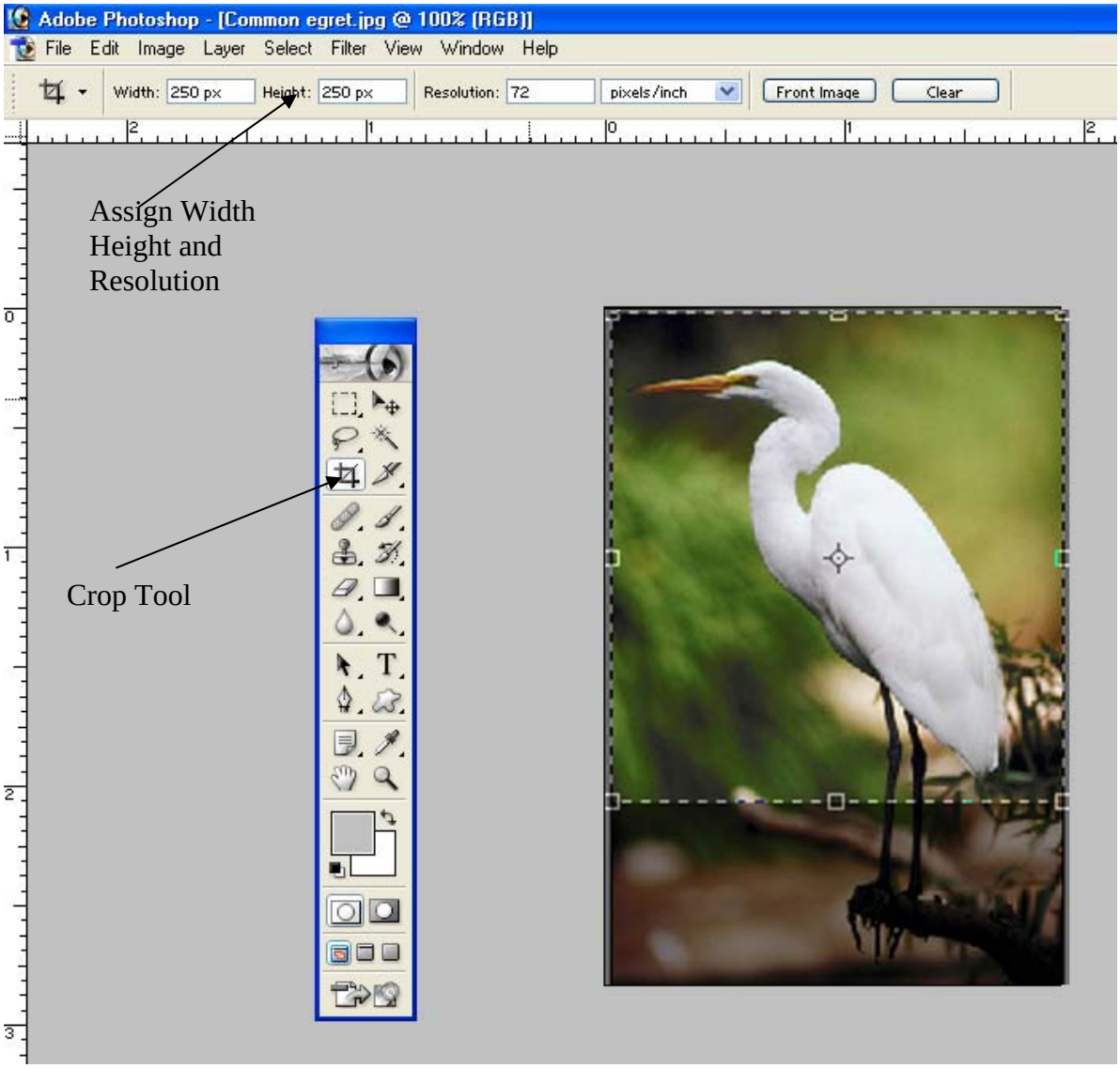

Figure 50: cropping picture using Adobe Photoshop

Displaying the picture

Using a code is the method of displaying any picture in the system. Picture is one of the multimedia file that doesn't need any control or component. To display the picture, the following code is used.

```
<IMG SRC='Pictures/" & rst1("Picturename") & "' width='250'
height='250'>
```

As mentioned above, all the picture will be cropped with pixel size of 250 X 250 but for the sake of safety, there is a code that will make all the picture to be displayed using the width = 250 and height =250

5.3.2 Audio

Audio is another multimedia content in this system and audio helps the users to know how the word will be pronounced correctly in both Amharic and English words.

Source

The source of audio is recording the sound using Microphone.

Editing/Processing

Cool Edit 2000 is the main software for audio processing. Cool Edit 2000 is the professional software editing program that helps the user to edit sound, crop the required part and save it in different format. In this project cool Edit helps to capture an audio file and crop only the required part of the recorded audio that makes helps to remove the silence part of the recorded audio, in addition to this, Cool Edit saves the audio in different format. For this project, MP3 is selected as the audio type because MP3 has a good compression that makes the file size to be small.

Playing an Audio File

To play the audio file, the pop up JavaScript code is selected; it runs on the client side. The audio player will be opened and play the audio file from the path specified dynamically when the page loaded/created.

SoundPlayer server control is another option to play an audio file, but this control makes the page to be rendered again. Every time when the user wants to hear an audio file, the page should be submitted again and all the controls in the page should be rendered, this makes the page to be downloaded again and again.

This is how to play audio file using soundplayer ASP. Net built-in control

```
Dim objPlayer As New SoundPlayer
objPlayer.SoundLocation = server.mappath(".")\audio\test.mp3"
objPlayer.Play()
```

5.3.3 Video

Video also one of the multimedia content in this project that is used to show some dictionary word with the moving picture and sound that makes it understand very

Source of Video File

The main source of Video file is recording, different CDs and other sources.

Processing

To process Video file, Adobe Premium software is selected. Using Adobe Premium the required part of the video can be easily cropped.

Displaying Video Files

To display video files, ASPNetVideo Component is used because [12]

- The ASPNetVideo Component Suite allows to add video files and streaming media into ASP.Net web forms.
- ASPNetVideo is unique in that it ASP.NET component to embed Windows Media Player, QuickTime and RealPlayer into ASP.Net websites.
- The ASPNetVideo Component allows streaming videos to be directly embedded into .Net ASPX web pages.
- The WebControl produces seemless W3C XHTML for QuickTime (.mov), Real Player (.ram, .rmv), Windows Media Player (.wmv, .avi)

How to use ASPNetVideo Control

- Register/add the control in the ASP.Net toolbox to be available for use.
- Three type of ASPNetVideo will be available for use
    - ASPNet WindowsMedia
    - ASPNet RealPlayer
    - ASPNet QuickTime
- Drag one of the ASPNetVideo controls from the toolbox onto your ASPX web form in design mode or create the ASPNetVideo controls dynamically as illustrated below. For this project, ASPNet WindowsMedia player is selected

since it is available in any windows machine and most users are familiar with it, it also support the most available video file format i.e. .avi file.

- Change the **VideoURL** property to point to a video file. This is normally a video file uploaded when the dictionary data registered (the dictionary video file is saved under a folder called Video). It may also be the URL of a remote video stream.

There are many video formats, and the control can support almost all of them.

- **.wmv** & **.avi** files for Windows Media Player
- **.ram** & **.rmv** for RealPlayer
- **.mov** files for QuickTime Player

Here is the sample code how the ASPNetVideo control is used in the system

```
Dim DictVideo As New ASPNetVideo.WindowsMedia
Dim tRow As New TableRow
Dim tCell As New TableCell

DictVideo.Visible = True
DictVideo.VideoURL = "~/video/" & rst("videoname")
DictVideo.AutoPlay = True
DictVideo.DataBind()

tCell.Controls.Add(DictVideo)
tRow.Cells.Add(tCell)
Table.Rows.Add(tRow)
```

Summary

Multimedia Files	**Capturing/Recording**	**Processing**	**Displaying/Playing**
Audio Files	Microphone Cool Edit 2000	Cool Edit 2000	Popup Audio Player
Picture Files	Scanner, Digital Camera, Clip Art CDs & Internet	Adobe Photoshop	Using code
Video Files	Video Camera, CDs, and other sources	Adobe Premier	ASPNetVideo control

6. Conclusion and Recommendations

6.1 Conclusion

In this project, we have developed an online dictionary that provides online translation service in Amharic and English language. To display Amharic text online, we have user a WEFT tool. WEFT is a Microsoft's utility for generating embeddable web fonts. It creates "Embedded Open Type" files for inclusion on the web site. These files usually use the extension "`.eot`". We have created a style sheet that uses the embedded Amharic Unicode font. To display Amharic text, we have applied the style sheet created using the embedded font. When ever a client request the dictionary service from the server, the client machine will use the embedded font instead looking for fonts found in the local machine. The embedded Amharic Unicode font makes Amharic words to be displayed in any computer with out the need for downloading or looking for an Amharic font.

There are 380 Amharic alphabets and for each alphabet there is a corresponding Unicode value. To write an Amharic text using the Unicode system, we have developed an Amharic typing JavaScript code that maps all Amharic Unicode values to their respective key on the physical keyboard. The JavaScript code we have developed tries to map the Amharic alphabet with the phonetic keyboard combination that most people are familiar with. Our Amharic typing script works like any Amharic phonetic keyboard combination. Using this script, users can write any Amharic text. We have also prepared a popup help page that is designed to support users while typing the key combination to produce an Amharic character.

In addition of displaying the meaning of the word with multimedia content, the online dictionary supports users' interaction. It allows the users to post comments, requests and even they can add new words in the dictionary. They can also give comment on the meaning of the word, post additional meaning or post a better meaning.

Our online dictionary also have a separate administration part that helps the system administrator to manage all dictionary functionalities that includes management of dictionary data, users, word category, security, conjugation, additional meanings, users privilege and users request.

For multimedia data, different tools are used for capturing/recording, processing and displaying. For an audio content, we have used microphone & cool edit 2000 for capturing and processing, JavaScript popup audio player to play the audio content. For picture content, the main sources of picture files are Clip Art CDs, Internet, scanner and Digital camera. For image processing, Adobe Photoshop is used for cropping the picture in to equal size and to make some modification. For video content, an ASPNetAudio component is used. The ASPNetVideo Component Suite allows to add video files and streaming media into ASP.Net web forms. For processing of Video files, Adobe premier is used.

We have used ASP.Net for the system development and SQL server for database Implementation.

6.2 Recommendations and Future work

To provide better services of the dictionary, it should be rich in multimedia content. The availability of the words in the dictionary enhanced by adding as many words as possible and its conjugation especially for Amharic words based on different existing dictionary sources. Therefore, we recommend an extended work towards entering more data and multimedia content into the dictionary database. This will make the dictionary to be complete and provide a better service to the users.

Users' comments and requests are also one of the means to improve the service of the dictionary and therefore, the administrator of this dictionary should consider users comments and requests on regular basis.

Among some of the future works that enhance the dictionary are:

- Spell Checker - the utility that will help the user when searching the dictionary by proposing a correctly spelled word or the nearest available word if the user makes spelling error.
- Text to Speech conversion - in this dictionary system, we used microphone (hardware) and cool edit (software) for recording of an audio file. It is time taking job and if there is text to Speech conversion tool, it will decrease the time of inserting dictionary data.

References

[1] http://www.en.wikipedia.org/wiki/Online dictionary, Visited on February 18, 2008

[2] Ze-Nian Li and Marks S. Drew (2004). Fundamentals of Multimedia, Pearson Prentice Hall, Pearson Education Inc., Upper Saddle River, New Jersey (NJ07458)

[3] http://www.amharicdictionary.com, visited on February 15, 2008

[4] http://www.ethiopiandictionary.com, visited on February 15, 2008

[5] http://www.w3.org/1999/xhtml, visited on February 16, 2008

[6] G. Booch, Object-Oriented Analysis and Design with Applications, Second Edition, Benjamin/Cummings Publishing Company, Inc., Redwood City, CA, 1994.

[7] Roger S. Pressman, Software Engineering: A Practitioner's Approach, Fifth Edition, McGraw-Hill Higher Education, 2001.

[7]http://www.codeproject.com/KB/aspnet/GoogleSuggestDictionary.aspx, visited on April 15, 2008

[8] http://en.wikipedia.org/wiki/Web_Embedding_Fonts_Tool, visited on Feb 22, 2008

[9] http://www.ethiotrans.com/amharic1.htm, visited on April 05, 2008

[10] http://tlt.its.psu.edu/suggestions/international/bylanguage/ethiopicchart.html, visited on February 10, 2008

[11] http://www.codeproject.com/KB/audio-video/fister.aspx, visited on April 10, 2008

[12] http://www.aspnet-video.com, visited on April 12, 2008

[13] http://wiki.eclipse.org/images/b/b7/Cola_client_server.png, visited on July 15, 2008

Glossary

Actors - is a person, organization, or external system that plays a role in one or more interactions with the system.

Attributes and Operations - indicate the class attributes and methods

Classes - Self-containing components depicted in rectangles comprising of name, attributes and methods. They should encapsulate only the information and functionality that is important to them and in context of the system.

Extend relationship - indicating the use case will occasionally use some another (extended) use case. Another way round, it means that extended action (to which an arrow points) can optionally include the extending action.

Generalization - identifies relationship between use cases, or between actors, in particular the inheritance.

Include relationship - Indicating the use case will always use some another (included) use case. It means that the included use case can be shared (reused) among other use cases.

Multiplicity - indicates how many objects of one class relate to one object of another class.

Relationships - connect two classes/actors with use cases together.

Roles - indicate the role of the class in relation to another class, or the name of the attribute (variable) referencing the object of another class.

System - something that performs a function

Subsystem - parts of the system can grouped into a subsystem.

Framework - reusable design for a software system (or subsystem).

SQL Server- Database Management system provided by Microsoft

ClipArt CD - Compact Disks that contains a collection of icons, button, and other useful image files, that can be inserted into pages

Annex I

Amharic PC Typing Script

```
<script language="javascript" type="text/javascript">
function isRootLetter(UniCodeValue)
{
      var isRoot = false;

      switch (UniCodeValue)
      {
        case 4608: //Ha h
        case 4616: //Le l
        case 4624: //Ha(Hamruha)    h + Shift
        case 4632: //Me m
        case 4640: //Se s
        case 4648: //Re r
        case 4656: //Se (Negusu)    s + Shift
        case 4664: //She      s + Caps Lock
        case 4672: //qe q
        case 4704: //Be b
        case 4712: //Ve v
        case 4720: //Te t
        case 4728: //Che      c
        case 4736: //Ha (Haile Selassie ) x
        case 4752: // Ne      n
        case 4760: // Nge     n + Shift
        case 4768:
        case 4776: // ke      k
        case 4792: // Hea     k + Shift
        case 4808: // we      w
        case 4816: // a a
        case 4824: // ze      z
        case 4832: // Zhea    z + Shift
        case 4840: //ye y
        case 4848:
        case 4856:
        case 4864: // je      j
        case 4872:
        case 4880:
        case 4888:
        case 4896: //te t
        case 4904: //Che      c + Shift
        case 4912: //Pe p
        case 4920: //Tse      t + shift
        case 4928: //Tze      t + caps lock
        case 4936:
        case 4944: //pe p
        case 4954:
            {
                  isRoot = true;
                  break;
            }
      }
      return isRoot;
};
function withinRange(numValue)
{
      if ( numValue >= 4608 && numValue <= 4988)
      {
            return true;
      } else{
```

```
                return false;
        }
};
function AmharicTyping(pressEvent, amharic){
        var keyCode = 0
        var isNetscape = false;
        var range;
        var newPos;
        var startPos = 0;
        var endPos = 0;

        if(window.event) {
                keyCode = pressEvent.keyCode;
                range = document.selection.createRange();
                newPos = getCursorPosition();
                startPos = getCursorPosition();
                endPos = getCursorPosition();
        }
        else if(pressEvent.which)
        {
                keyCode = pressEvent.which;
                isNetscape = true;
        }
        var shiftKey = false;
var controlKey = false;
if (window.event)
{ shiftKey = window.event.shiftKey; controlKey=window.event.ctrlKey;}
else if (pressEvent.which)
{ shiftKey = pressEvent.shiftKey; controlKey=pressEvent.ctrlKey;}
if( controlKey) return true;
        var keyString = String.fromCharCode(keyCode).toLowerCase();
        var rawString = String.fromCharCode(keyCode);

        if ( keyCode > 4000 || (keyCode >= 33 && keyCode <= 57 &&
keyCode != 44) || (keyCode >= 91 && keyCode <= 93) || (keyCode >= 60
&& keyCode <= 64) || (keyCode >= 123 && keyCode <= 125))
        {
                return true;
        }
        if (amharic.selectionStart || amharic.selectionStart == '0')
        {
                startPos = amharic.selectionStart;
                endPos = amharic.selectionEnd;
                newPos = startPos;
        }
        var lastCharPlain = amharic.value.charCodeAt(startPos-1);

        if (keyCode != 16 && keyCode != 8 && keyCode != 13 && keyCode
!= 0 && keyCode != 32){
                var NewValue = "";
                var UniCodeValue = keyString;
                var tempExtraLetter =
amharic.value.substring(0,amharic.value.length - 1);
                var consonantRetVal =
convertEnglishConsonantToAmharic(lastCharPlain, UniCodeValue,
shiftKey);
                if (consonantRetVal != null && consonantRetVal.length >
0)
                {
                        var tempStartPos = startPos;
                        if( UniCodeValue == "h" && shiftKey == false)
                        {
                                var hconsValue = 0;
```

```
                        if ( lastCharPlain == 4624)
                        {hconsValue = 4736;}
                        else if (lastCharPlain == 4608)
                        {hconsValue = 4624;} else
                        {hconsValue = 4608; newPos = newPos + 1;
tempStartPos = startPos + 1;}
amharic.value = amharic.value.substring(0, tempStartPos-1)
+ String.fromCharCode(hconsValue )
+ amharic.value.substring(startPos, amharic.value.length);
                  }else if ( UniCodeValue == "s" && shiftKey ==
false)
                  {     var tempStartPos = startPos;
                        var hconsValue = 0;
                        if ( lastCharPlain == 4656){
                              hconsValue = 4640;
                        } else {hconsValue = 4656; tempStartPos =
startPos + 1; newPos = newPos + 1}
amharic.value = amharic.value.substring(0, tempStartPos-1)
+ String.fromCharCode(hconsValue )
+ amharic.value.substring(startPos, amharic.value.length);

                  } else if (UniCodeValue == ",")
                  {
                        var tempStartPos = startPos;
                        var hconsValue = 0;
                        if ( lastCharPlain == 4963){ //Comma
                              hconsValue = 44;
                        } else {hconsValue = 4963; tempStartPos =
startPos + 1; newPos = newPos + 1}
amharic.value = amharic.value.substring(0, tempStartPos-1)
+ String.fromCharCode(hconsValue )
+ amharic.value.substring(startPos, amharic.value.length);

                  }else
                  {
amharic.value = amharic.value.substring(0, startPos)
+ consonantRetVal
+ amharic.value.substring(endPos, amharic.value.length);
newPos = newPos + 1;}}
            var aOffset = 3;
            var eOffset = 5;
            var iOffset = 2;
            var uOffset = 1;
            var oOffset = 6;
            var eeOffset = 4;
            var offset = 0;
            var vowelCharacter ;
            var otherVowelCharacter = 0;
if ( (UniCodeValue == "a" && shiftKey == false) || UniCodeValue ==
"4")
            {offset = aOffset;}
else if ( (UniCodeValue == "e" && shiftKey == false) || UniCodeValue
== "1")     {
                  offset = eOffset;
                  vowelCharacter = 4773;
if ( lastCharPlain == 4773 ) { otherVowelCharacter = 4816;}
}else if ( (UniCodeValue == "i") || UniCodeValue == "3")
            {     offset = iOffset;
                  vowelCharacter = 4770;
if ( lastCharPlain == 4770 ) { otherVowelCharacter = 4818;}
}else if (( UniCodeValue == "o") || UniCodeValue == "7")
            {     offset = oOffset;
                  vowelCharacter = 4774;
```

```
if ( lastCharPlain == 4774 ) { otherVowelCharacter = 4822;}
}else if ( (UniCodeValue == "u") || UniCodeValue == "2")
            {                   offset = uOffset;
                  vowelCharacter = 4769;
if ( lastCharPlain == 4769 ) { otherVowelCharacter = 4817;}
}else if ( (UniCodeValue == "e" && shiftKey ==true) || UniCodeValue
== "5")
            {     offset = eeOffset;
                  vowelCharacter = 4772;
if ( lastCharPlain == 4772 ) { otherVowelCharacter = 4820;}
}  else if( UniCodeValue == ";" )
            {
amharic.value = amharic.value + String.fromCharCode(4964);//Semi
Colon
                  newPos = newPos + 1;
            } else if( UniCodeValue == ":" )
            {
                  amharic.value = amharic.value +
String.fromCharCode(4961);//space
                  newPos = newPos + 1;
}
      var vowelCharacterChar = String.fromCharCode(vowelCharacter);
      var lastCharEtymology = amharic.value.charCodeAt(startPos-1);

            if ( offset != 0 )
            {
                  if ( amharic.value.length < 1 )
                  {
                  amharic.value = amharic.value + vowelCharacterChar
;
                              newPos = newPos + 1;
                  }else if(amharic.value.charCodeAt(startPos-1) == 32
)
                  {
                  amharic.value = amharic.value + vowelCharacterChar
;
                              newPos = newPos + 1;

                  }
                  else if (lastCharEtymology  == 4883 ||
lastCharEtymology == 4683 || lastCharEtymology == 4803 ||
lastCharEtymology == 4787 || lastCharEtymology == 4747)
                  {
if ( UniCodeValue == "e" && shiftKey == false) lastCharEtymology =
lastCharEtymology - 3;
if ( UniCodeValue == "i") lastCharEtymology = lastCharEtymology - 1;
if ( UniCodeValue == "e" && shiftKey == true) lastCharEtymology =
lastCharEtymology + 1;
if ( UniCodeValue == "u") lastCharEtymology = lastCharEtymology + 2;

amharic.value = amharic.value.substring(0, startPos-1)+
String.fromCharCode(lastCharEtymology )+
amharic.value.substring(startPos, amharic.value.length);
      } else
                  {
      var lastchar = amharic.value.charCodeAt(startPos-1) + offset;
            if ( otherVowelCharacter > 0)
                        {
amharic.value = amharic.value.substring(0, startPos-1)+
String.fromCharCode(otherVowelCharacter)+
amharic.value.substring(startPos, amharic.value.length);
                                    }
      else if ( isRootLetter(lastCharPlain) )
```

```
                        {
      amharic.value = amharic.value.substring(0, startPos-1)
            + String.fromCharCode(lastchar)
            + amharic.value.substring(startPos,
amharic.value.length);
                        } else
                        {
      amharic.value = amharic.value.substring(0, startPos)+
vowelCharacterChar+ amharic.value.substring(startPos,
amharic.value.length);
                        newPos = newPos + 1;
                        }}}
            if ( UniCodeValue == "w" && shiftKey == true )
                  var lastchar2 = amharic.value.charCodeAt(startPos-
1);
                  if (amharic.value.length < 1)
                  {
                        lastCharModified = 4775;
                        amharic.value = String.fromCharCode(4775)
                        newPos = newPos + 1;
                  }
if( ! (lastchar2 == 4840 || lastchar2 == 4808 || lastchar2 == 4928 ||
lastchar2 == 4768  || lastchar2 == 4816) )
                  {
var lastCharModified;
if (lastchar2 == 4672 || lastchar2 == 4872  || lastchar2 == 4776  )
                        {
                              lastCharModified = lastchar2 + 6;
                        } else if (lastchar2 == 4608)
                        {
                              lastCharModified = 4747;
                        } else if (lastchar2 == 4792)
                        {
                              lastCharModified = 4803;
                        }
                        else if ( lastchar2 == 4736)
                        {
                              lastCharModified = 4747;
                        } else
                        {
                              lastCharModified = lastchar2 +2;
                        }

                        if (isRootLetter(lastchar2))
                        {
amharic.value = amharic.value.substring(0, startPos-1)+
String.fromCharCode(lastCharModified)+
amharic.value.substring(startPos, amharic.value.length);
                        }}}
                  range.collapse();
                  range.moveStart("Character", newPos);
                  range.select();
            }
            return false;
      }
      return true;
};
function convertEnglishConsonantToAmharic(lastChar, UniCodeValue,
shiftKey)         {
            var NewValue;
            var lastcharConsonant = lastChar;

            if( UniCodeValue == "q" )
```

```
{
      NewValue = 4672;
}else if ( UniCodeValue == "h" && shiftKey == true)
{
      NewValue = 4624;

} else if ( UniCodeValue == "k" && shiftKey == true)
{
      NewValue = 4792;
}else if ( UniCodeValue == "w" && shiftKey == false)
{
      NewValue = 4808;
} else if ( UniCodeValue == "r")
{
      NewValue = 4648;
}  else if ( UniCodeValue == "y")
{
      NewValue = 4840;
}else if ( UniCodeValue == "t" && shiftKey == true)
{
      NewValue = 4896;
}else if ( UniCodeValue == "t" && shiftKey == false)
{
      NewValue = 4720;
}else if ( UniCodeValue == "p" && shiftKey == false)
{
      NewValue = 4944;
} else if ( UniCodeValue == "s" && shiftKey == false)
{
      NewValue = 4656;
} else if ( UniCodeValue == "d")
{
      NewValue = 4848;
} else if ( UniCodeValue == "f")
{
      NewValue = 4936;
}else if ( UniCodeValue == "g")
{
      NewValue = 4872;
}else if ( UniCodeValue == "h" && shiftKey == false)
{
      NewValue = 4608;
}else if ( UniCodeValue == "j")
{
      NewValue = 4864;
}else if ( UniCodeValue == "k" && shiftKey == false)
{
      NewValue = 4776;
}else if ( UniCodeValue == "l")
{
      NewValue = 4616;
}else if ( UniCodeValue == "z" && shiftKey == false)
{
      NewValue = 4824;
}else if ( UniCodeValue == "z" && shiftKey == true)
{
      NewValue = 4832;
}else if ( UniCodeValue == "x" && shiftKey == false)
{
      NewValue = 4768;
}else if ( UniCodeValue == "x" && shiftKey == true)
{
      NewValue = 4920;
```

```
                }else if ( UniCodeValue == "c" && shiftKey == false)
                {
                      NewValue = 4728;
                }else if ( UniCodeValue == "c" && shiftKey == true)
                {
                      NewValue = 4904;
                }else if ( UniCodeValue == "v")
                {
                      NewValue = 4712;
                }else if ( UniCodeValue == "b")
                {
                      NewValue = 4704;
                }else if ( UniCodeValue == "n" && shiftKey == true)
                {
                      NewValue = 4760;
                }else if ( UniCodeValue == "n" && shiftKey == false)
                {
                      NewValue = 4752;
                }else if ( UniCodeValue == "m")
                {
                      NewValue = 4632;
                }else if (UniCodeValue == "p" && shiftKey == true) //[
                {
                      NewValue = 4912;
                } else if (UniCodeValue == "s" && shiftKey == true)
                {
                      NewValue = 4664;

                } else if( UniCodeValue == ",")
                {
                      NewValue = 4963;
                }else if( UniCodeValue == "-")
                {
                      NewValue = 4954;
                }

                if ( NewValue )
                {
                      return String.fromCharCode(NewValue);
                } else
                {
                      return "";
                }
};
function withinRange(numValue)
{
      if ( numValue >= 4608 && numValue <= 4951)
      {
            return true;
      } else{
            return false;
      }
};
function getCursorPosition(){
      var range;
      range = document.selection.createRange();
      range.moveStart("Textedit", -1);
      return range.text.length;
};
</script>
```

Declaration

Submitted by

_____	_____	_____
Student	Signature	Date

Approved by:

_____	_____	_____
Advisor	Signature	Date