UNIVERSITY OF THE PHILIPPINES MANILA
COLLEGE OF ARTS AND SCIENCES
DEPARTMENT OF PHYSICAL SCIENCES AND MATHEMATICS

VOCALHANDS: AN ONLINE LEARNING SYSTEM FOR THE AMERICAN SIGN LANGUAGE

A Special Problem in partial fulfillment
of the requirements for the degree of
Bachelor of Science in Computer Science

Submitted by:

Leo Angelo S. Quigao
April 2010

ACCEPTANCE SHEET

The Special Problem entitled "Vocalhands: An Online Tutorial System for the American Sign Language" prepared and submitted by Leo Angelo S. Quigao in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science has been examined and is recommended for acceptance.

Geoffrey A. Solano, M.S.
Adviser

EXAMINERS:

	Approved	Disapproved
1. Gregorio B. Baes, Ph.D. (candidate)		
2. Avegail D. Carpio, M.S.		
3. Richard Bryann L. Chua, M.S.		
4. Aldrich Colin K. Co, M.S. (candidate)		
5. Ma. Sheila A. Magboo, M.S.		
6. Vincent Peter C. Magboo, M.D., M.S.		
7. Bernie B. Terrado, M.S. (candidate)		

Accepted and approved as partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science.

Geoffrey A. Solano, M.S.
Unit Head
Mathematical and Computing Sciences Unit
Department of Physical Sciences and Mathematics

Marcelina B. Lirazan, Ph.D.
Chair
Department of Physical Sciences and Mathematics

Reynaldo H. Imperial, Ph.D.
Dean
College of Arts and Sciences

ABSTRACT

Sign Language is the primary language used by the Deaf. It is also used by their parents, children and many of those who find themselves constantly communicating with its primary users. Because of this, Sign Language is the 4th most widely used language worldwide. It gives the Deaf the opportunity to communicate with other people with a language they can use. Because of this, learning the language is essential to Deaf children and their families.

The availability of an online learning system for Sign Language would definitely be helpful for the families of the Deaf who can't afford to learn outside the home. There are available online tools for learning Sign Language. However, most of them fall short of useful features and some are not free. These tools provide signed words and phrases through recorded videos which are not feasible for combining to form sentences. There have been studies made in translating the English language to American Sign Language, but there is none available online for practical use.

Vocalhands, an online learning system for the American Sign Language, addresses these concerns by providing fully featured tutorials, including dictionaries, tests and English to ASL translator. They are presented in a form of a Content Management System to ease the administration for its back-end users, the Sign Language consultants, the Flash Animators and System Administrators.

The Vocalhands translator uses Lexical Functional Transfer to successfully translate English to ASL with syntactic accuracy. The transfer is governed by a finite manageable group of translation rules and is processed through Grammar parsing and XML manipulation.

Keywords: Sign Language, E-learning, Content Management System, Machine Translation

TABLE OF CONTENTS

I. INTRODUCTION

A. Background of the Study

It has been told that symbols and gestures were the first forms communication that our human ancestors learned and used when they first walked the Earth. These forms of communication have since then been an important if not critical social skill we possess. This fact is truly more applicable to many of those who rely only on these symbols and gestures to ‘talk’ with others. There are people around us who are either deaf or mute. Losing the sense of hearing or being unable to talk and communicate is potentially a heavy burden to every individual experiencing it. However, the use of sign language eases the burden they receive by giving them the opportunity to relate with others by communicating with the language they can use. It is their primary language. Sign Language is not only used by the Deaf but also by their parents, children and those who find themselves in contact with the Deaf. An inexhaustible list includes teachers, doctors, health care professionals, social workers and many others. And because of this, sign language is the fourth most used language worldwide [1].

The Deaf in any group comprise from one to three individuals in every 1000 births (Parkhurst, 1997). Most Deaf children (more than 90%) are born into families where both parents are hearing (Mitchell, 2005). They do not learn to speak because they cannot hear, and they do not learn to sign because there is no one to teach them. Most deaf worldwide grow up without any language. [2]

Educating the deaf and speech impaired have been a special task and concern of many existing institutions and governments. And obviously, sign language is an important skill every one of them has to master. There are several ways it can be taught, ranging from specialized classrooms, to video instructional materials, to simple flashcards. With the proliferation of computers, these too have also been media to learn sign language. Programs have been developed to teach its users this language.

However the technologies' currently more ubiquitous innovation, the internet, has not been fully utilized to provide sign language tutorials.

In the Philippines, there are about three kinds of sign language. Signed Exact English (SEE) is the oldest used of the three and it translates English text almost exactly, it is manually performed English. It is said to be awkward to use, and is therefore becoming unpopular. The second one is Filipino Sign Language (FSL). It makes use of signs to represent Filipino words. However, its use is limited and is commonly found in households. The third and most significant is the American Sign Language (or ASL, Ameslan for short). It is also the dominant sign language in most countries [3]. ASL destroys the misconception that sign language is only manually performed English. It is a full language, with its own set of syntax, punctuation and grammatical rules [1].

Sign language is not merely a visual representation of another language, say English, used for convenience by those who cannot hear. It is a natural language, arising in communities of Deaf people just as spoken languages arise in communities of hearing people. It has linguistic structure, idiom, metaphor, etc.; it can represent abstract as well as concrete ideas. As a linguistic medium, it is in no way impoverished or restricted in the type or complexity of ideas that it can convey. [4]

In times when hospital situations arise for Deaf patients, it helps them to properly convey what they want to say or what they feel. Doctors, nurses, and health care professionals, who commonly interact with the Deaf, have problems comprehending their patients because of their lack of skill to understand their patients' language.

American Sign Language is composed precise handshapes, palm positions, and the use of the space around the signer. Together with facial expressions and body language it is able to convey subtle, complex and or abstract ideas [1]. Its standardization was started by Laurent Clerc and Thomas H. Gallaudet back in 1817 and has been the essential link of the deaf culture ever since [5].

B. Statement of the Problem

Learning ASL in the present day is not a trivial task. Not all schools and academic institutions have a course that teaches sign language. Even finding someone who is willing to teach sign language is not too easy. For ease and convenience, sign language can be learned even without going to school and being online can possibly do that. The Information Superhighway has been around for a time now but sadly its boundaries are yet to be fully explored concerning the field of ASL.

Dictionaries for reference to American Sign Language have been available in the internet (ASLpro.com for example) but so comprehensive as these dictionaries could be, there is no proper tutorial that teaches it in a systematic manner. One will find it hard to learn from these dictionaries as it is not described what is best to learn first. Tutorials on the other hand will have to be paid to access their services and are incomplete as they lack dictionaries and translators. In other words there is still no fully-featured sign language tutorial available.

Signs are often depicted as static images or as motion-captured videos. It has been available in many formats, including GIF, Quicktime Video, and Flash Video. However a similar problem with them is that the videos themselves are inconsistent in terms of who is signing, and also where and when the video is taken. The inconsistency makes the sign representations remove their flexibility or the potential to be used more than just a dictionary entry. Putting together the videos/images to form sentences is almost unfeasible because an end result would be seeing a sentence performed by different persons or at different times.

Additionally, there is no available ASL translator of any kind online. Many studies have been made with machine translating English to American Sign Language, and though some succeeded in varying degrees, none of them were published online.

C. Objectives of the Study

Generally, this study aims to provide its users the opportunity to learn sign language in the internet, by undergoing systematic tutorials. These users include the following: the deaf, mute, their parents or children, or anyone who needs or wants to learn American Sign Language.

To appropriately deliver this aim, this project has the following functionalities:

I. Allow its guest users to:
   a. View news & articles
   b. Take tutorials for learning ASL
   c. Read the dictionary consisting of ASL words with cross references to similar, but more complicated words. For example, anxious is a synonym of worried and both will have the same sign language representation. The dictionary will be categorized alphabetically.
   d. Search for words where users can input a word and the system will display the ASL equivalent if available.
   e. Translate English sentences to American Sign Language using a translator that is especially inclined to medical and health care use.
   f. Take evaluation tests for the users to know their ASL proficiency.

II. Allow the administrators to:
   a. Consultants
      i. View, create and update articles
      ii. View, create and update lessons
      iii. Create, update, and delete courses
      iv. Create and update simple words
      v. Create and update complex words

vi. View, create, and update grammar rules

vii. Set exam properties

viii. Send and read private messages to/from other administrators

b. Flash animator

i. Reserve words for animation development

ii. Free up a word he has reserved

iii. Upload sign language animation for a word

iv. Send and read private messages to/from other administrators

c. Sign Language Administrator

i. View, approve and unpublish articles

ii. View, approve and unpublish lessons

iii. View, approve, unpublish and change (animation) development status of simple words

iv. Approve and unpublish complex words

v. View, approve and unpublish grammar rules

vi. Update 'about us' site information

vii. Send and read private messages to/from other administrators

d. Site Administrator

i. Create, update and delete user information

ii. Send and read private messages to/from other administrators

D. Significance of the Study

The availability of online tutorials allows everyone who wishes to learn ASL to do so wherever possible, especially in the comfort of their own homes. This is particularly useful for the parents and relatives of the Deaf, as they can take courses in ASL online and thus eases the learning process.

Vocalhands, as medical information tool helps the Deaf interact well in a hospital/medical facility and convey what he/she needs. This can also teach doctors, nurses and health care professionals to understand what their Deaf patients want to say.

Vocalhands presents a learning environment that is free and yet fully-featured with its tutorials, dictionary, translating tool and evaluation tests. It can teach signs in a systematic way such that the user will easily discern what to learn first and what to learn next.

The flash animation format of Vocalhands' signed words allows many different flash animators to develop signed animations for the website, as the flash SWF format is already popular and there are many animators available. The use of a central template in animations allow consistency and therefore reduce 'noise' experienced in existing online dictionaries, more so when stringing several animations is needed.

Vocalhands' translating tool is a first among publicly available online tutorials for Amslang. Taking previous studies surrounding the topic in mind, it will give its user the answer to the recurring question, "what would be the signed equivalent for this sentence" and aids the learning experience.

The development of an online tutorial for learning American Sign Language can be a valuable tool for those who need to learn ASL, especially the deaf and speech impaired. It can also be important to those who communicate to them like their parents, children or relatives. The language has been called the bridge of the deaf culture because of its significance to its community. It is also dedicated to the families of Deaf people, so that they may be able to communicate with them in the best ways possible. It is useful for those who aspire to use ASL in their daily work routines.

Being able to master this language will prove to be essential to interact socially with other people, and ultimately live normal lives as human beings.

E. Scope and Limitations

1. This study only covers hand and arm movement in its animations, and thus does not include non-manual signs. The inclusion of these will make translation and animation creation much more complex. However, tutorials for these can still be made available in the form of tips and guidelines.
2. Because of the customizable nature of the tutorial system, the content of the courses also has to depend on the administrators handling the system.
3. The strength of translations depends on the robustness of the dictionary and the completeness of grammatical rules present in the database.
4. The translator can only interpret and translate sentences up to syntax level – this means the semantic meaning of the sentence will not be in concern.
5. Words/sentences shall be interpreted one by one only when translating English to ASL.
6. For the purpose of this study, ASL sign animations only include words that are used in FAQs (frequently asked questions) on the hospital setting.

F. Assumptions

1. The courses are in beginner, intermediate and advanced difficulties. The user does not need to finish an easier course to advance to another one.

2. It is assumed that consultants are experts in their field and therefore have knowledge on grammar, sentence structure and parts of speech. They should also understand Labeled Bracket Notation.

II. REVIEW OF RELATED LITERATURE

This review will pass through the current facts and issues surrounding this study, as well as the past and present systems that have features similar to the one in this project. This review will also introduce the concepts and framework that is going to be used in this system in relation to how they were utilized in their respective past studies.

A 2004-2005 report on sign languages of the Philippines by Hope M. Hurlbut presented us the current facts concerning the Deaf in the Philippines. [2]

There are now about thirty elementary schools which have classes for Deaf children. Some of them are exclusively for Deaf children, whereas some are mixed with handicapped children with various disabilities. These schools have elementary classes from grades 1–6. In addition ten or more secondary schools consist of four years of education, and some Bible Institutes are wholly or partially for Deaf students. At the tertiary level there are classes for the Deaf in at least ten colleges, institutes and universities, and two of these are exclusively for Deaf students. Some of the tertiary institutions grant degrees up to the MA level.

There are at least thirty training and vocational centers for Deaf and disabled run by churches, other private groups and foundations, and another fourteen under government sponsorship. These are mostly geared to training adults. Some of the schools use the oral approach, that is, they try to teach the children to speak and read lips. This is a difficult task for profoundly deaf children who cannot hear the spoken word, but it is less of a problem for the hard of hearing children.

Signing Online is a commercial website that provides tutorials for learning ASL. It features 4 courses (each course requiring $49.95 for 4 months of access) for learning ASL and it also has a dictionary as well as fingerspelling and number signing references. [6]

ASL University is an online curriculum resource for American Sign Language students, instructors, interpreters, and parents of deaf children. It offers free lessons. These lessons are grouped into parts. At the end of each part, there is an evaluation quiz. It also features a dictionary. Sadly, each sign is only represented by image clips, and most of the time, the clips are not understandable. [7]

Michigan State University's CommTechLab provides an online dictionary for ASL. The dictionary is grouped alphabetically. Each animation is represented by a motion-captured Quick Time video. Since its conception in 1997, it now houses thousands of ASL signs for reference. [8]

A Medical ASL Phrasebook has been published by Eileen Carpenter, MD to list the important signs and know-hows of communicating with the Deaf in medical institutions. The Phrasebook has divided into several categories on where a certain sign can be used (e.g. eyes are found in the anatomy category). There is also a list of commonly used phrases in hospitals and the corresponding ASL translations for them. There is also a short description of ASL's grammar, as well as several tips and guidelines on serving Deaf patients. [9]

The website relies on MSU's ASL Browser to display its sign animations. Each animation also has a written explanation on how it is performed.

Grieve-Smith developed English to American Sign Language Machine Translation application that is specifically used for Weather Reporting. The application consists of four components: a lexical analyzer, a parser, a transfer module, and a generation module, all typical in a machine translator. [10]

The lexical analyzer tags each text with a particular lexical category, for example, "today" is tagged with "<day>".

After building a parse tree for the text, the elements of the tree are looked-up in a table containing the translated ASL written in Newkirk notation and then the parse tree is recreated.

The generation module takes an inventory of the kinds of information present in the semantic representation, and generates a formulaic phrase for each one. These formulas all use ASL grammar, including topic-comment structure and non-manual grammatical morphemes. The content that is output by the transfer module is then plugged in to the formulas, producing fluent ASL.

Huenerfauth's dissertation has created an English-to-ASL MT design capable of producing classifier predicates. The classifier predicate generator inside this design has a planning based architecture that uses a 3D “visualization” model of the arrangement of objects in a scene discussed by the English input text. This generator would be one pathway in a multi-path English-to-ASL MT design; a separate processing pathway would be used to generate classifier predicates, to generate other ASL sentences, and to generate animations of Signed English (if the system lacked lexical resources for some input). [11]

Instead of representing the ASL animation as a string (of individual signs to perform), this system encodes the multimodal language signal as multiple channels that are hierarchically structured and coordinated over time.

As part of the European Union’s ViSiCAST project, Ian Marshall and Éva Sáfár at the University of East Anglia implemented a system for translating from English text into British Sign Language. The system was also considered a research vehicle for translation to German or Dutch sign languages, but that capability has not yet been implemented. Their approach uses the CMU Link Parser to analyze an input English text, and then uses Prolog declarative clause grammar rules to convert this linkage output into a Discourse Representation Structure (DRS). During the generation half of the translation process, Head Driven Phrase Structure rules are used to produce a symbolic sign language representation script. This script is in the system’s proprietary Signing Gesture Markup Language, a symbolic coding scheme for the movements required to perform a natural sign language. [12]

The ZARDOZ system [Veale et al. 1998] was a proposed (and somewhat prototyped) English-to-Sign-Languages translation system using a set of hand-coded schemata as an Interlingua for a translation component. (The discussion herein will focus on the ambitious proposed architecture rather than the implemented portion.) While the implemented portion of the system focused on American Sign Language, the authors were developing their framework with British, Irish, and Japanese Sign Language also in mind. Some of the research foci of this system were the use of AI knowledge representation, metaphorical reasoning, and blackboard system architecture; so, the translation design is very knowledge and reasoning heavy. During the analysis stage, English text would undergo sophisticated idiomatic concept decomposition before syntactic parsing in order to fill slots of particular concept/event/situation schemata. The advantage of the logical propositions and labeled slots provided by schemata architecture was that commonsense and other reasoning components in the system could later easily operate on the semantic information. Because of the amount of hand-coding needed to produce new schemata, the system would only be feasible for limited domains. To compensate for these limitations, if a schema did not exist for a particular input text, the system would instead perform sign-for-word transliteration. [13]

TEAM was an English-to-ASL translation system built at the University of Pennsylvania that employed Synchronous Tree Adjoining Grammar rules to build an ASL syntactic structure while an English dependency tree was built during analysis [Zhao et al. 2000]. The output of the linguistic portion of the system was a written ASL "gloss notation with embedded parameters" that encoded limited information about morphological variations, facial expressions, and sentence mood. This project took advantage of the virtual human modeling research also being performed at the University of Pennsylvania by using one of the Human Modeling and Simulation laboratory's animated virtual humans as the signing avatar. While the extensibility of the system's STAG translation approach can be questioned because of the simplifying assumptions inherent to the system's "gloss" notation, the project had particular success at generating aspectual and adverbial information in ASL using the emotive capabilities of the animated character. [14]

In his dissertation, Speers described an approach to designing a machine translation system that generates a representation of American Sign Language (ASL) from English. He represented ASL based on the Move-Hold (MH) model (Liddell and Johnson 1989), a sign notation system that allows for both precision of sign description and predictable variation of surface forms based on grammatical detail. Empty features are used in MH notations of lexical forms, which are instantiated with spatial data during generation. [4]

The generation system is implemented as an LFG correspondence architecture (Kaplan and Bresnan 1982, Kaplan et al 1989). Correspondence functions are defined that convert an English f-structure into an ASL f-structure; build an ASL c-structure from the f-structure; and build the phonetic representation level (p-structure, where spatial and non-manual variations are revealed) from the c-structure. The concepts presented in his dissertation have been implemented in a software application, ASL Workbench.

Borra, et al. developed an LFG-based machine translation engine developed for an English-Filipino bi-directional translator. The whole engine includes the analysis to f-structure, transfer of source to target f-structure, and generation from f-structure. Initial linguistic resources were established to test the engine and to develop the full bidirectional English-Filipino machine translator system. [15]

These linguistic resources include the formal grammar rules for the English and Filipino language, mono-lingual dictionaries for both languages and the transfer dictionaries, which include transfer rules (structural level) and transfer dictionary (word level). Testing involved subjecting the system to different sentences and sentence constructions in both languages (English and Filipino). Results show that translation quality is extremely dependent on the available linguistic resources.

Kudo and Nomura presented a transfer framework called LFT (Lexical-functional Transfer) for a machine translation system based on LFG (Lexical-functional Grammar). The translation process consists of sub-processes of analysis, transfer and generation. We adopt the so called f-structures of LFG as the intermediate representations or interfaces between those sub-processes, thus the transfer process converts a source f-structure into a target f-structure. Since LFG is a grammatical framework for sentence structure analysis of one language, for the purpose, we propose a new framework for specifying transfer rules with LFG schemata, which incorporates corresponding lexical functions of two different languages into an equational representation. The transfer process, therefore, is to solve equations called target f-descriptions derived from the transfer rules applied to the source f-structure and then to produce a target f-structure. [16]

Dan Klein presented a novel generative model for natural language tree structures in which semantic (lexical dependency) and syntactic (PCFG) structures are scored with separate models. This factorization provides conceptual simplicity, straightforward opportunities for separately improving the component models, and a level of performance comparable to similar, non-factored models. Most importantly, unlike other modern parsing models, the factored model admits an extremely effective A* parsing algorithm, which enables efficient, exact inference. [17]

Dan Klein and Christopher Manning demonstrated that an unlexicalized PCFG can parse much more accurately than previously shown, by making use of simple, linguistically motivated state splits, which break down false independence assumptions latent in a vanilla treebank grammar.[18]

III. CONCEPTUAL/THEORETICAL FRAMEWORK

A. Deaf

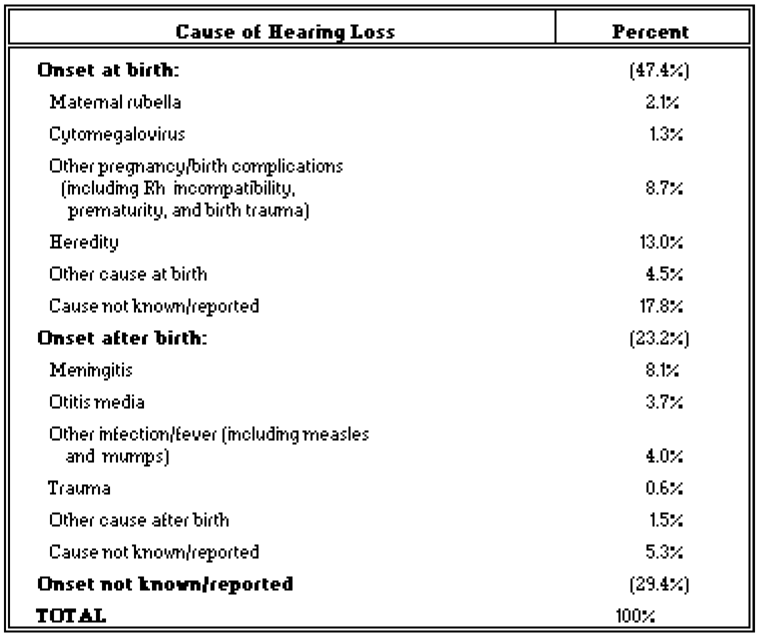
Clinical and legal definitions of deafness

In a clinical context, the term "deaf" (written with a lower case d) refers to a physical condition characterized by a severe or total lack of auditory sensitivity to sound.[1] Within the law, deafness is categorized by the degree of hearing loss, although the word "deaf" is usually reserved for the most severe hearing losses. For lesser degrees, the term "hearing loss" is used. These degrees include profound or total deafness (90 dB - 120 dB or more of hearing loss), severe hearing loss (60 dB - 90 dB), moderate hearing loss (30 dB - 60 dB), and mild hearing loss (10 dB - 30 dB). [18]

Woodward (1972) first proposed the convention of using capital-D "Deaf" to refer to the linguistic and cultural community of deaf people who use sign language, and the word "deaf" with a lower-case "d" to refer to people with the audiological condition of deafness. This convention has been used in many publications ever since. [2]

Etiology information is available for approximately one-half of the students reported to the 1992-93 Annual Survey of Hearing Impaired Children and Youth, conducted by the Center for Assessment and Demographic Studies. It is estimated that this survey represents 60-65% of the population of deaf and hard-of-hearing students in the U.S. who receive special education services. [19]

Heredity, at 13%, is the leading known cause of hearing impairment at birth, followed by pregnancy/birth complications (including Rh incompatibility, prematurity, and birth trauma) at 8.7%. Meningitis, at 8.1%, is the leading known cause of hearing impairment occurring after birth.



Cause of Hearing Loss	Percent
Onset at birth:	(47.4%)
Maternal rubella	2.1%
Cytomegalovirus	1.3%
Other pregnancy/birth complications (including Rh incompatibility, prematurity, and birth trauma)	8.7%
Heredity	13.0%
Other cause at birth	4.5%
Cause not known/reported	17.8%
Onset after birth:	(23.2%)
Meningitis	8.1%
Otitis media	3.7%
Other infection/fever (including measles and mumps)	4.0%
Trauma	0.6%
Other cause after birth	1.5%
Cause not known/reported	5.3%
Onset not known/reported	(29.4%)
TOTAL	100%

Source: 1992-93 Annual Survey of Hearing Impaired Children and Youth, Center for Assessment and Demographic Studies, Gallaudet University.

Figure 3.1: Percentage of deaf people categorized by cause of deafness

B. Sign Language

Sign language, gestural communication used as an alternative or replacement for speech. Sign languages resemble oral languages in every way other than their modality. As with oral languages, sign languages are acquired spontaneously and have highly intricate, rule-governed grammar and phonology. The three classes of features that make up individual signs are hand configuration, movement, and position to the body. Sign languages include those of Trappist monks, who have a rule of silence, and Plains Indians, where speakers of mutually unintelligible languages communicated freely. Australian aborigines and people of Sudan and the Sahara also have a complete sign language. Many languages have conventionalized body gestures elaborated to accompany or supplement speech, e.g., the Neapolitan gesture language. [20]

The widely used manual language of the deaf, or language of signs, was first systematized in the 18th century by the French abbé Charles Michel de l'Épée. It was brought to the United States by T. H. Gallaudet. As with any sign language, only a small percentage of signs suggest the form of thought they represent. Such sign languages also may have a syntax and grammar that differs dramatically from the language spoken locally. This is true, for instance, of American Sign Language, which, developed for the deaf, is a non-English system used in the United States and parts of Canada. A number of written systems for representing manual languages have been developed, and dictionaries of signs have been compiled. Often sign language is taught along with speech-reading and with a manual alphabet, i.e., a method of forming the letters of the alphabet by fixed positions of the fingers in the air.

C. American Sign Language

American Sign Language, known as ASL, is the natural native language of the American Deaf community. ASL is used as the primary form of communication in the daily lives of the Deaf. ASL is a full language with its own syntax, punctuation and grammar. American Sign Language is composed of precise handshapes, palm positions, movements, and the use of space around the signer. [1]

These elements, movements, and handshapes, supported by facial expressions and body language are capable of conveying complex and abstract ideas. ASL is constantly evolving and often changes regionally. The following combined elements serve to make ASL an exciting, effective form of communication.

- ASL signs
- Limited fingerspelling
- Facial expressions
- Body language
- Head movement
- Use of space and directional movement

A. American Sign Language Grammar

The American Sign Language is a full language and as such, its grammar should not in any way be derived from English. This section shall describe good compliance rules in translating to ASL.

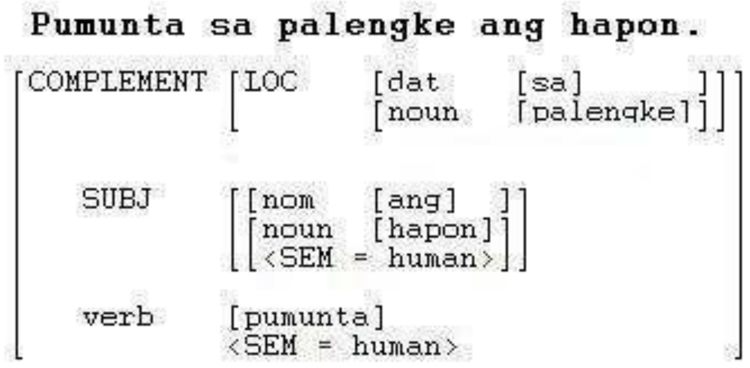
ASL verbs don't vary with tense, person or number -- the time something occurred when one begins signing is indicated first, and all verbs are assumed to be in agreement until you change the time with a new sign. Nouns, adjectives and verbs that are derived from each other and have variant forms in other languages (flight, flying, fly) usually have the same or very similar signs in ASL, and there are minimal changes for gender. Plurals are formed by repeating the last movement of a sign. Pronouns are simpler yet more specific -- when one mentions the noun he is going to talk about, it is pointed to a place in the space around signer, and that place will stand for that noun until it is redefined. There is usually no need to use the words "a," "an," or "the”. There are not dozens of synonyms with minor variations in meaning, since the facial expression and body language of the signer modifies the word being signed. Adverbs and adjectives are often unnecessary altogether; since the way the verb or noun signed they refer to tells something about it already. [9]

Examples:

- “The cat chases the rat” becomes “cat chase rat”
- “Filipino morals are deteriorating quickly” becomes “Filipino moral quick weaken”
- “The dog is sleeping under the table” becomes “under table dog sleep”.
- “I ate at Jolibee yesterday” becomes “Yesterday Jolibee I eat”.
- “Why is the world round?” becomes “round world why?”
- “I flew the airplane for a few minutes” becomes “(Past) Few minutes, I fly airplane.”
- “I am late and I still need to brush my teeth” becomes “I late. Brush my teeth I still need.”

B. Lexical Functional Grammar (LFG)

A framework for sentence structure analysis and has a simple framework for representing lexical and grammatical information. It analyzes a sentence in two steps, a phrase structure analysis and a functional structure analysis. C-structure is a syntactic analysis and produces constituent structures.The f-structure consists of several procedures, attaching lexical functions to components in the c-structure to produce a functional structure. [16]

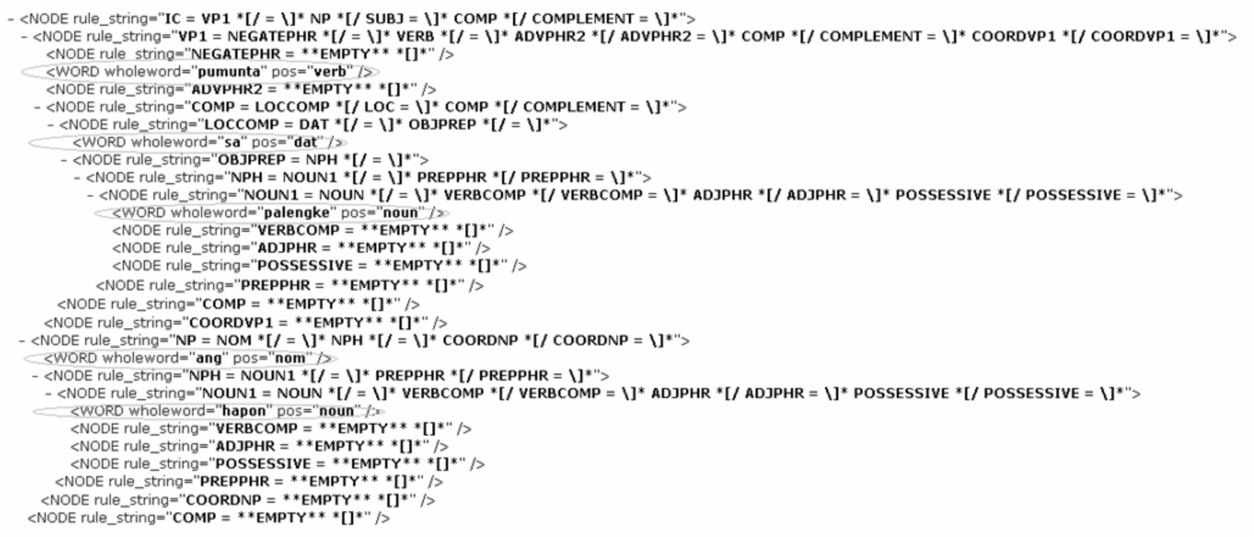


```
Pumunta sa palengke ang hapon.
[COMPLEMENT [LOC    [dat    [sa]       ]]]
            [       [noun   [palengke]]]]

    SUBJ    [[nom    [ang]   ]]
            [[noun   [hapon]]]
            [[<SEM = human>]]

    verb    [pumunta]
            <SEM = human>                   ]
```

Figure 3.2: An example on a sentence in F-Structure form



```
- <NODE rule_string="IC = VP1 *[/ = \]* NP *[/ SUBJ = \]* COMP *[/ COMPLEMENT = \]*">
  - <NODE rule_string="VP1 = NEGATEPHR *[/ = \]* VERB *[/ = \]* ADVPHR2 *[/ ADVPHR2 = \]* COMP *[/ COMPLEMENT = \]* COORDVP1 *[/ COORDVP1 = \]*">
      <NODE rule_string="NEGATEPHR = **EMPTY** *[]*" />
      <WORD wholeword="pumunta" pos="verb" />
      <NODE rule_string="ADVPHR2 = **EMPTY** *[]*" />
    - <NODE rule_string="COMP = LOCCOMP *[/ LOC = \]* COMP *[/ COMPLEMENT = \]*">
      - <NODE rule_string="LOCCOMP = DAT *[/ = \]* OBJPREP *[/ = \]*">
          <WORD wholeword="sa" pos="dat" />
        - <NODE rule_string="OBJPREP = NPH *[/ = \]*">
          - <NODE rule_string="NPH = NOUN1 *[/ = \]* PREPPHR *[/ PREPPHR = \]*">
            - <NODE rule_string="NOUN1 = NOUN *[/ = \]* VERBCOMP *[/ VERBCOMP = \]* ADJPHR *[/ ADJPHR = \]* POSSESSIVE *[/ POSSESSIVE = \]*">
                <WORD wholeword="palengke" pos="noun" />
                <NODE rule_string="VERBCOMP = **EMPTY** *[]*" />
                <NODE rule_string="ADJPHR = **EMPTY** *[]*" />
                <NODE rule_string="POSSESSIVE = **EMPTY** *[]*" />
              <NODE rule_string="PREPPHR = **EMPTY** *[]*" />
        <NODE rule_string="COMP = **EMPTY** *[]*" />
      <NODE rule_string="COORDVP1 = **EMPTY** *[]*" />
  - <NODE rule_string="NP = NOM *[/ = \]* NPH *[/ = \]* COORDNP *[/ COORDNP = \]*">
      <WORD wholeword="ang" pos="nom" />
    - <NODE rule_string="NPH = NOUN1 *[/ = \]* PREPPHR *[/ PREPPHR = \]*">
      - <NODE rule_string="NOUN1 = NOUN *[/ = \]* VERBCOMP *[/ VERBCOMP = \]* ADJPHR *[/ ADJPHR = \]* POSSESSIVE *[/ POSSESSIVE = \]*">
          <WORD wholeword="hapon" pos="noun" />
          <NODE rule_string="VERBCOMP = **EMPTY** *[]*" />
          <NODE rule_string="ADJPHR = **EMPTY** *[]*" />
          <NODE rule_string="POSSESSIVE = **EMPTY** *[]*" />
        <NODE rule_string="PREPPHR = **EMPTY** *[]*" />
      <NODE rule_string="COORDNP = **EMPTY** *[]*" />
    <NODE rule_string="COMP = **EMPTY** *[]*" />
```

Figure 3.3: F-Structure in XML form

C. Lexical Functional Transfer (LFT)

The LFT consists of both a representative framework for a two-way dictionary and transfer rules and a processing mechanism of transferring an f-structure of source language into an f-structure of target language. [16]

The whole process is divided into three sub processes, analysis, Transfer and generation as usual translation systems. The analysis process is nothing but LFG analysis.

The transfer process, LFT converts an f-structure of a source language into a corresponding f-structure of a target language. At first, a transfer dictionary is looked up and transfer rules are selected. Next, the conditions in the rule are checked. If they are satisfied, the schemata of target language in the transfer rule are instantiated. And then the functional descriptions of target language are obtained. They are called the target functional descriptions (target f-descriptions). After setting up the target f-descriptions, the task of the transfer process is reduced to solve them and then produce an f-structure of the target language. The processes of instantiation and solving target f-descriptions are the same mechanism within LFG.

The generation process is tentatively defined as a linearization process of the structured relationships in the target f-structure and a insertion process of inflected words.

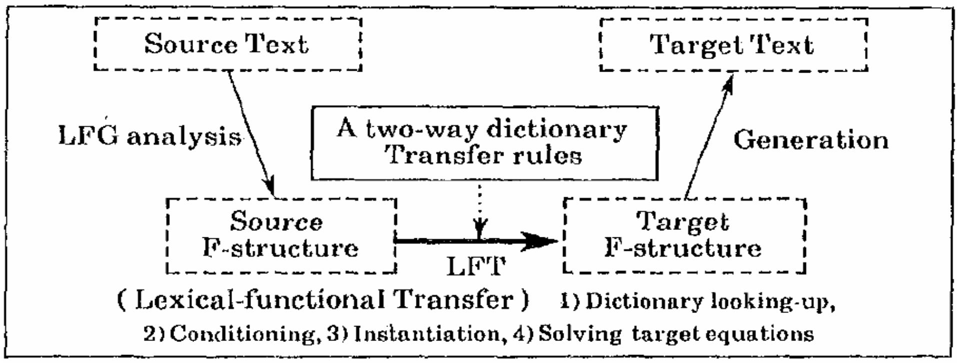

Source Text
Target Text
LFG analysis
A two-way dictionary
Transfer rules
Generation
Source
F-structure
Target
F-structure
LFT
(Lexical-functional Transfer) 1) Dictionary looking-up,
2) Conditioning, 3) Instantiation, 4) Solving target equations

Figure 3.4: Diagram for LFT. A transfer rules dictionary translates a source f-structure to its target f-structure.

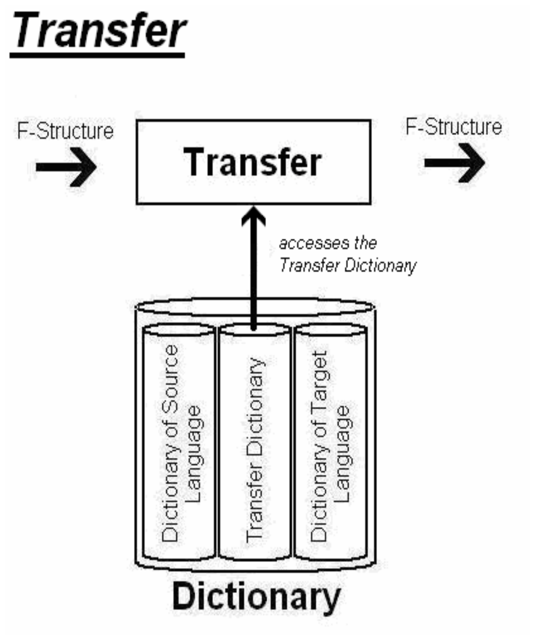

Transfer
F-Structure
Transfer
F-Structure
accesses the Transfer Dictionary
Dictionary of Source Language
Transfer Dictionary
Dictionary of Target Language
Dictionary

Figure 3.5: Diagram for Lexical Functional Transfer

D. Natural Language Processing

Natural Language Processing is the computer analysis and generation of natural language text. The goal is to enable natural languages, such as English, French, or Japanese, to serve either as the medium through which users interact with computer systems such as database management systems and expert systems (natural language interaction), or as the object that a system processes into some more useful form such as in automatic text translation or text summarization (natural language text processing). [21]

In the computer analysis of natural language, the initial task is to translate from a natural language utterance, usually in context, into a formal specification that the system can process further. Further processing depends on the particular application. In natural language interaction, it may involve reasoning, factual data retrieval, and generation of an appropriate tabular, graphic, or natural language response. In text processing, analysis may be followed by generation of an appropriate translation or a summary of the original text, or the formal specification may be stored as the basis for more accurate document retrieval later. Given its wide scope, natural language processing requires techniques for dealing with many aspects of language, in particular, syntax, semantics, discourse context, and pragmatics.

The first aspect of natural language processing, and the one that has perhaps received the most attention, is syntactic processing, or parsing. Syntactic processing is important because certain aspects of meaning can be determined only from the underlying structure and not simply from the linear string of words. A second phase of natural language processing, semantic analysis, involves extracting context-independent aspects of a sentence's meaning. Given that most natural languages allow people to take advantage of discourse context, their mutual beliefs about the world, and their shared spatio-temporal context to leave things unsaid or say them with minimal effort, the purpose of a third phase of natural language processing, contextual analysis, is to elaborate the semantic representation of what has been made explicit in the utterance with what is implicit from context. A fourth phase of natural language processing, pragmatics, takes into account the speaker's goal in uttering a particular thought in a particular way—what the utterance is being used to do.

E. Natural Language Parser

A natural language parser is a program that works out the grammatical structure of sentences, for instance, which groups of words go together (as "phrases") and which words are the subject or object of a verb. Probabilistic parsers use knowledge of language gained from hand-parsed sentences to try to produce the most likely analysis of new sentences. These statistical parsers still make some mistakes, but commonly work rather well. Their development was one of the biggest breakthroughs in natural language processing in the 1990s.

F. Brown Corpus

The Brown Corpus was a carefully compiled selection of current American English, totaling about a million words drawn from a wide variety of sources. Kucera and Francis subjected it to a variety of computational analyses, from which they compiled a rich and variegated opus, combining elements of linguistics, psychology, statistics, and sociology. It has been very widely used in computational linguistics, and was for many years among the most-cited resources in the field. It has a grammatical tag set that compiles known parts of speech.

G. Machine Translation

Machine Translation (MT) is the traditional and standard name for computerized systems responsible for the production of translations from one natural language into another, with or without human assistance. [22]

Systems are designed either for one particular pair of languages (bilingual systems) or for more than two languages (multilingual systems), either in one direction only (uni-directional systems) or in both directions (bi-directional systems).

IV. DESIGN AND IMPLEMENTATION

A. System Design

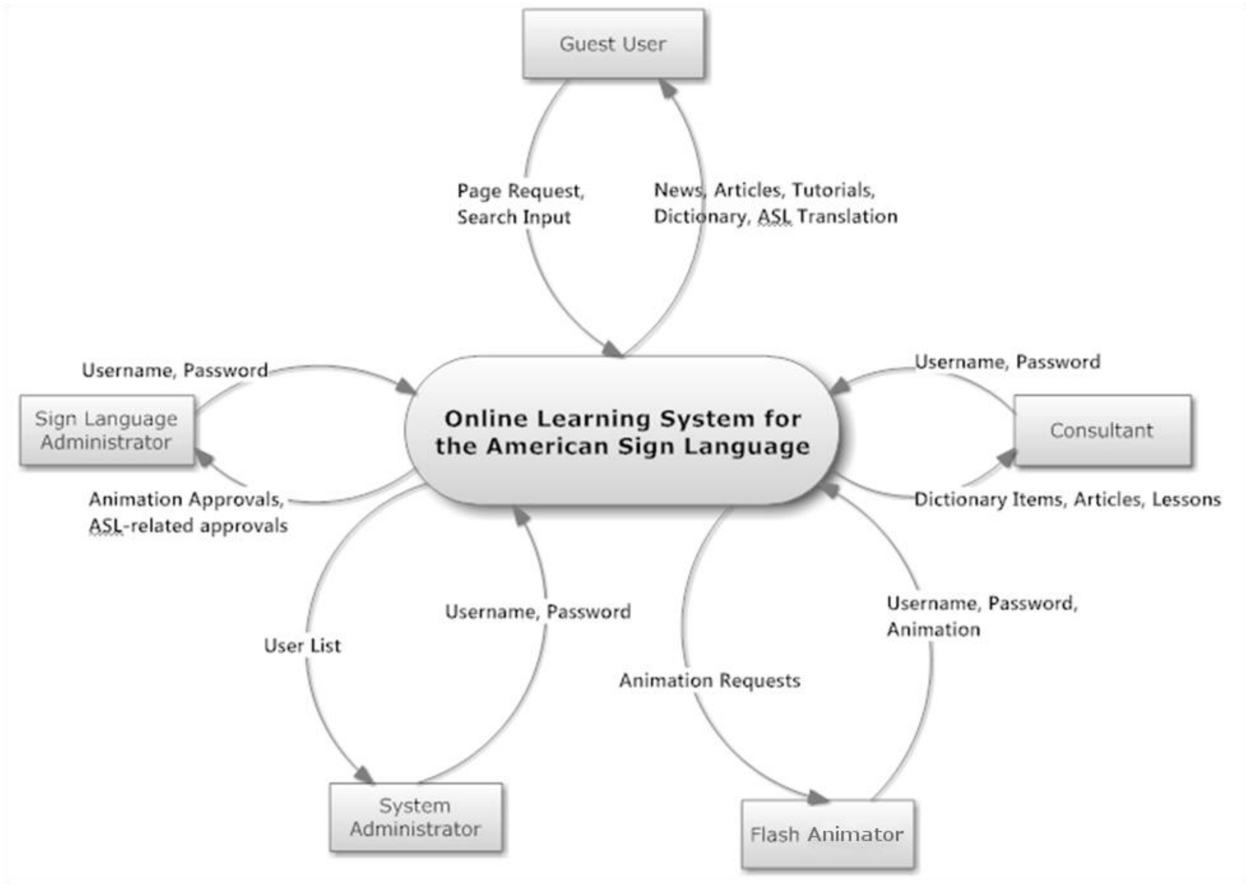

Guest User
Page Request, Search Input
News, Articles, Tutorials, Dictionary, ASL Translation
Username, Password
Sign Language Administrator
Animation Approvals, ASL-related approvals
Online Learning System for the American Sign Language
Username, Password
Consultant
Dictionary Items, Articles, Lessons
User List
Username, Password
System Administrator
Animation Requests
Username, Password, Animation
Flash Animator

Figure 4.1: Context Diagram for Vocal Hands

An overview of the whole Vocalhands system, as shown in figure 4.1, defines the types of users who will be using the system, as well as the flow of information between them and the system. The users are as follows:

- The guest is a site visitor. He is any unlogged user of the website. He can view site information and read news & articles. He is also given the privileges to take tutorials, self-assessment exams, read the dictionary and translate English to ASL.

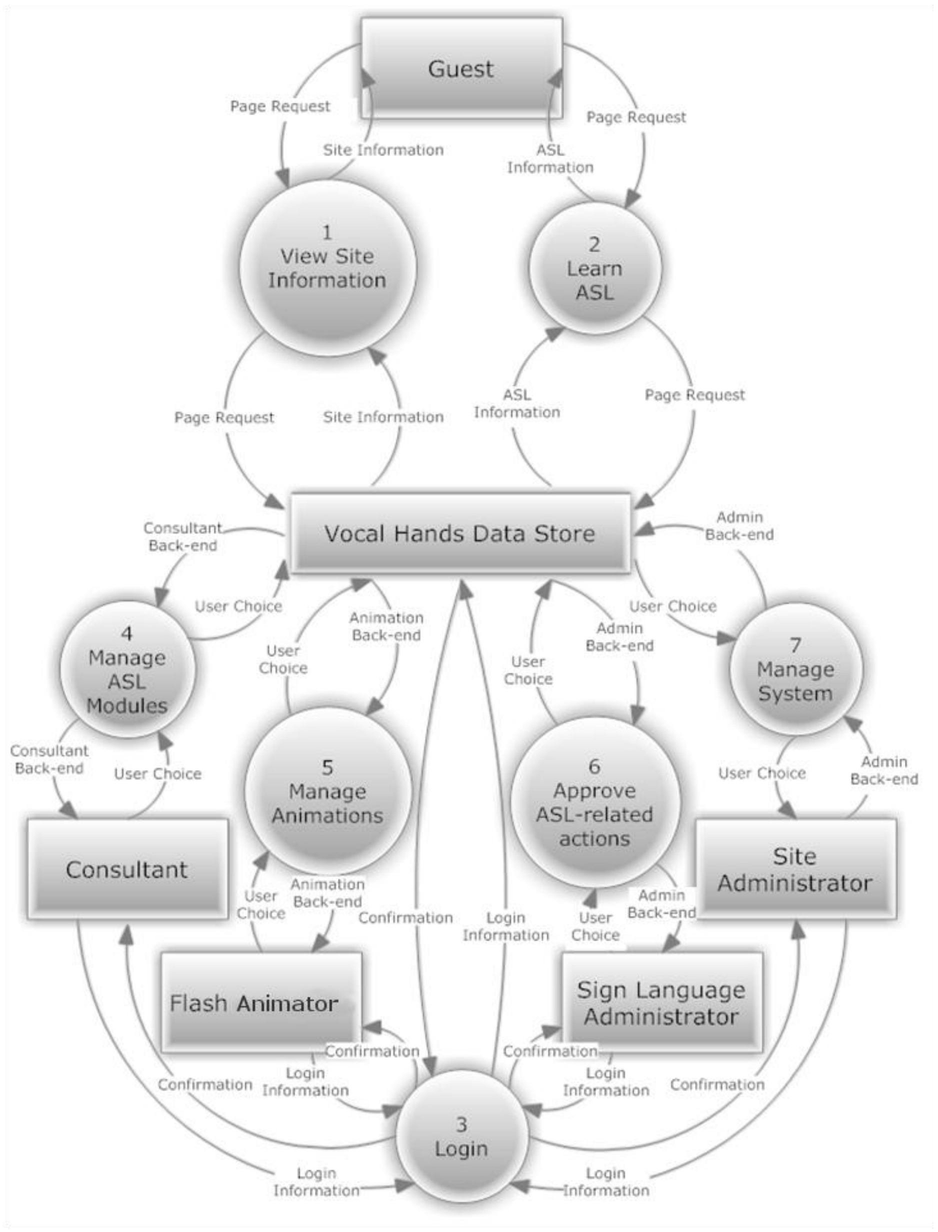


Figure 4.2: Top Level Data Flow Diagram for Vocal Hands

- A consultant is an expert in the field of ASL. He is given the task to manage tutorials, dictionaries and grammatical rules. The consultant may be one or more users.
- A flash animator is knowledgeable in flash animation and is tasked to create flash animations for each sign. He builds animation based on templates to preserve consistency. The flash developer type may also be entitled to several individuals.

- The <u>sign language administrator</u> ensures the quality and accuracy of flash animations and approves sign language-related actions made within the site. Because of this, he also must be an expert in the field of American Sign Language.
- The <u>site administrator</u> installs the system and manages it and also manages the site's users. He doesn't necessarily need to know ASL.

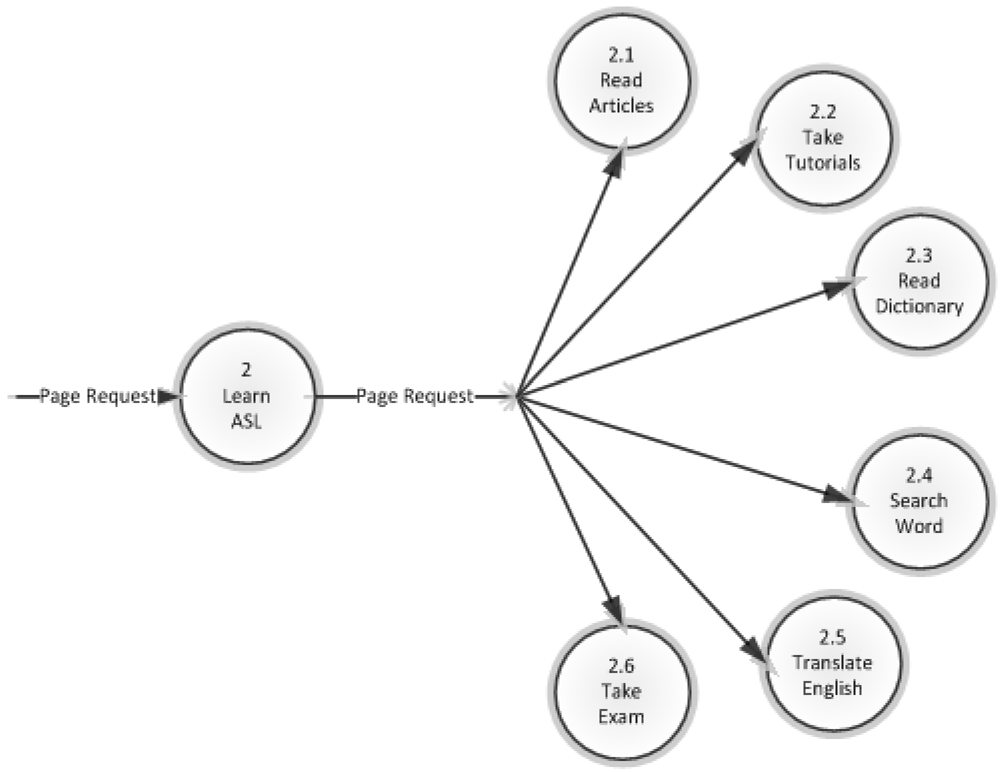


Figure 4.3: Sub-Explosion for 'Learn ASL'

'Read Articles' allow the user to read articles, news, and announcements related the American Sign Language and Vocalhands. The user may select from a list of articles and read it.

'Take tutorial' allows the user to learn ASL courses. Courses are divided by difficulty. Selecting a lesson can immediately take place after choosing 'Take Tutorials' because all categories are already expanded. From there, the user can read the content of the lessons.

‘Read Dictionary’ allows the user to browse the site’s dictionary or search for specific entries. The dictionary is divided into alphabetical categories. After selecting a letter, the user may select a word that starts with the said letter.

‘Search Word’ allows the user to look for a word for its ASL counterpart. It will be a substring search which means the query will accept substrings of words.

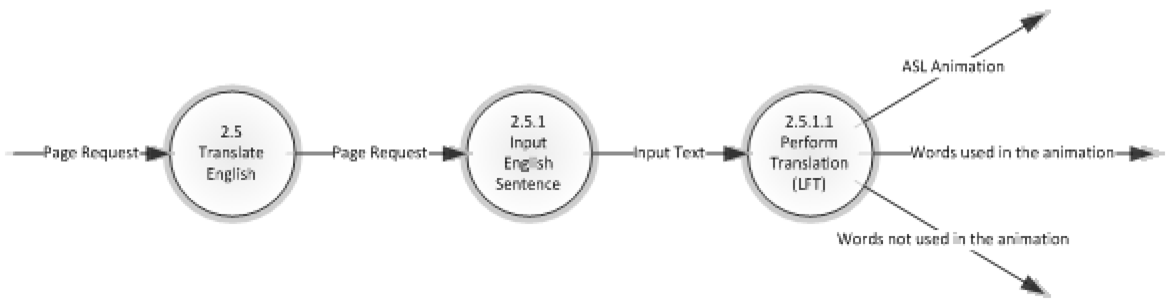


Figure 4.4: Sub-Explosion for ‘Translate English’

‘Translate English’ allows the user to input English text and the system will translate it to ASL. It makes use of Lexical Functional Transfer which will be discussed in a latter part.

‘Take exam’ allows the user to take self-assessment exams. They differ by level of difficulty. The exam presents the user with random multiple choice and true or false questions, which the user must answer. After a number of questions, the quiz will assess the score the user achieved.

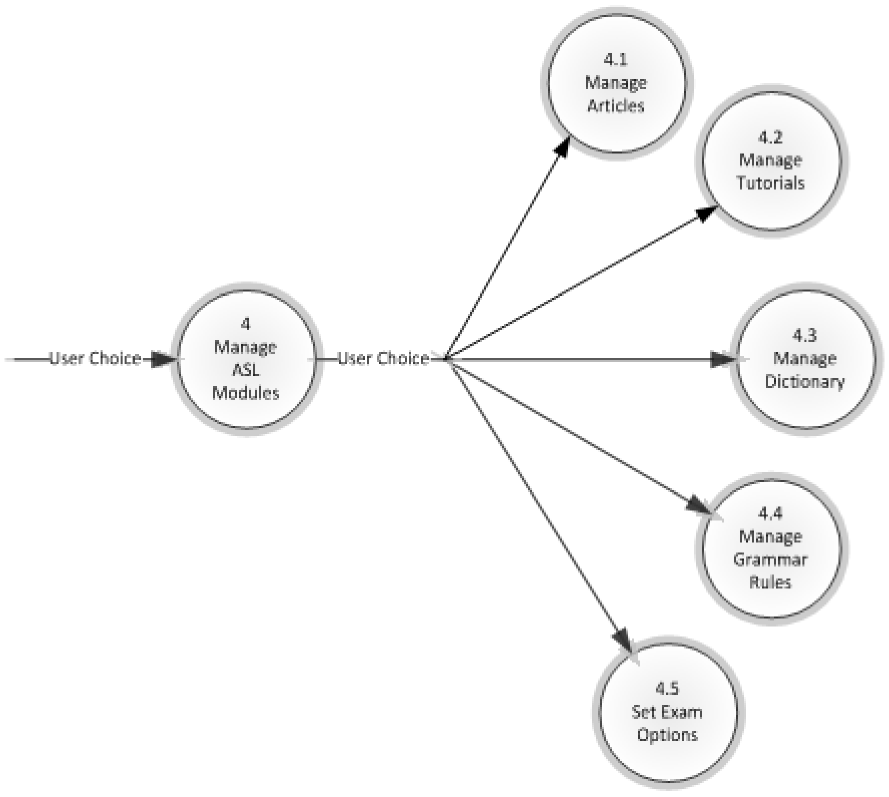


Figure 4.5: Sub-Explosion for 'Manage ASL Modules'

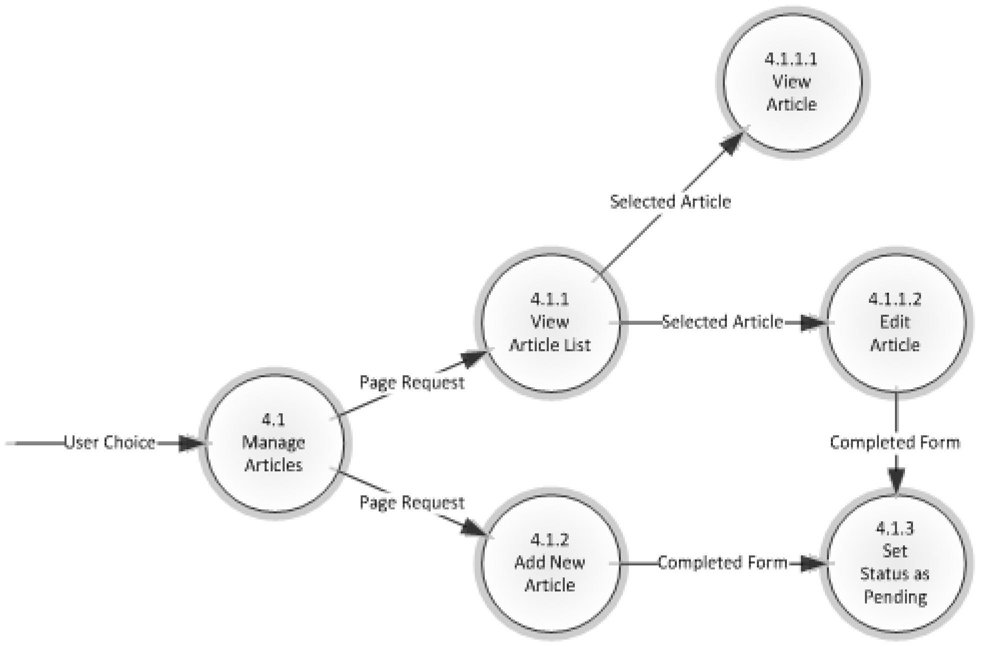


Figure 4.6: Sub-Explosion of 'Manage Articles'

The content of articles will make use of WYSIWYG editors for it to be fully customizable. New and edited articles shall be subject to approval by the sign language administrator.

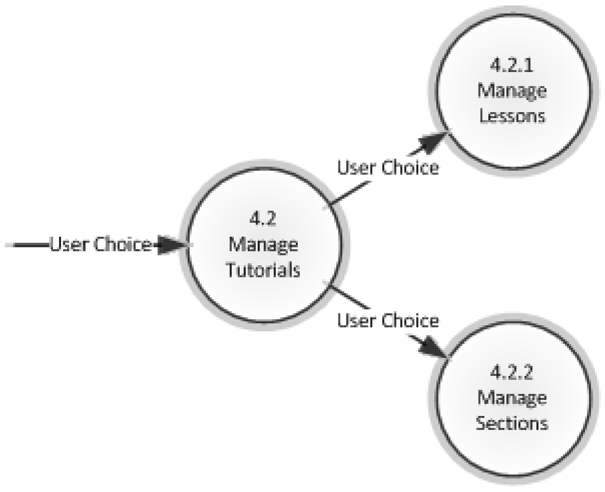


Figure 4.7: Sub-Explosion for 'Manage Tutorials'

'Manage lessons' is applicably the same as 'Manage Articles'. It allows viewing, adding and updating of lessons, with the inclusion of reordering lessons. Users can add links to words in the dictionary inside the content of a lesson. Each lesson must be a member of a section (difficulty).

'Manage Sections' allows the addition, edit, reordering and deletion of sections. Deleting a section does not necessarily mean deleting related lessons.

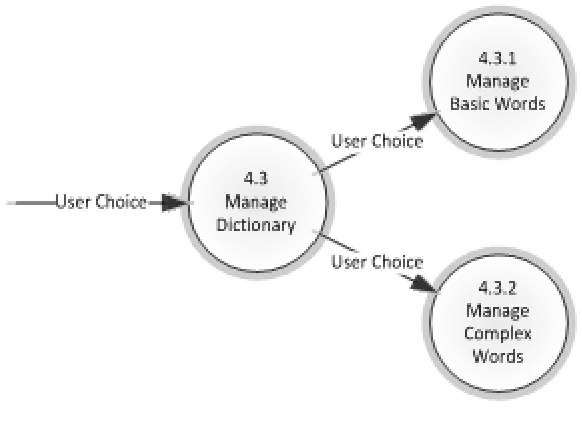


Figure 4.8: Sub-Explosion for 'Manage Dictionary'

Basic words are the building blocks of Vocalhands. Each basic word has meaning, part of speech, animation, possibly an image and textual description of how the movement is made. 'Manage Basic Words' allow the consultant to view, add and edit basic words. Upon creation of a new word, it will be labeled as 'open for animation'. More of this will be discussed later.

Complex words are derived from basic words and therefore they share the same animation. 'Manage Complex Words' allow the consultants to create new and edit them. Each complex word is mapped to a basic word. This means deletion of a basic word will delete its related complex words.

'Manage grammar rules' allows the consultants to manually edit the grammatical rules the system will use for English-ASL translation. A grammar rule follows the labeled bracket notation format and the tags used by the Brown Corpus.

'Set test options' allows the consultants to specify options that will reflect on how the evaluation tests be given to the students.

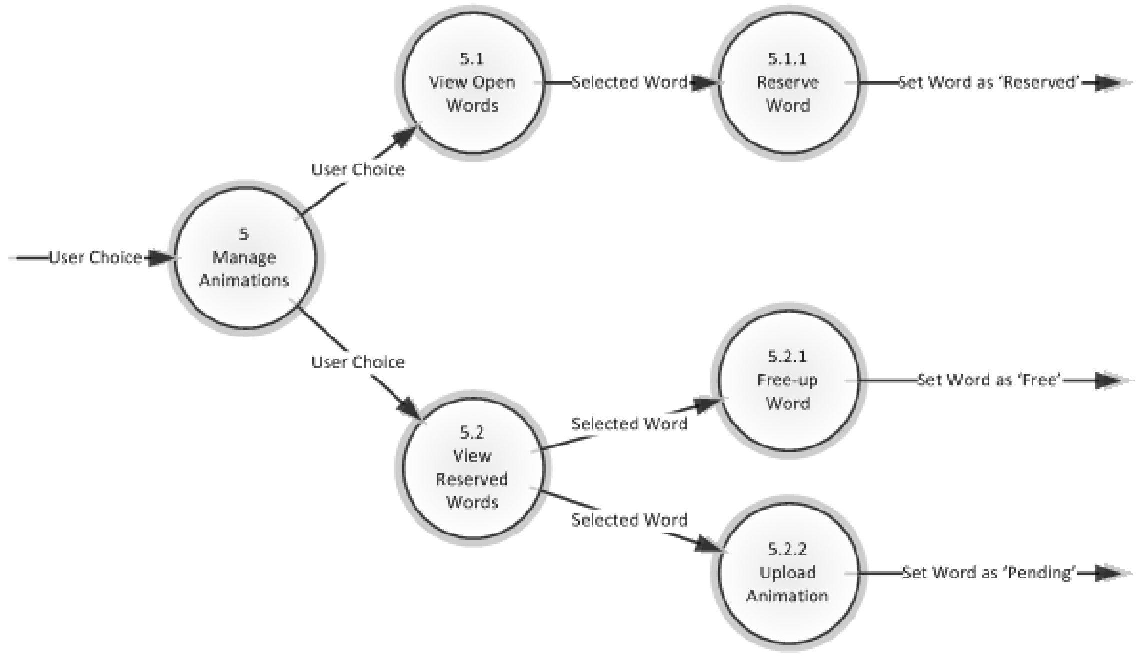


Figure 4.9: Sub-Explosion for 'Manage Animations'

‘Manage animations’ lets the flash developers know what animations they must make based on the request of the consultant. They build on a template to ensure consistency in every animation. After which, they upload the animation to the system’s data store.

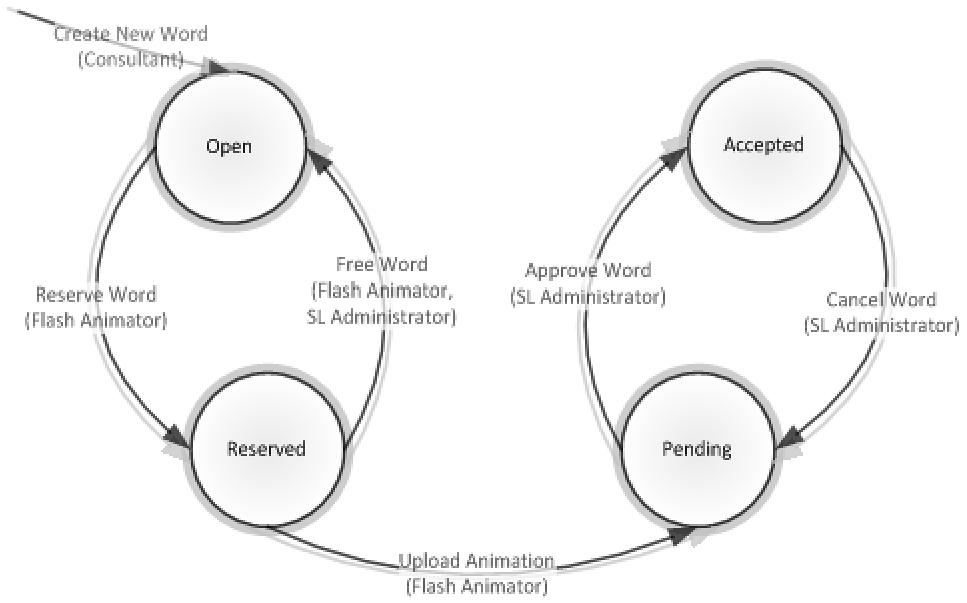


Figure 4.10: Lifecycle of a word

When a word is created, its status is set to open. During which, animators can reserve words they want to animate. At this stage, the either flash animator can free-up the word or upload the corresponding animation for the word. When an animation has been uploaded the word’s status turns to pending for approval by the Sign Language Administrator.

The Sign Language Administrator has the power to intervene during the cycle, as he can free-up a reserved word (in case the development has become standstill for the animator, so the word can be animated by others) or cancel a word he has already accepted. He also has the power to delete a word at any time.

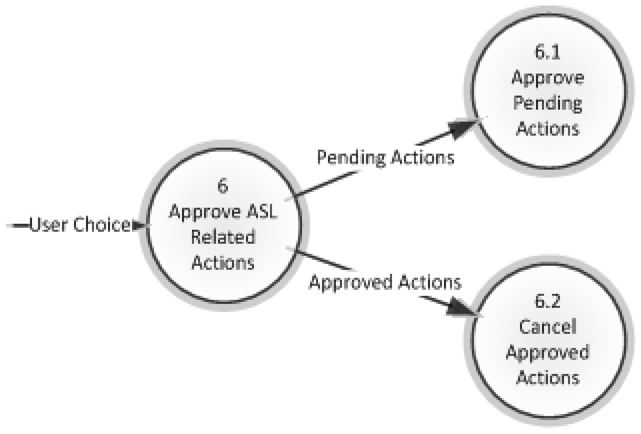


Figure 4.11: Sub-Explosion for ‘Approve ASL Related Actions’

Choosing ‘Approve ASL Related Actions’ will present the user a list of all existing ASL data, including articles, lessons, words and grammar rules which are categorized according to their type. From here, the administrator can approve pending actions, cancel approved ones or delete them. In addition, the administrator can also free up reserved simple words, as described above.

‘Manage System’ allows the Site Administrator to create new, edit and delete users.

The system also features a messaging system where each registered user can send and receive messages to/from one another. When sending a message, an email notification will also be sent to the recipient.

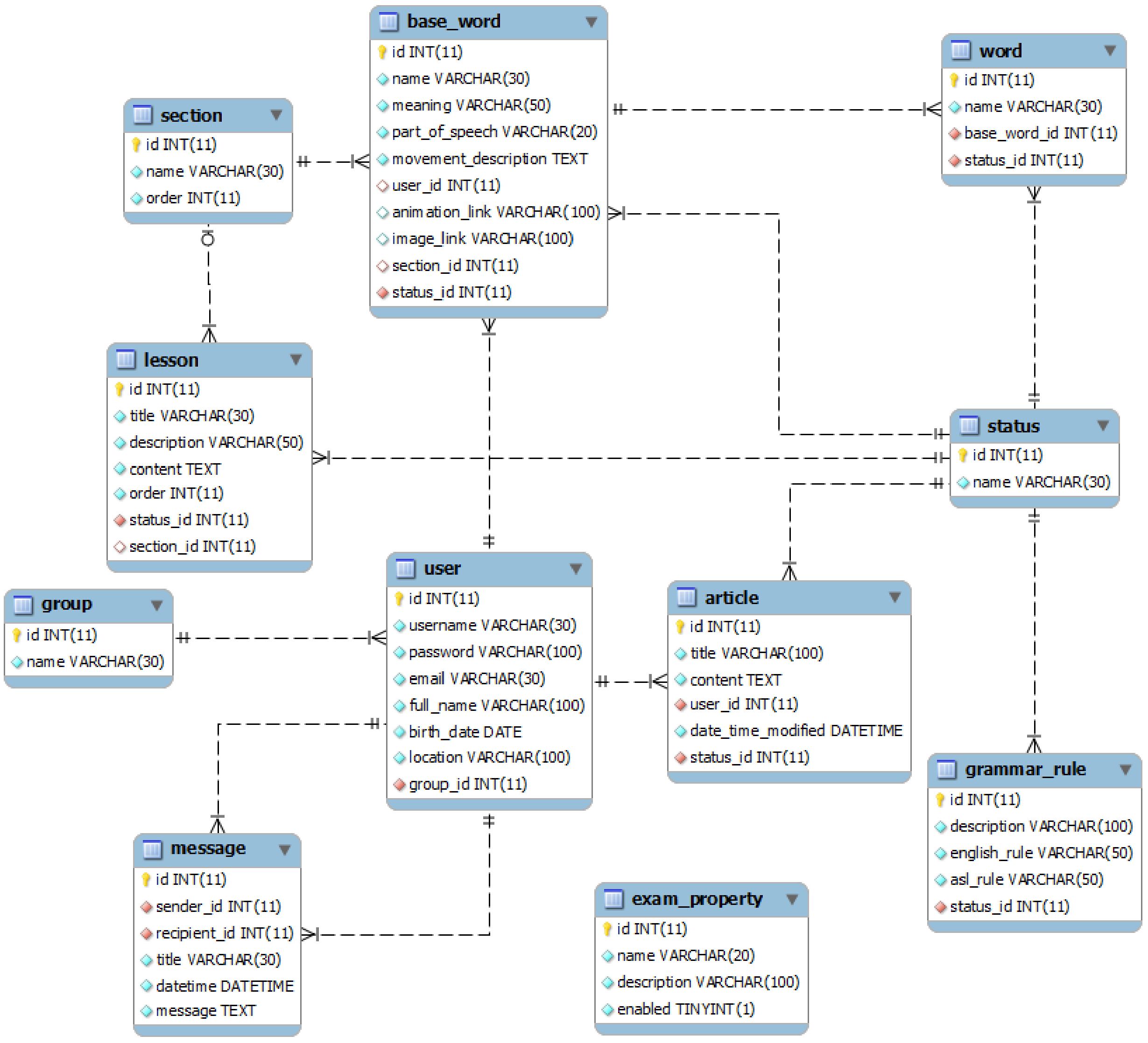


Figure 4.12: Entity-Relationship Diagram for Vocalhands

B. Data Dictionary

GROUP – contains the 4 user groups: Consultant, Flash Animator, Sign Language Administrator and Site Administrator.

Data Field	Data Type	Description
id	INT(11)	Group's primary key
name	VARCHAR(30)	Name of the group

USER – contains user data.

Data Field	Data Type	Description
id	INT(11)	User's primary key
username	VARCHAR(30)	User's login name
password	VARCHAR(100)	User's encrypted password
email	VARCHAR(30)	User's email address
full_name	VARCHAR(100)	User's full name
birth_date	DATE	User's date of birth
location	VARCHAR(100)	User's location
group_id	INT(11)	User's group type

ACTION_REQUEST - contains data for a message

Data Field	Data Type	Description
id	INT(11)	Message's primary key
sender_id	INT(11)	ID of sender
recipient_id	INT(11)	ID of recipient
title	VARCHAR(30)	Title of message
datetime	DATETIME	Date and time sent
message	TEXT	Message content

STATUS – current state of data/action. It has the following default rows: 1-PENDING, 2-ACCEPTED, 3-OPEN, 4-RESERVED.

Data Field	Data Type	Description
id	INT(11)	Status's primary key
name	INT(11)	Status name

ARTICLE – contains article data.

Data Field	Data Type	Description
id	INT(11)	Article's primary key
title	VARCHAR(100)	Title of article
content	TEXT	Content of article
user_id	INT(11)	Id of author
datetime_modified	DATETIME	Date and time modified
status_id	INT(11)	Article status

SECTION – level difficulty for lessons or words.

Data Field	Data Type	Description
id	INT(11)	Section's primary key
name	VARCHAR(30)	Name of section
order	INT(11)	Section's order in the list

LESSON – contains lesson data.

Data Field	Data Type	Description
id	INT(11)	Lesson’s primary key
title	VARCHAR(30)	Title of lesson
description	VARCHAR(50)	Short description of lesson
content	TEXT	Content of lesson
order	INT(11)	Lesson’s order in the list
status_id	INT(11)	Lesson status
section_id	INT(11)	Lesson difficulty

BASE_WORD – a basic word. It holds the animation.

Data Field	Data Type	Description
id	INT(11)	Base word’s primary key
name	VARCHAR(30)	Name of word
meaning	VARCHAR(50)	Meaning of the word
part_of_speech	VARCHAR(20)	Part of speech of the word
movement_description	TEXT	Description of how the sign is done
user_id	INT(11)	ID of author
animation_link	VARCHAR(100)	Link to animation
image_link	VARCHAR(100)	Link to image
section_id	INT(11)	Word difficulty
status_id	INT(11)	Word status

WORD – a complex word. It is mapped to a basic word.

Data Field	Data Type	Description
id	INT(11)	Complex word’s primary key
name	VARCHAR(30)	Name of complex word
base_word_id	INT(11)	Base word where complex word is mapped
status_id	INT(11)	Complex word status

GRAMMAR_RULE – contains rules used for English to ASL translation

Data Field	Data Type	Description
id	INT(11)	Grammar rule’s primary key
description	VARCHAR(100)	Description of the rule
english_rule	VARCHAR(50)	Grammar rule in English
asl_rule	VARCHAR(50)	Grammar rule in ASL
status_id	INT(11)	Grammar rule status

EXAM_PROPERTY – contains a property used by the self-assessment exam

Data Field	Data Type	Description
Id	INT(11)	Exam property’s primary key
name	VARCHAR(20)	Name of the property
description	VARCHAR(100)	Description of the property
enabled	TINYINT(1)	Tells whether the property is enabled or not

C. Algorithm

English to American Sign Language Translator

Translation for English to ASL follows the framework specified in Chapter 3. As the source text inputted by the user, in will undergo LFG analysis.

The system will accomplish this through the Stanford Parser. This package is a Java implementation of probabilistic natural language parsers, both highly optimized PCFG and lexicalized dependency parsers, and a lexicalized PCFG parser.

The parsed f-structure will be in XML.

After the text has been transformed to f-structure, it goes through lexical transfer. The transfer translates an English f-structure to ASL f-structure. The system does this in the following steps:

1. The program will create a DOM document to contain the English f-structure.
2. The program will collect all available grammar rules in the database.
3. Each grammar rule will then be parsed from string to array, with the less atomic rules first processed. The program will also resolve cases where there are duplicate tags within a rule.
4. For every grammar rule, the program will iterate through every node in the English f-structure, where it does the following steps:
    a. Determine whether the English sub-tree follows the grammar rule.
    b. Build a new tree that will contain the rearranged English sub-tree that follows the ASL grammar rules.
    c. Replace the old English sub-tree with the new ASL tree.

After the process, the translated ASL is returned by simply reading the F-structure. The program will clean out whitespaces the XML parsing has created. Each token of the string will be looked up in the lexical dictionary to retrieve animation data. Animation data will be compiled together to form a single translated animation. The translation will output the words used in the animation, as well as the words that were not used.

D. Technical Architecture

Vocalhands will be web-based; therefore it will not require anything from the client computer, aside from an internet browser.

Vocalhands will be implemented in PHP. In development, it will use XAMPP 1.7.1, where it will use the following software:

- Apache 2.2.11
- MySQL 5.1.33
- PHP 5.2.9
- phpMyAdmin 3.1.3.1
- XAMPP Control Version 2.5
- XAMPP CLI Bundle 1.3

Vocalhands will be implemented in CakePHP framework (version 1.25).

Stanford Parser requires at least Java Runtime Environment (JRE) 1.5 to run.

Displaying flash animations will require at least Adobe Flash Player 10 to run.

Developing flash SWF animations require software such as Adobe Flash CS4.

V. RESULTS

Vocalhands' home page is shown in figure 5.1. The homepage contains links to Vocalhands' guest features which are News & Articles, Tutorials, Dictionary, Word Search, Translate Sentence and Exams. It also has links to Site Information and Administrator login page.

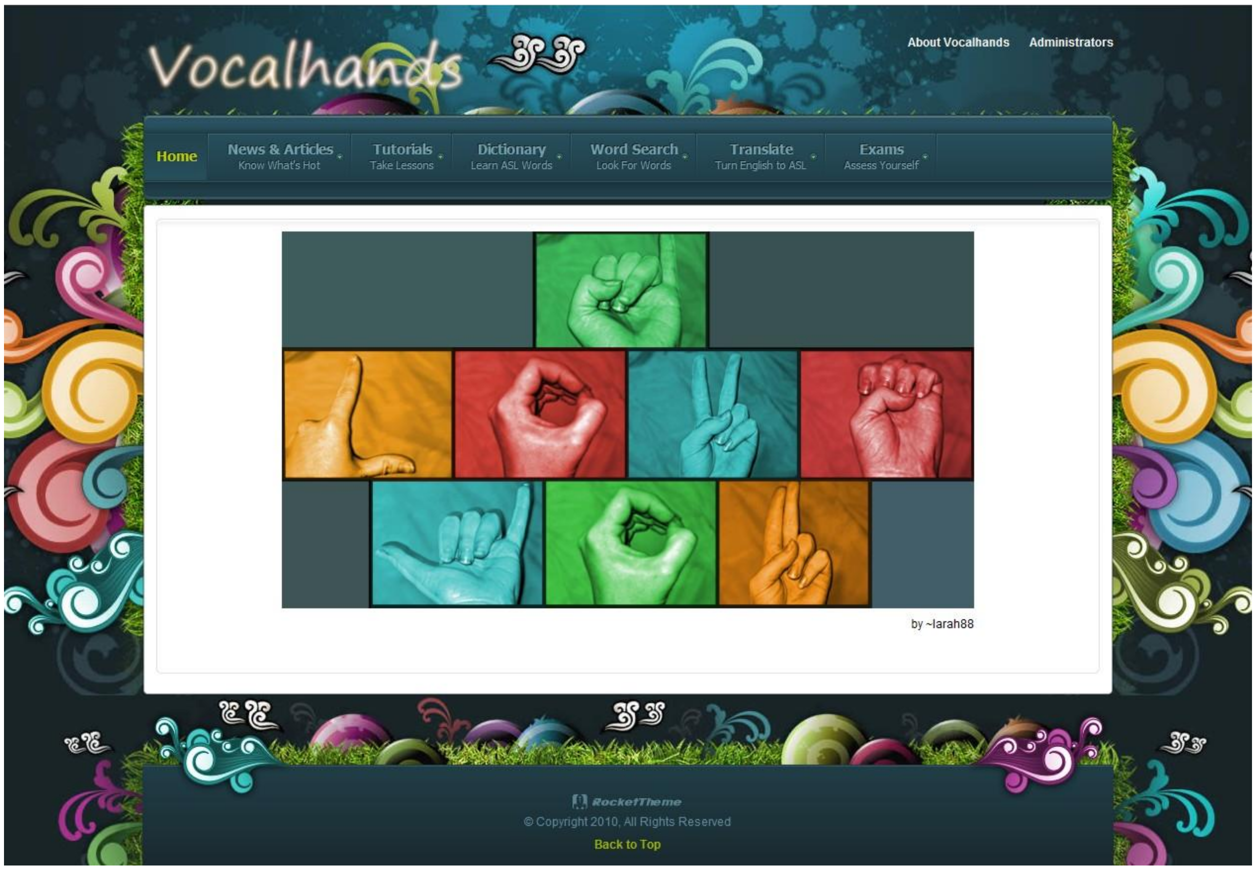
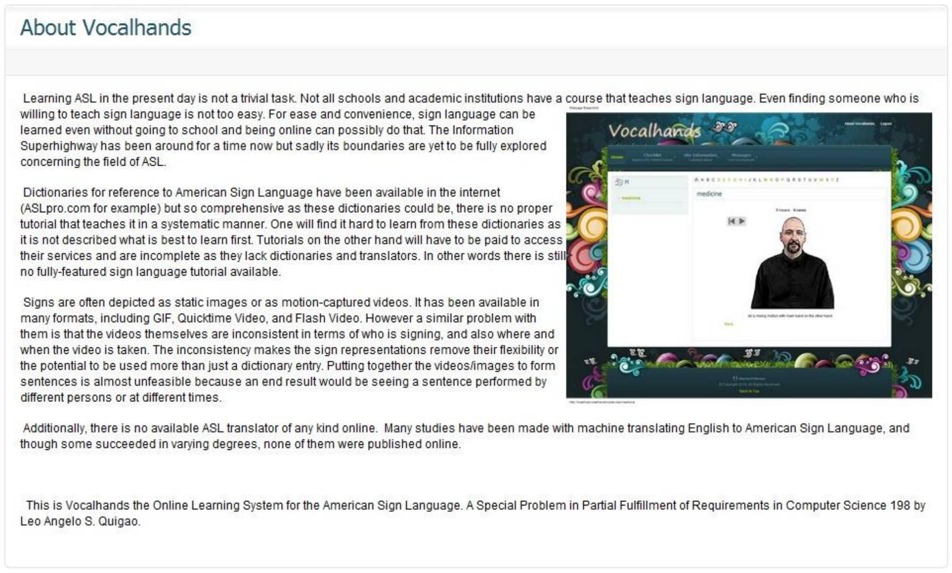


Figure 5.1: Vocalhands Home Page

Site Information is displayed like an article just like in figure 5.2.



About Vocalhands

Learning ASL in the present day is not a trivial task. Not all schools and academic institutions have a course that teaches sign language. Even finding someone who is willing to teach sign language is not too easy. For ease and convenience, sign language can be learned even without going to school and being online can possibly do that. The Information Superhighway has been around for a time now but sadly its boundaries are yet to be fully explored concerning the field of ASL.

Dictionaries for reference to American Sign Language have been available in the internet (ASLpro.com for example) but so comprehensive as these dictionaries could be, there is no proper tutorial that teaches it in a systematic manner. One will find it hard to learn from these dictionaries as it is not described what is best to learn first. Tutorials on the other hand will have to be paid to access their services and are incomplete as they lack dictionaries and translators. In other words there is still no fully-featured sign language tutorial available.

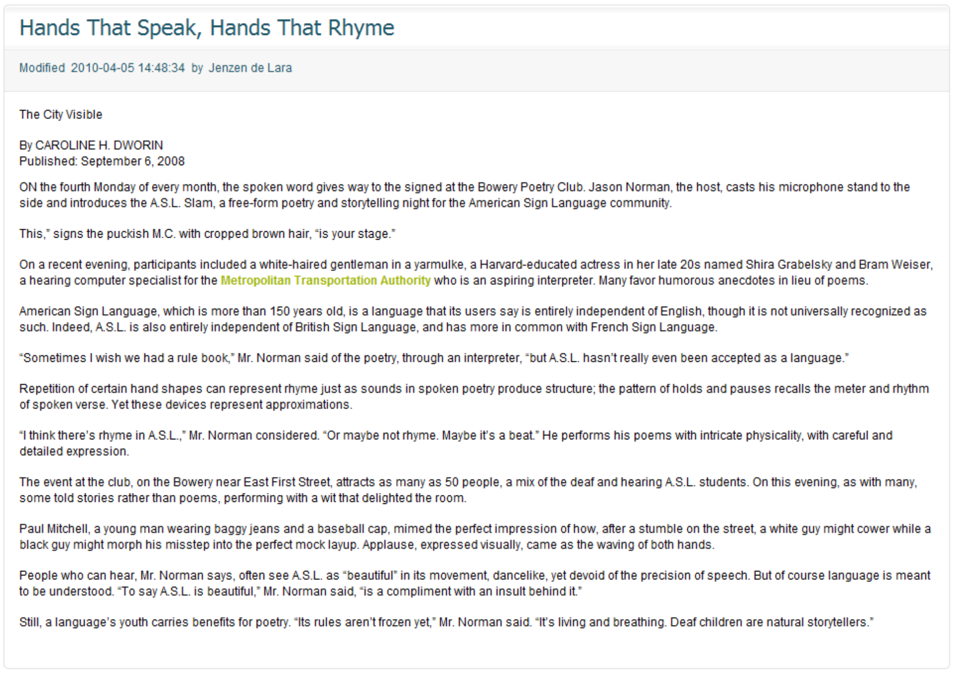
Signs are often depicted as static images or as motion-captured videos. It has been available in many formats, including GIF, Quicktime Video, and Flash Video. However a similar problem with them is that the videos themselves are inconsistent in terms of who is signing, and also where and when the video is taken. The inconsistency makes the sign representations remove their flexibility or the potential to be used more than just a dictionary entry. Putting together the videos/images to form sentences is almost unfeasible because an end result would be seeing a sentence performed by different persons or at different times.

Additionally, there is no available ASL translator of any kind online. Many studies have been made with machine translating English to American Sign Language, and though some succeeded in varying degrees, none of them were published online.

This is Vocalhands the Online Learning System for the American Sign Language. A Special Problem in Partial Fulfillment of Requirements in Computer Science 198 by Leo Angelo S. Quigao.

Figure 5.2: Vocalhands Site Information

By selecting News & Articles tab, the user will have access to available news and articles displayed in blog format. Each article is displayed with its content along with author and modified date, as shown in figure 5.3.



Hands That Speak, Hands That Rhyme

Modified 2010-04-05 14:48:34 by Jenzen de Lara

The City Visible

By CAROLINE H. DWORIN
Published: September 6, 2008

ON the fourth Monday of every month, the spoken word gives way to the signed at the Bowery Poetry Club. Jason Norman, the host, casts his microphone stand to the side and introduces the A.S.L. Slam, a free-form poetry and storytelling night for the American Sign Language community.

This," signs the puckish M.C. with cropped brown hair, "is your stage."

On a recent evening, participants included a white-haired gentleman in a yarmulke, a Harvard-educated actress in her late 20s named Shira Grabelsky and Bram Weiser, a hearing computer specialist for the Metropolitan Transportation Authority who is an aspiring interpreter. Many favor humorous anecdotes in lieu of poems.

American Sign Language, which is more than 150 years old, is a language that its users say is entirely independent of English, though it is not universally recognized as such. Indeed, A.S.L. is also entirely independent of British Sign Language, and has more in common with French Sign Language.

"Sometimes I wish we had a rule book," Mr. Norman said of the poetry, through an interpreter, "but A.S.L. hasn't really even been accepted as a language."

Repetition of certain hand shapes can represent rhyme just as sounds in spoken poetry produce structure; the pattern of holds and pauses recalls the meter and rhythm of spoken verse. Yet these devices represent approximations.

"I think there's rhyme in A.S.L.," Mr. Norman considered. "Or maybe not rhyme. Maybe it's a beat." He performs his poems with intricate physicality, with careful and detailed expression.

The event at the club, on the Bowery near East First Street, attracts as many as 50 people, a mix of the deaf and hearing A.S.L. students. On this evening, as with many, some told stories rather than poems, performing with a wit that delighted the room.

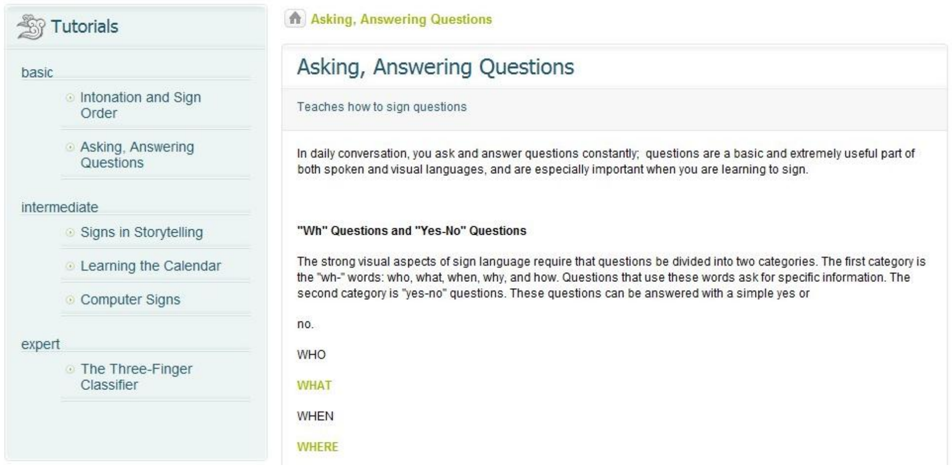
Paul Mitchell, a young man wearing baggy jeans and a baseball cap, mimed the perfect impression of how, after a stumble on the street, a white guy might cower while a black guy might morph his misstep into the perfect mock layup. Applause, expressed visually, came as the waving of both hands.

People who can hear, Mr. Norman says, often see A.S.L. as "beautiful" in its movement, dancelike, yet devoid of the precision of speech. But of course language is meant to be understood. "To say A.S.L. is beautiful," Mr. Norman said, "is a compliment with an insult behind it."

Still, a language's youth carries benefits for poetry. "Its rules aren't frozen yet," Mr. Norman said. "It's living and breathing. Deaf children are natural storytellers."

Figure 5.3: News and Articles Index Page

An example of a tutorials page is shown in Figure 5.4, where a list of sections and its lessons is displayed in the left bar and the page's main content is the selected lesson.



Tutorials

basic
- Intonation and Sign Order
- Asking, Answering Questions

intermediate
- Signs in Storytelling
- Learning the Calendar
- Computer Signs

expert
- The Three-Finger Classifier

Asking, Answering Questions

Asking, Answering Questions

Teaches how to sign questions

In daily conversation, you ask and answer questions constantly; questions are a basic and extremely useful part of both spoken and visual languages, and are especially important when you are learning to sign.

"Wh" Questions and "Yes-No" Questions

The strong visual aspects of sign language require that questions be divided into two categories. The first category is the "wh-" words: who, what, when, why, and how. Questions that use these words ask for specific information. The second category is "yes-no" questions. These questions can be answered with a simple yes or

no.

WHO

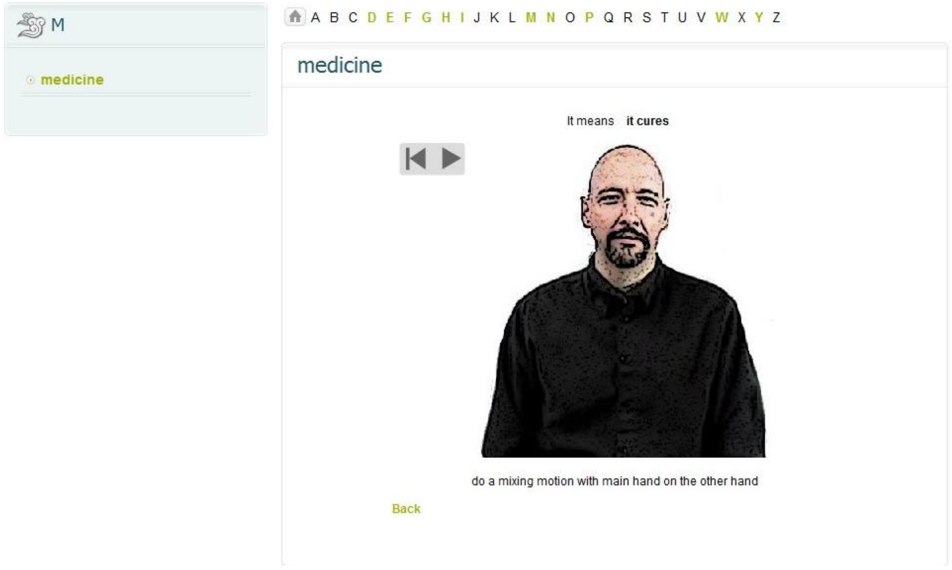
WHAT

WHEN

WHERE

Figure 5.4: View Lesson Page

The dictionary, as shown in figure 5.5, displays the word list categorized by the initial letter of each word. On selecting a word, its information is displayed together with the meaning, movement description and signed animation, and features playback options such as play/pause and reset.



medicine

It means **it cures**

do a mixing motion with main hand on the other hand

Figure 5.5: View Word Page

The Word Search as displayed in figure 5.6 allows the user to search for a word in the dictionary. The search results returns a list of words, if there is any available, as displayed in figure 5.7.

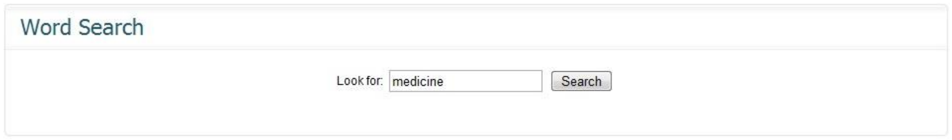
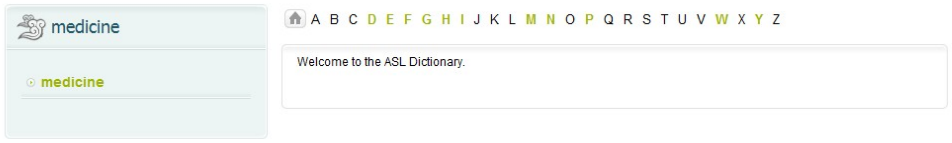


Figure 5.6: Word Search



Welcome to the ASL Dictionary.

Figure 5.7: Word Search Result

The Translate English page allows the user to input a string of sentences he want translated, as displayed in figure 5.8. The output is the translated ASL animation, together with the list of words that were signed and those that were not.

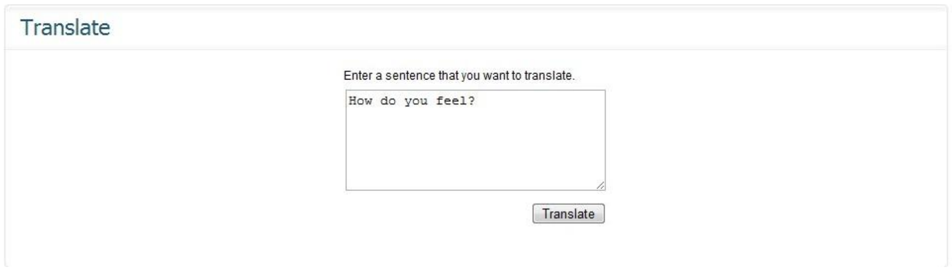
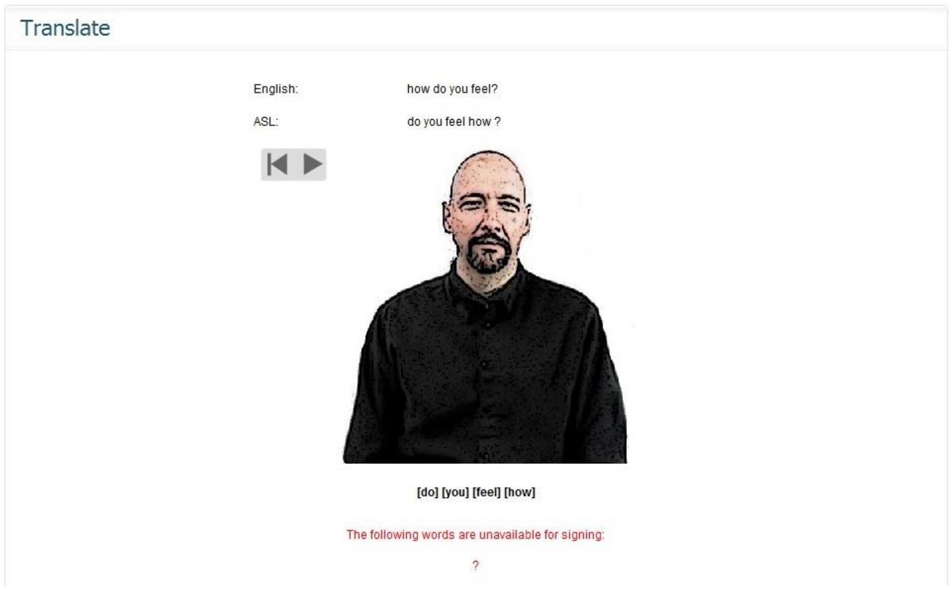


Figure 5.8: Translate English Page


Translate

English: how do you feel?

ASL: do you feel how ?

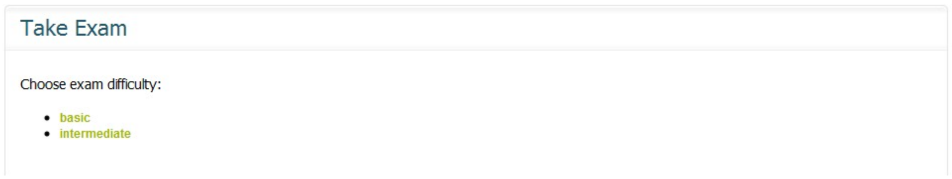
[do] [you] [feel] [how]

The following words are unavailable for signing:

?

Figure 5.9: Translation Output

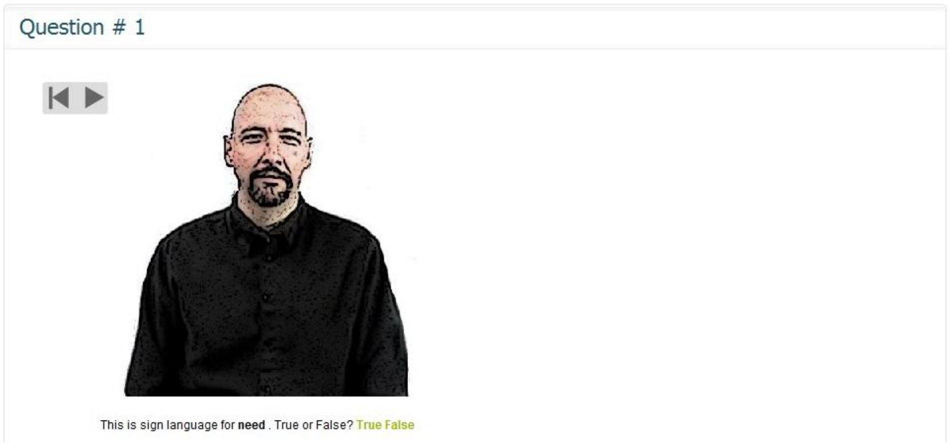
The Take Exam page gives the user a self-assessment exam based on a given difficulty, as displayed in figure 5.10. The user is given questions that are in true or false and multiple choices, as shown in figure 5.11. After an administrator-specified number of questions, the user is shown the result of his exam, as displayed in figure 5.12.


Take Exam

Choose exam difficulty:

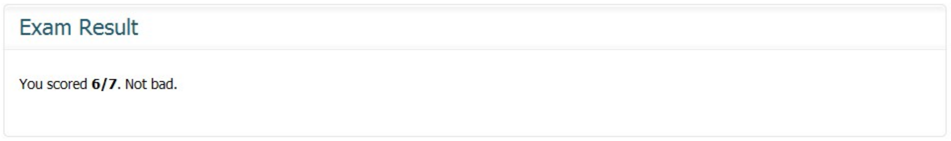
- basic
- intermediate

Figure 5.10: Take Exam Index Page



This is sign language for **need** . True or False? True False

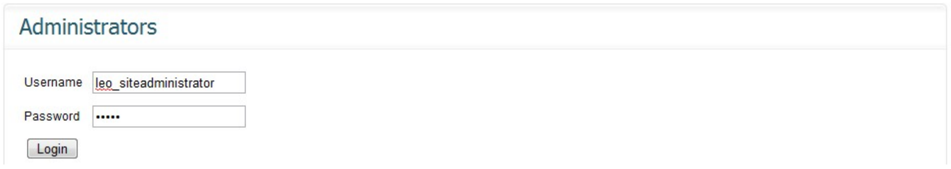
Figure 5.11: Exam Question Page


Exam Result

You scored **6/7**. Not bad.

Figure 5.12: Exam Result Page

The Administrator login page, as displayed in figure 5.13, allows the user to login to an administrator page, designated by the administrator type. The user is given username and password fields to fill in.

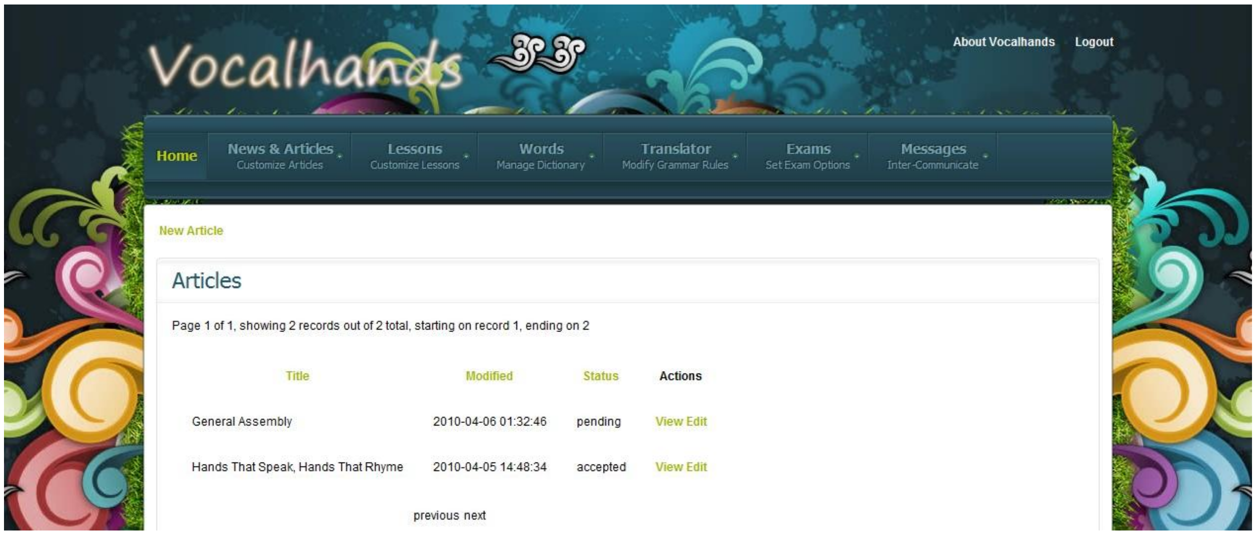

Administrators

Username leo_siteadministrator

Password •••••

Figure 5.13: Administrator Login Page

As the consultant logs in, he is displayed a list of features which are Customize Articles, Customize Lessons, Manage Dictionary, Modify Grammar Rules, Set Exam Options, and Messages. Upon selecting Customize Articles, a list of articles is displayed with which the consultant can either view or edit, just like in figure 5.14.



Articles

Page 1 of 1, showing 2 records out of 2 total, starting on record 1, ending on 2

Title	Modified	Status	Actions
General Assembly	2010-04-06 01:32:46	pending	View Edit
Hands That Speak, Hands That Rhyme	2010-04-05 14:48:34	accepted	View Edit

previous next

Figure 5.14: List Articles Page

As the consultant chooses to create a new article, he is given a form where he can input the title and content (which can be entered in WYSIWYG format), as shown in figure 5.15.

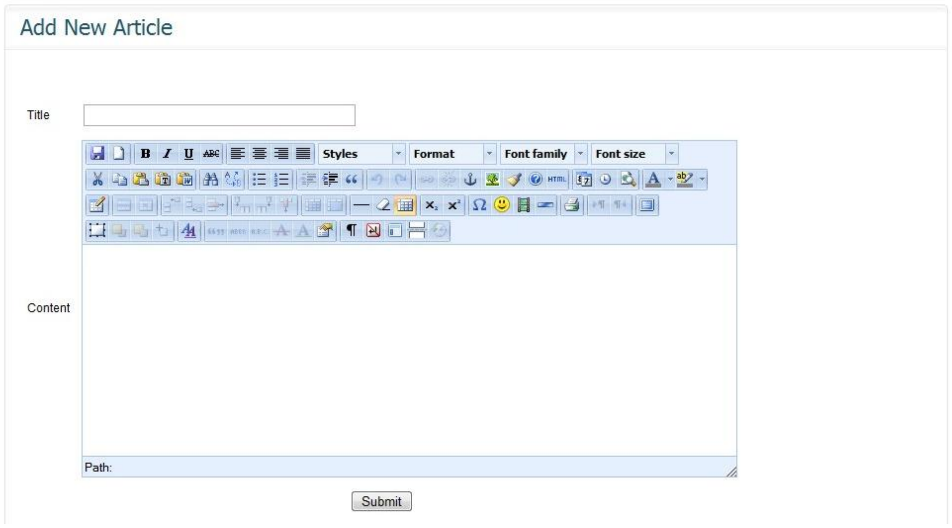
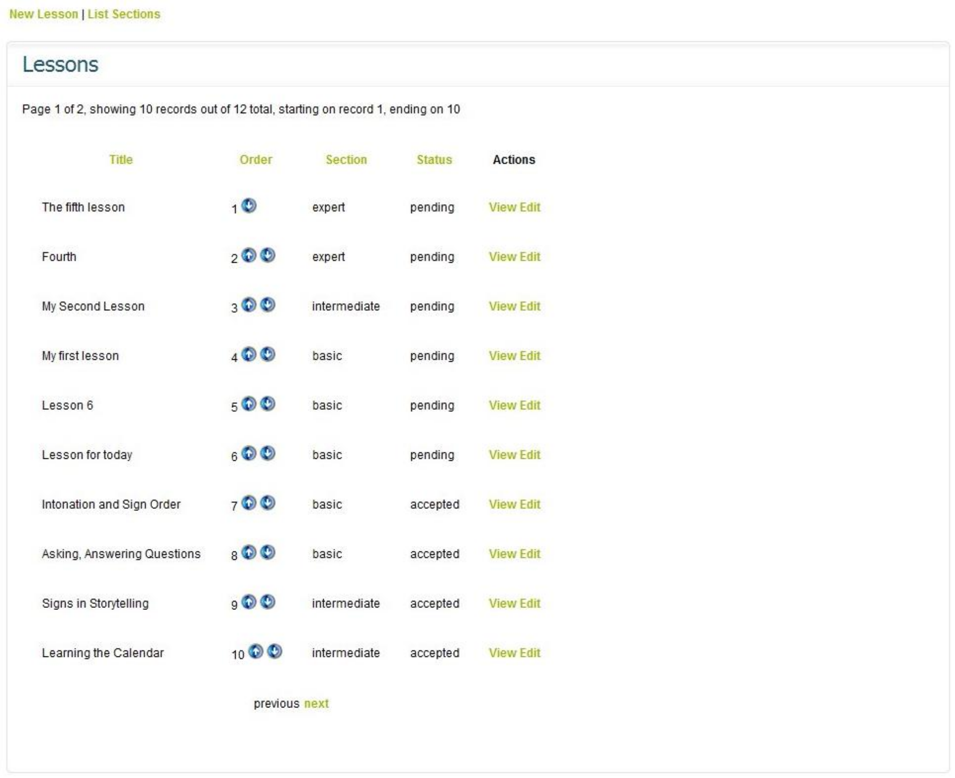

Add New Article

Figure 5.15: New Article Page

The Customize Lessons feature is shown in figure 5.16. It allows the user to view, edit, and rearrange lessons. Links to creating new lessons and the sections list are also be displayed within the page.



New Lesson | List Sections

Lessons

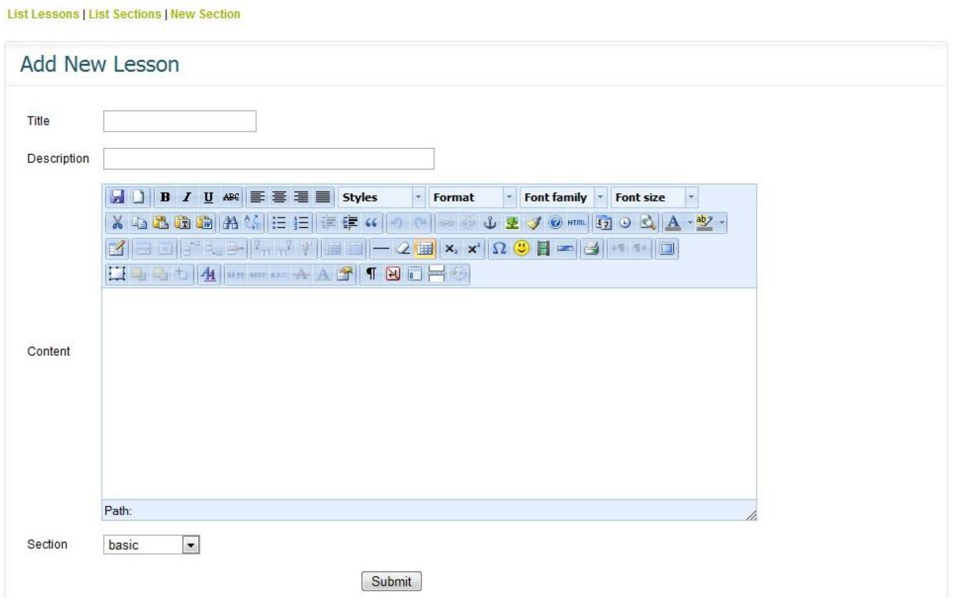
Page 1 of 2, showing 10 records out of 12 total, starting on record 1, ending on 10

Title	Order	Section	Status	Actions
The fifth lesson	1	expert	pending	View Edit
Fourth	2	expert	pending	View Edit
My Second Lesson	3	intermediate	pending	View Edit
My first lesson	4	basic	pending	View Edit
Lesson 6	5	basic	pending	View Edit
Lesson for today	6	basic	pending	View Edit
Intonation and Sign Order	7	basic	accepted	View Edit
Asking, Answering Questions	8	basic	accepted	View Edit
Signs in Storytelling	9	intermediate	accepted	View Edit
Learning the Calendar	10	intermediate	accepted	View Edit

previous next

Figure 5.16: Lesson List Page

When the consultant chooses to create a new lesson, he is required to input lesson information such as title, description, content and section as displayed in figure 5.17. The user is allowed to create links to words found in the dictionary. The user can also choose to create a new section here. Upon creating a lesson, it is assigned to the last order in the lesson list.

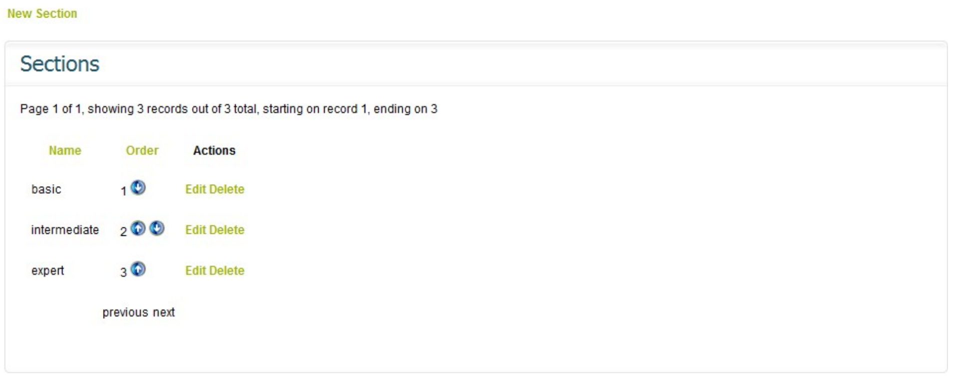

List Lessons | List Sections | New Section

Add New Lesson

Title	
Description	
Content	
Section	basic

Figure 5.17: New Lesson Page

The Section List page displays the list of available sections the user can choose from when creating lessons and words. They can be reordered and edited, as shown in figure 5.18. In creating a new section, the user is required to input the section of the name. Upon creating a new section, it is ordered last in the list, just like a lesson.


New Section

Sections

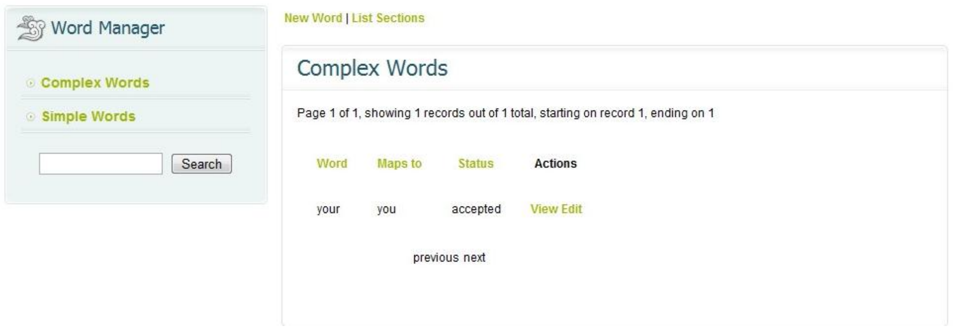
Page 1 of 1, showing 3 records out of 3 total, starting on record 1, ending on 3

Name	Order	Actions
basic	1	Edit Delete
intermediate	2	Edit Delete
expert	3	Edit Delete

previous next

Figure 5.18: List Sections Page

As the consultant selects Manage Dictionary, he can choose to manage complex and simple words. The page allows the user to search for recorded words to avoid duplication of data. The user can create new, view and edit complex words, as shown in figure 5.19.



Complex Words

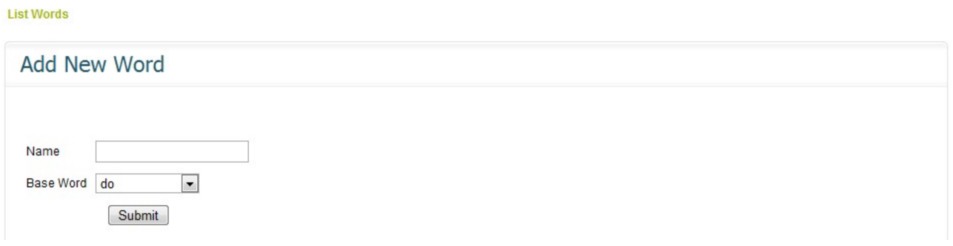
Page 1 of 1, showing 1 records out of 1 total, starting on record 1, ending on 1

Word	Maps to	Status	Actions
your	you	accepted	View Edit

previous next

Figure 5.19: Complex Words List

When the user chooses to create a new complex word, he is required to input its name as well as select the basic word it must be mapped to, as shown in figure 5.20.

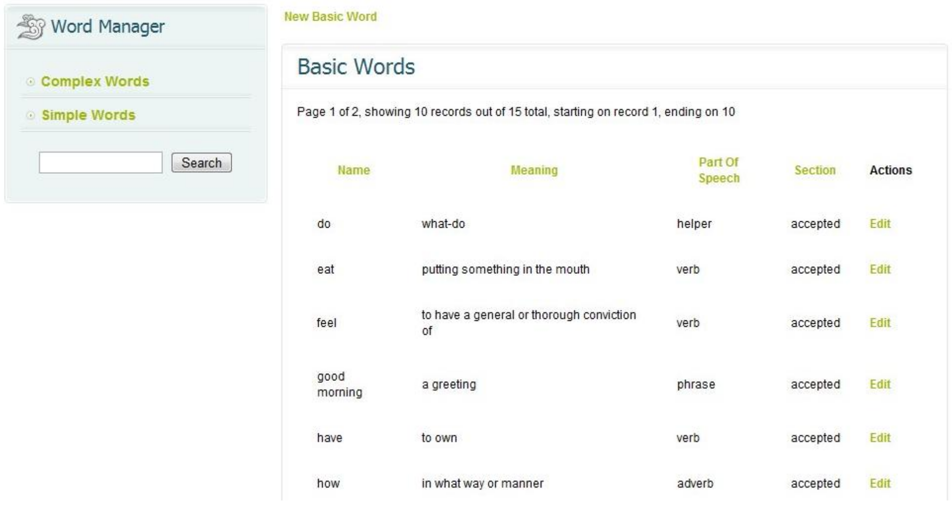


Add New Word

Name	
Base Word	do

Figure 5.20: Add New Complex Word

Figure 5.21 shows the list of basic words. Here, the user can also search the database for existing words to avoid duplication of data. The user can create new, view and edit basic words.


Word Manager

Complex Words

Simple Words

Search

New Basic Word

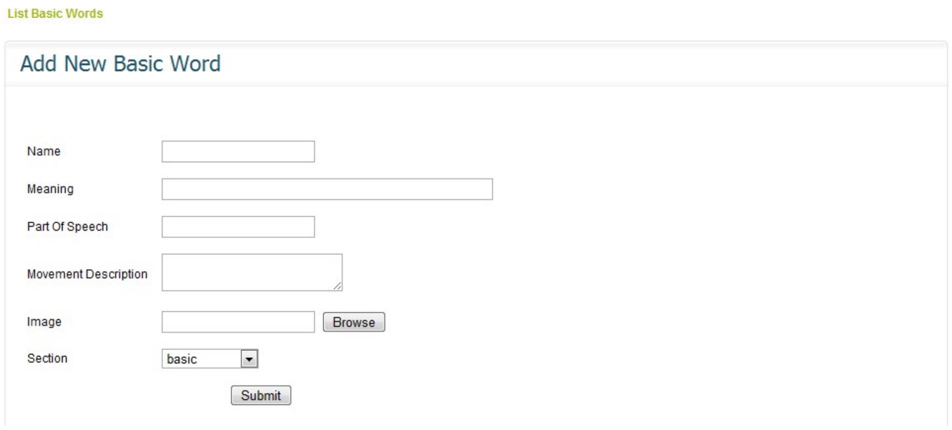
Basic Words

Page 1 of 2, showing 10 records out of 15 total, starting on record 1, ending on 10

Name	Meaning	Part Of Speech	Section	Actions
do	what-do	helper	accepted	Edit
eat	putting something in the mouth	verb	accepted	Edit
feel	to have a general or thorough conviction of	verb	accepted	Edit
good morning	a greeting	phrase	accepted	Edit
have	to own	verb	accepted	Edit
how	in what way or manner	adverb	accepted	Edit

Figure 5.21: Basic Words List

When adding a new basic word, the user is required to input the all fields, with the exception of the image field as shown in figure 5.22.


List Basic Words

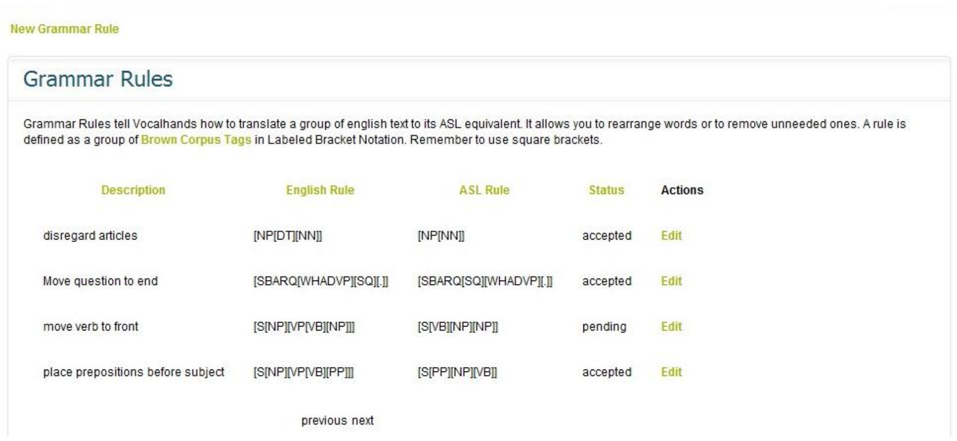
Add New Basic Word

Name	
Meaning	
Part Of Speech	
Movement Description	
Image	Browse
Section	basic

Submit

Figure 5.22: Add New Basic Word

The grammar rules index page allows the user to see a list of all recorded grammar rules in Vocalhands. Here, the user is able to see all the attributes of each grammar rule. He can add new grammar rules, as well as edit the existing ones. An available manual on part-of-speech tagging is linked to the page.


New Grammar Rule

Grammar Rules

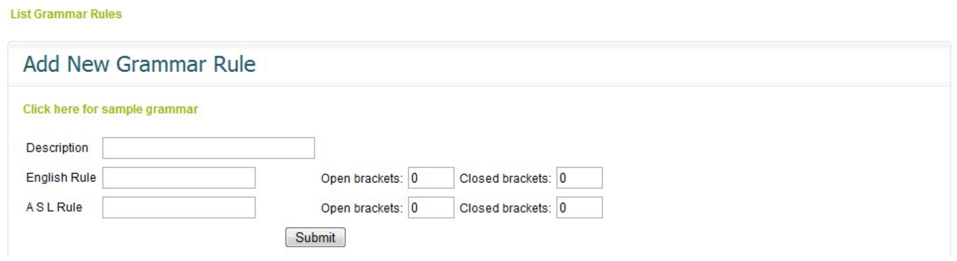
Grammar Rules tell Vocalhands how to translate a group of english text to its ASL equivalent. It allows you to rearrange words or to remove unneeded ones. A rule is defined as a group of Brown Corpus Tags in Labeled Bracket Notation. Remember to use square brackets.

Description	English Rule	ASL Rule	Status	Actions
disregard articles	[NP[DT][NN]]	[NP[NN]]	accepted	Edit
Move question to end	[SBARQ[WHADVP][SQ][.]]	[SBARQ[SQ][WHADVP][.]]	accepted	Edit
move verb to front	[S[NP][VP[VB][NP]]]	[S[VB][NP][NP]]	pending	Edit
place prepositions before subject	[S[NP][VP[VB][PP]]]	[S[PP][NP][VB]]	accepted	Edit

previous next

Figure 5.23: List Grammar Rules

In adding new grammar rules, the user must input a description, the English rule and the corresponding ASL rule, as shown in figure 5.24. The user can also see the number of brackets already placed in the text boxes. To know the grammar structure of a sentence, the user can click on an external link.


List Grammar Rules

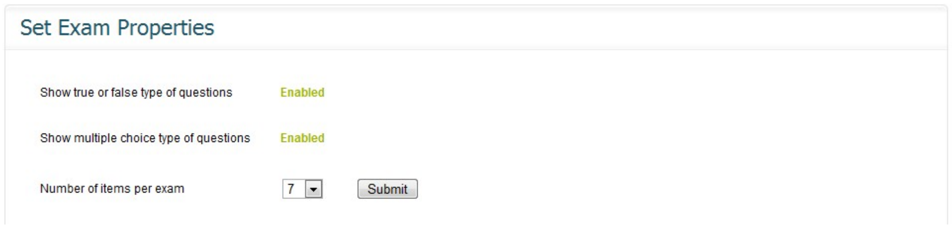
Add New Grammar Rule

Click here for sample grammar

Description					
English Rule		Open brackets:	0	Closed brackets:	0
A S L Rule		Open brackets:	0	Closed brackets:	0

Figure 5.24: Add New Grammar Rule

In setting exam properties, the user can choose to use the two types of questions or not. He can also set the number of questions in a particular exam. (Figure 5.25)

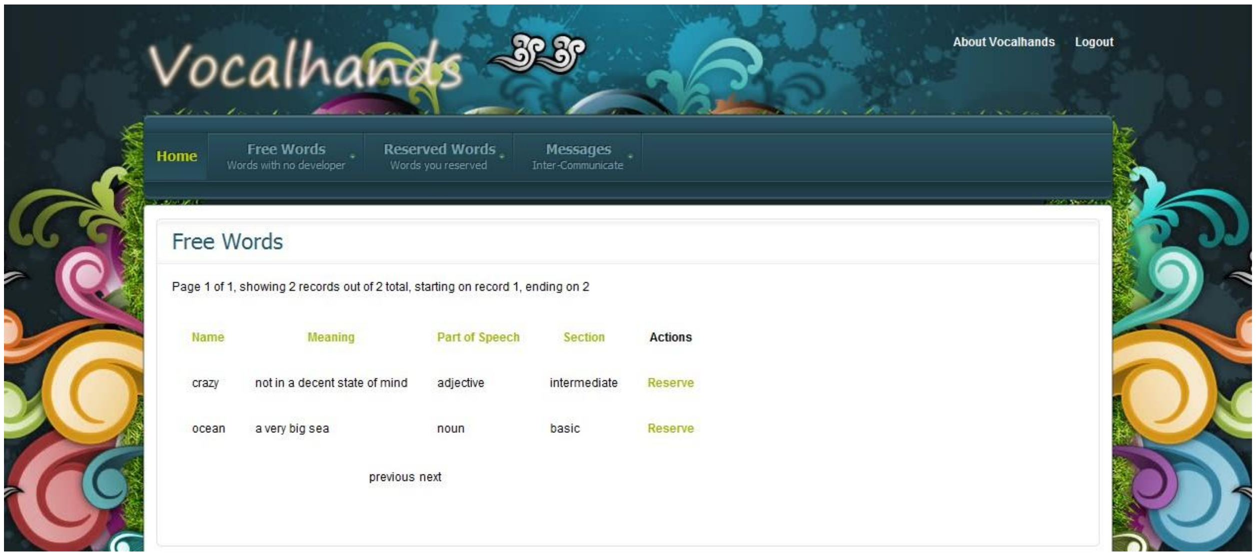


Set Exam Properties

Show true or false type of questions	Enabled
Show multiple choice type of questions	Enabled
Number of items per exam	7

Figure 5.25: Set Exam Properties Page

As a Flash Animator, he can see a list of free words that he can reserve to animate, as shown in figure 5.26. Reserved words then are either freed up or have an animation uploaded. (Figure 5.27)

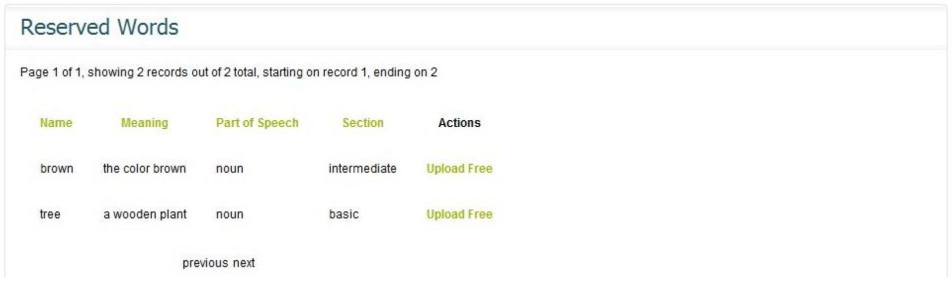


Free Words

Page 1 of 1, showing 2 records out of 2 total, starting on record 1, ending on 2

Name	Meaning	Part of Speech	Section	Actions
crazy	not in a decent state of mind	adjective	intermediate	Reserve
ocean	a very big sea	noun	basic	Reserve

Figure 5.26: List Free Words



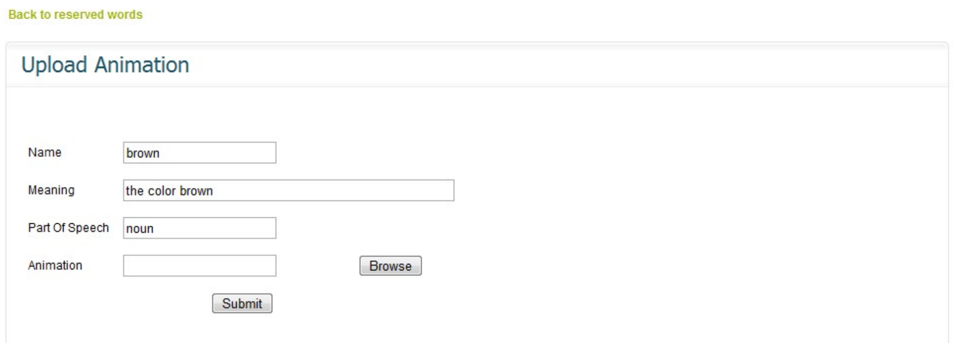
Reserved Words

Page 1 of 1, showing 2 records out of 2 total, starting on record 1, ending on 2

Name	Meaning	Part of Speech	Section	Actions
brown	the color brown	noun	intermediate	Upload Free
tree	a wooden plant	noun	basic	Upload Free

Figure 5.27: List Reserved Words

In uploading an animation for a basic word, the flash animator can only change the animation field, wherein he can browse his local files for the SWF animation. (Figure 5.28)

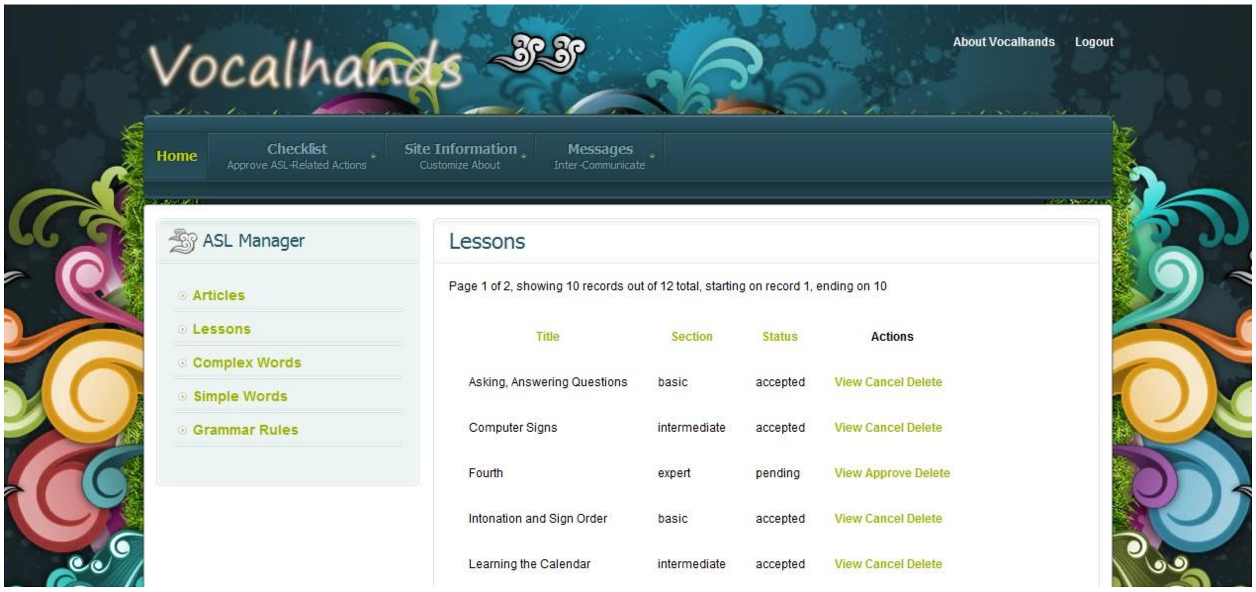


Name	brown
Meaning	the color brown
Part Of Speech	noun
Animation	

Figure 5.28: Upload Animation Page

As the Sign Language Administrator logs in to Vocalhands, the ASL Manager Checklist is displayed. Here, he can approve and cancel various Vocalhands entities such as articles, lessons, words and grammar rules. (Figure 5.29)

He can also edit site information, just like how an article is being edited.

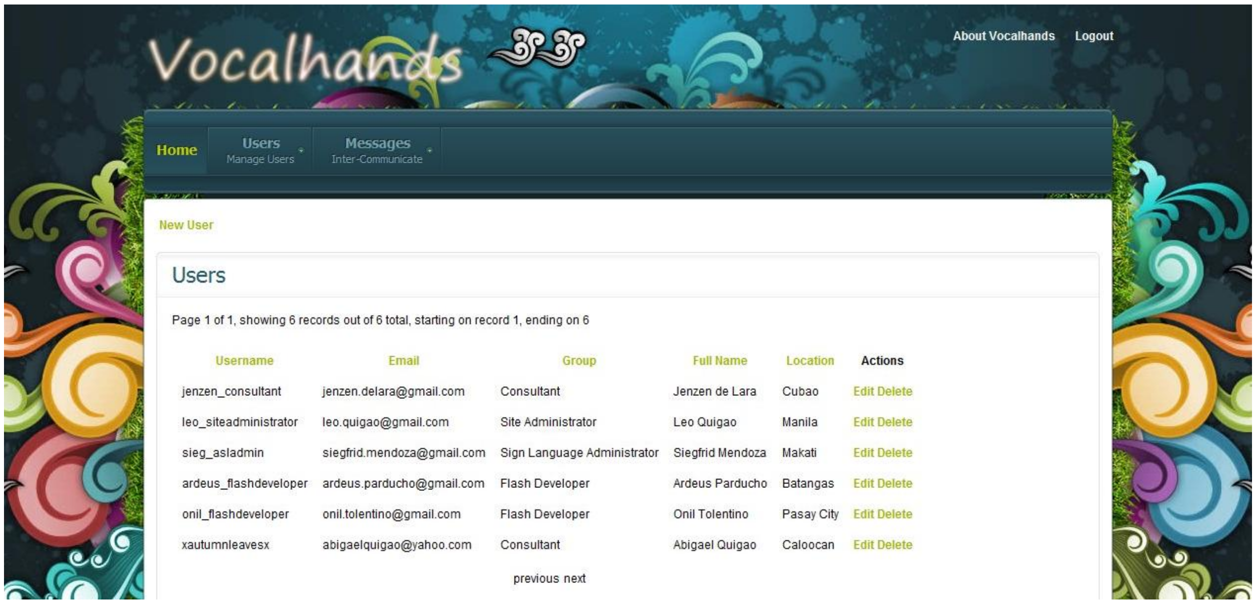


Page 1 of 2, showing 10 records out of 12 total, starting on record 1, ending on 10

Title	Section	Status	Actions
Asking, Answering Questions	basic	accepted	View Cancel Delete
Computer Signs	intermediate	accepted	View Cancel Delete
Fourth	expert	pending	View Approve Delete
Intonation and Sign Order	basic	accepted	View Cancel Delete
Learning the Calendar	intermediate	accepted	View Cancel Delete

Figure 5.29: Approve Lessons Page

The Site Administrator has the power to manage all users in Vocalhands. He can create new users, as well as edit or delete existing ones. (Figure 5.30)



Users

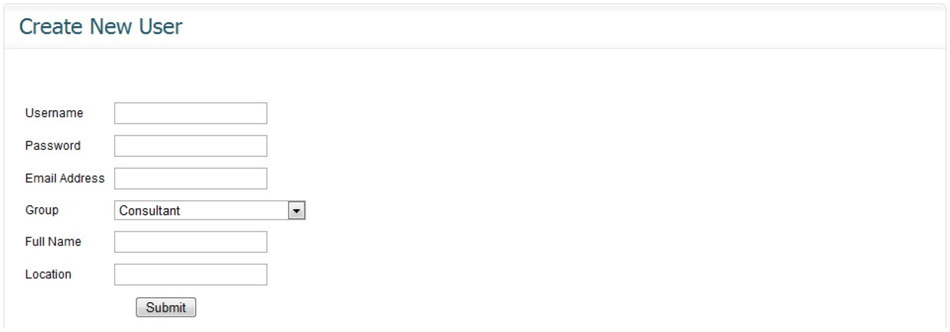
Page 1 of 1, showing 6 records out of 6 total, starting on record 1, ending on 6

Username	Email	Group	Full Name	Location	Actions
jenzen_consultant	jenzen.delara@gmail.com	Consultant	Jenzen de Lara	Cubao	Edit Delete
leo_siteadministrator	leo.quigao@gmail.com	Site Administrator	Leo Quigao	Manila	Edit Delete
sieg_asladmin	siegfrid.mendoza@gmail.com	Sign Language Administrator	Siegfrid Mendoza	Makati	Edit Delete
ardeus_flashdeveloper	ardeus.parducho@gmail.com	Flash Developer	Ardeus Parducho	Batangas	Edit Delete
onil_flashdeveloper	onil.tolentino@gmail.com	Flash Developer	Onil Tolentino	Pasay City	Edit Delete
xautumnleavesx	abigaelquigao@yahoo.com	Consultant	Abigael Quigao	Caloocan	Edit Delete

previous next

Figure 5.30: User List Index Page

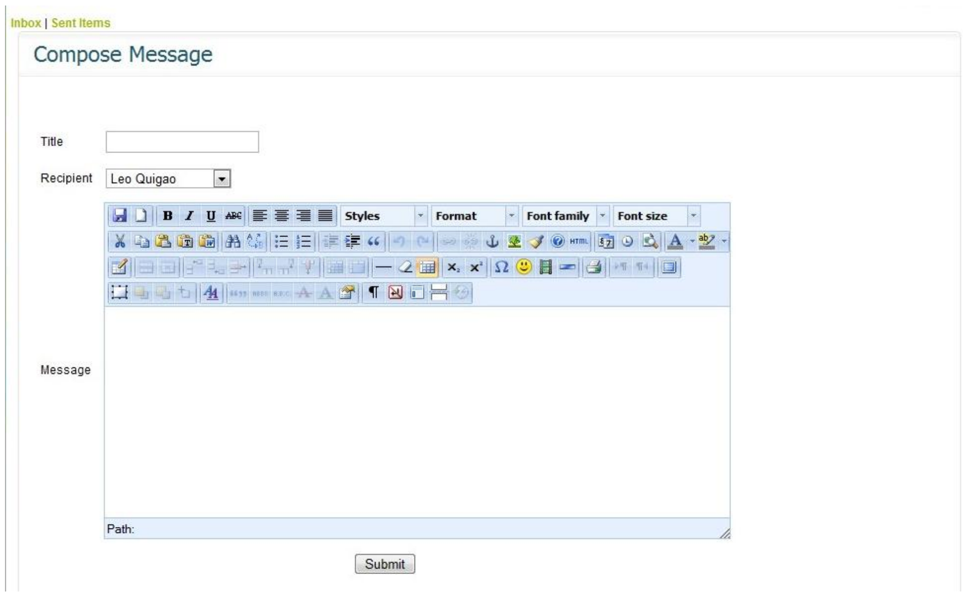
When the Site Administrator chooses to create a new user, he must enter all fields in the form, which are the username, password, email address, group, full name and location (Figure 5.31).


Create New User

Username	
Password	
Email Address	
Group	Consultant
Full Name	
Location	

Figure 5.31: Create New User Page

All administrators can use the messaging system. Figure 5.32 shows a compose message form. He can also view his inbox or view his sent items.


Inbox | Sent Items

Compose Message

Title	
Recipient	Leo Quigao
Message	

Figure 5.32: Compose Message Page

VI. DISCUSSION

Vocalhands is a web-based system that is centered on the ASL translation engine and its sign language tutorials. Because of this, Vocalhands is considered a content-oriented system. For the end-users to fully benefit from its content and its administrators to gain flexible and easy management, Vocalhands has been developed into a CMS (Content Management System). Though not a true fully featured CMS compared to the likes of Joomla or Drupal, Vocalhands allows its administrators to create, edit, and manage content like lessons with ease of use through link functionality access (see figure 5.16) and the use of WYSIWYG forms to create customized web pages without the need of software development intervention.

Vocalhands' role-based backend architecture allows for separation of privileges and responsibilities among the different types of users depending on their knowledge and skill. Flash animators, for example can upload sign language animations because they are skilled in making flash animations but they don't necessarily need expertise in sign language communication as consultants do. Also above all, there is the sign language administrator who ensures the quality of the site's content by verifying and approving submitted content by the consultants and flash animators.

The Lexical Functional Transfer mechanism used by the ASL translator allows for syntactic accuracy of translating English language to ASL. It should also be noted that sentences are parsed per word, so signs with corresponding English that may be said in a bunch of words are translated to one sign per word. For example, 'I love you' can be signed with a single gesture but will be translated by the system per single word, unless the entire phrase is found as a base word on the database. Though this kind of method can produce understandable ASL to its speakers, it is far from the one usually used. The strength of ASL lies on delivering the meaning of sentences so it will be a drastic improvement to change the translator's engine to use semantic transfer.

VII. CONCLUSION

The ubiquity of the World Wide Web has given new grounds for learning general. Vocalhands has used this ground to provide high quality tutorials for people to learn the American signed language. The ease of access the internet provides allows its users to learn even at the comfort of their own homes without having to apply and learn at schools that teaches the subject. It allows its user to study ASL on the right pace he/she wants to learn, as the lessons are self-studied. It provides content needed for users to become well-versed in signing through lessons, the reference dictionary and the translator.

Vocalhands as a CMS allows the different kinds of administrators to easily fill and manage the content displayed on the web site. Using WYSIWYG, the sign language consultants can write lessons and articles that are highly customizable.

The syntactic English-ASL translator allows for better and faster learning of sign language by stringing together input into one animation. It used Lexical Functional Transfer that allows translation governed by grammatical rules that can be customized by its administrators. If provided by enough amounts of animated base words and rules, it can translate English to sign language at a high accuracy that will be easy to understand by the learner.

VIII. RECOMMENDATION

For future iterations of the system, the author recommends the following actions taken.

To fully realize Vocalhands' potential as CMS, it would be a good idea to port it to a full-blown CMS. Doing this will provide the developers and administrators a better interface, easier maintenance and access to more features.

To improve the accuracy of translations done by the ASL translator, semantic-based parsing should be researched and implemented in place of the current syntactic parser. Signed animations should be heading towards more-humanlike animation and seamless transition between words. Using 3-D to fully render animations will allow a more realistic display and is also important because there are many signs that will not be totally understood in 2-D.

IX. BIBLIOGRAPHY

1. Duke, Irene. “Learn Sign Language in a Hurry”. Adams Media. 2009.
2. Hurlbut, Hope. “Philippine Signed Languages Survey: A Rapid Appraisal”. SIL International. 2008.
3. “American Sign Language”. http://en.wikipedia.org/wiki/American_sign_language. 2009
4. Speers, d’Armond. “Representation of American Sign Language for Machine Translation”. Doctoral Dissertation. Georgetown University. 2001.
5. “ASLInfo.com – About ASL”. http://aslinfo.com/aboutasl.cfm. 2009.
6. “Signing Online”. http://www.signingonline.com/.
7. “ASL University”. http://www.lifeprint.com/asl101/
8. “American Sign Language Browser”. http://commtechlab.msu.edu/sites/aslweb/browser.htm
9. Carpenter, Eileen, M.D. “Medical Sign Language Phrasebook”. http://www.angelfire.com/pa3/ecarpenter/ASLphrasebook.htm
10. Grieve-Smith, Angus. “English to American Sign Language Machine Translation of Weather Reports”. Proceedings of the Second High Desert Student Conference in Linguistics. 1999.
11. Huenerfauth, Matthew. “Generating American Sign Language Classifier Predicates for English-to-ASL Machine Translation”. Doctoral Dissertation. University of Pennsylvania. 2006.
12. Elliot R., Glauert J.R.W., Kennaway J.R., and Marshall I. “The development of language processing support for the ViSiCAST project”. In ASSETS 2000 - Proc. 4th International ACM Conference on Assistive Technologies, November 2000, Arlington, Virginia, pages 101-108, 2000.

13. Huenerfauth, Matthew. “A Survey and Critique of American Sign Language Natural Language Generation and Machine Translation Systems”. Technical Report MS-CIS-03-32, Computer and Information Science, University of Pennsylvania. 2003.
14. Zhao L., Kipper K., Schuler W., Volger C., Badler N., and Palmer M. “A Machine Translation System from English to American Sign Language”. In Association for Machine Translation in the Americas. Springer-Verlag, 2000.
15. Borra A., Chan E.A., Lim C.I., Tan R.B., and Tong M.C. “LFG-Based Machine Translation English for English and Filipino”. College of Computer Studies, De La Salle University. 2007.
16. Kudo I., and Nomura H. “Lexical-Functional Transfer: A Transfer Framework in a Machine Translation System Based on LFG”. The 11th International Conference on Computational Linguistics. Coling 1986 Volume 1.
17. Dan Klein and Christopher D. Manning. “Fast Exact Inference with a Factored Model for Natural Language Parsing”. In Advances in Neural Information Processing Systems 15 (NIPS 2002), Cambridge, MA: MIT Press, pp. 3-10. 2003.
18. Dan Klein and Christopher Manning. “Accurate Unlexicalized Parsing”. Proceedings of the 41st Meeting of the Association for Computational Linguistics, pp. 423-430.
19. “Hearing Impairment”. http://en.wikipedia.org/wiki/Deaf. 2009
20. Holt J., Hotto S., and Cole K. “Demographic Aspects of Hearing Impairment: Questions and Answers”. Gallaudet University. 1994.
21. “Sign Language”. Columbia Electronic Encyclopedia. Columbia University Press
22. “Natural Language Processing”. McGraw-Hill Encyclopedia of Science and Technology, 5th edition. The McGraw-Hill Companies, Inc.
23. “Stanford Parser: A Statistical Parser”. http://nlp.stanford.edu/software/lex-parser.shtml
24. Hutchins J., and Somers H. “An Introduction to Machine Translation”. London: Academic Press, 1992.

XII. APPENDIX

A. Source Code

CONTROLLERS

action_requests_controller.php
```
<?php

/**
 * @property ActionRequest $ActionRequest
 */

class ActionRequestsController extends AppController {
    var $name = 'ActionRequests';
    var $helpers = array('Html', 'Form', 'TinyMce');
    var $components = array('Email');

    function index() {
        $this->ActionRequest->recursive = 0;
    }

    function viewInbox() {
        $id = $this->Session->read('Auth.User.id');
        $this->set('in_messages',$this->paginate(($this-
>ActionRequest),array('recipient_id'=>$id)));
    }

    function viewSent() {
        $id = $this->Session->read('Auth.User.id');
        $this->set('out_messages',$this->paginate(($this-
>ActionRequest),array('sender_id'=>$id)));
    }

    function readMessage($id = null) {
        $user = $this->Session->read('Auth.User.id');
        $isRecipient = $this->ActionRequest-
>find('count',array('conditions'=>array('ActionRequest.id'=>$id, 'recipient_id'=>$user)));
        $isSender = $this->ActionRequest-
>find('count',array('conditions'=>array('ActionRequest.id'=>$id, 'sender_id'=>$user)));
        if(!$id) {
            $this->Session->setFlash(__('Invalid Message', true));
            $this->redirect(array('action' => 'index'));
        }
        else if(!$isRecipient && !$isSender) {
            $this->Session->setFlash(__('You cannot read this message.', true));
            $this->redirect(array('action' => 'index'));
        }
        else {
            $this->set('message',$this->ActionRequest->read(null,$id));
        }
    }

    function composeMessage($id = null) {
        $sender = $this->Session->read('Auth.User.id');
        if (!empty($this->data)) {
            $this->data['ActionRequest']['sender_id'] = $sender;
            date_default_timezone_set('Asia/Manila');
            $this->data['ActionRequest']['datetime'] = date("Y-m-d H:i:s");
            $this->ActionRequest->create();
            if ($this->ActionRequest->save($this->data)) {
                //do Email logic here
                $rdetails = $this->ActionRequest->Recipient-
>find('first',array('condition'=>array('Recipient.id'=>$this-
>data['ActionRequest']['recipient_id']), 'fields'=>'email'));
```

```
                $sdetails = $this->ActionRequest->Sender-
>find('first',array('condition'=>array('Sender.id'=>$sender), 'fields'=>'full_name'));
                $this->Email->to = $rdetails['Recipient']['email'];
                $this->Email->from = 'notifications@vocalhands.net';
                $this->Email->subject = 'New Message Notification';
                $content = 'Hi '.$sdetails['Sender']['full_name'].', has sent you a message. 
Please login to your account to view it.';
                $this->Email->send($content);
                //end Email logic
                $this->Session->setFlash(__('Message Sent.', true));
                $this->redirect(array('action' => 'viewSent'));
            } else {
                $this->Session->setFlash(__('The User could not be saved. Please, try again.', 
true));
            }
        }
        if($id) $this->set('selected',$id);
        else $this->set('selected',1);
        $recRaw = $this->ActionRequest->Recipient->find('all', 
array('conditions'=>array('Recipient.id !='=>$sender), 
'fields'=>array('Recipient.id','Recipient.full_name')));
        $recipients = array();
        foreach($recRaw as $record) $recipients[$record['Recipient']['id']] = 
$record['Recipient']['full_name'];
        $this->set(compact('recipients'));
    }
}
?>
```

articles_controller.php

```
<?php

class ArticlesController extends AppController {

    var $name = "Articles";
    var $helpers = array('Html', 'Form', 'TinyMce');
    var $components = array('Auth');
    var $paginate = array(
            'limit'=> 20,
            'order'=>array(
                            'date_time_modified' => 'desc'
            )

    );

    function index() {
        $this->Article->recursive = 0;
        $this->set('articles',$this->paginate($this->Article, array('status_id'=>2)));
    }

    function backend() {
        $this->Article->recursive = 0;
        $this->paginate = array(
                'conditions'=>array(
                        'user_id'=>$this->Auth->user('id')
                ),
                'limit'=>20,
                'order'=>array(
                        'Article.title'=>'asc'
                )
        );
        $this->set('articles',$this->paginate($this->Article));
    }

    function view($link = null) {
        if (!$link) {
            $this->Session->setFlash(__('Invalid Article', true));
            $this->redirect(array('action' => 'index'));
        }
        $this->Session->write('Article.activeArticle',Inflector::humanize($link));
        $id = $this->Article->getIdByTitle(Inflector::humanize($link));
        $this->Session->write('Article.activeArticle',$id);
        $this->set('article', $this->Article->read(null, $id));
    }

    function add() {
        if (!empty($this->data)) {
            $this->data['Article']['user_id'] = $this->Auth->user('id');
            date_default_timezone_set('Asia/Manila');
            $this->data['Article']['date_time_modified'] = date("Y-m-d H:i:s");
            $this->data['Article']['status_id'] = 1;
            $this->Article->create();
            if ($this->Article->save($this->data)) {
                $this->Session->setFlash(__('The Article has been saved', true));
                $this->redirect(array('action' => 'backend'));
            } else {
                $this->Session->setFlash(__('The Article could not be saved. Please, try again.',
true));
            }
        }
    }

    function edit($id = null) {
        if (!$id && empty($this->data)) $this->Session->setFlash(__('Invalid Article', true));
        if (!empty($this->data)) {
            $this->data['Article']['user_id'] = $this->Auth->user('id');
            date_default_timezone_set('Asia/Manila');
            $this->data['Article']['date_time_modified'] = date("Y-m-d H:i:s");
            $this->data['Article']['status_id'] = 1;
            if ($this->Article->save($this->data)) $this->Session->setFlash('The Article has been
saved.',true);
            else $this->Session->setFlash(__('The Article could not be saved. Please, try
again.', true));
            if($this->Auth->user('group_id') == '1') $this->redirect(array('action' =>
'backend'));
        }
        if (empty($this->data)) $this->data = $this->Article->read(null, $id);
    }

    function approve($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Article', true));
        else if ($this->Article->approve($id)) $this->Session->setFlash(__('Article approved',
true));
        else $this->Session->setFlash(__('The Article could not be approved. Please, try again.',
true));
        if($this->Auth->user('group_id') == '1') $this->redirect(array('action' => 'backend'));
        else $this->redirect(array('controller'=>'asl_administrators', 'action' =>
'viewArticleList'));
    }

    function cancel($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Article', true));
        else if ($this->Article->cancel($id)) $this->Session->setFlash(__('Article cancelled',
true));
        else $this->Session->setFlash(__('The Article could not be cancelled. Please, try
again.', true));
        if($this->Auth->user('group_id') == '1') $this->redirect(array('action' => 'backend'));
        else $this->redirect(array('controller'=>'asl_administrators', 'action' =>
'viewArticleList'));
    }

    function delete($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Article', true));
        else if ($this->Article->del($id)) $this->Session->setFlash(__('Article deleted', true));
        else $this->Session->setFlash(__('The Article could not be deleted. Please, try again.',
true));
        if($this->Auth->user('group_id') == '1') $this->redirect(array('action' => 'backend'));
        else $this->redirect(array('controller'=>'asl_administrators', 'action' =>
'viewArticleList'));
    }

    function beforeFilter() {
        $this->Auth->loginAction = array('controller' => 'users', 'action' => 'login');
        $this->Auth->allow('view','index');
        $this->Auth->loginRedirect = array('controller' => 'pages', 'action' => 'display');
        $this->Auth->authorize = 'controller';
    }

    function isAuthorized() {
        if(($this->Auth->user('group_id') == '3' || $this->Auth->user('group_id') == '1') &&
($this->action != 'approve')) return true;
        else if($this->Auth->user('group_id') == '3' && $this->action == 'approve') return true;
        else {
            $this->redirect(array('controller'=>'pages','action'=>'display'));
            return false;
        }
    }
}
?>
```

asl_administrators_controller.php

```
<?php

class AslAdministratorsController extends AppController {

    var $name = 'AslAdministrators';
    var $helpers = array('Html');
    var $uses = array('Lesson','Word','BaseWord','Article','GrammarRule');
    var $components = array('Auth');

    function aslManager() {

    }

    function viewArticleList() {
        $this->Article->recursive = 0;
        $this->paginate = array(
            'limit'=>10,
            'order'=>array(
                'Article.title'=>'asc'
            )
        );
        $this->set('articles', $this->paginate('Article'));
    }

    function viewLessonList() {
        $this->Lesson->recursive = 0;
        $this->paginate = array(
            'limit'=>10,
            'order'=>array(
                'Lesson.title'=>'asc'
            )
        );
        $this->Lesson->recursive = 0;
        $this->set('lessons', $this->paginate('Lesson'));
    }

    function viewGrammarRuleList() {
        $this->GrammarRule->recursive = 0;
        $this->paginate = array(
            'limit'=>10,
            'order'=>array(
                'GrammarRule.description'=>'asc'
            )
        );
        $this->set('grammarRules', $this->paginate('GrammarRule'));
    }

    function viewWordList() {
        $this->paginate = array(
            'limit'=>10,
            'order'=>array(
                'Word.name'=>'asc'
            ),
            'fields'=>array(
                'Word.id','Word.name','BaseWord.name','Status.name','Word.status_id'
            ),
            'conditions'=>array(
                "Word.name != BaseWord.name"
            )
        );
        $this->set('words', $this->paginate('Word'));
    }

    function viewBaseWordList() {
        $this->BaseWord->recursive = 0;
        $this->paginate = array(
            'limit'=>10,
            'order'=>array(
                'BaseWord.name'=>'asc'
            ),
        );
        $this->set('baseWords', $this->paginate('BaseWord'));
    }

    function beforeFilter() {
        $this->Auth->loginAction = array('controller' => 'users', 'action' => 'login');
        $this->Auth->loginRedirect = array('controller' => 'pages', 'action' => 'display');
        $this->Auth->authorize = 'controller';
    }

    function isAuthorized() {
        if($this->Auth->user('group_id') == '3') return true;
        else {
            $this->redirect(array('controller'=>'pages','action'=>'display'));
            return false;
        }
    }
}
?>
```

base_words_controller.php

```
<?php

/**
 * @property Word $Word
 * @property BaseWord $BaseWord
 */
class BaseWordsController extends AppController {

    var $name = 'BaseWords';
    var $helpers = array('Html', 'Form');
    var $components = array('Auth');
    var $paginate = array(
            'limit' => 10,
            'order' => array(
                'BaseWord.name' => 'asc'
            )
    );

    function backend() {
        if(!empty($this->data)) {
            $criteria = $this->data['searchCriteria'];
            $this->paginate = array(
                'conditions'=>array(
                    'BaseWord.name'=>$criteria
                )
            );
        }
        $this->BaseWord->recursive = 0;
        $this->set('baseWords', $this->paginate());
    }

    function openWords() {
        $this->BaseWord->recursive = 0;
        $this->set('baseWords', $this->paginate('BaseWord',array('BaseWord.status_id'=>3)));
    }

    function reservedWords() {
        $this->BaseWord->recursive = 0;
        $this->set('baseWords', $this-
>paginate('BaseWord',array('BaseWord.status_id'=>4,'BaseWord.user_id'=>$this->Auth-
>user('id'))));
    }

    function add() {
        if (!empty($this->data)) {
            $this->BaseWord->create();
            if ($this->BaseWord->save($this->data)) {

                $this->BaseWord->Word->create();
                $synArr = array();
                $synArr['Word']['name'] = $this->data['BaseWord']['name'];
                $synArr['Word']['base_word_id'] = $this->BaseWord->getLastInsertID();
                $synArr['Word']['status_id'] = 1;
                $this->BaseWord->Word->save($synArr);

                $this->Session->setFlash(__('The BaseWord has been saved', true));
                $this->redirect(array('action' => 'backend'));
            } else {
                $this->Session->setFlash(__('The BaseWord could not be saved. Please, try
again.', true));
            }
        }
        $sections = $this->BaseWord->Section->find('list');
        $statuses = $this->BaseWord->Status->find('list');
        $this->set(compact('statuses','sections'));
    }

    function edit($id = null) {
        $oldName = $this->BaseWord->read(array('name'), $id);
        if (!$id && empty($this->data)) {
            $this->Session->setFlash(__('Invalid BaseWord', true));
            $this->redirect(array('action' => 'backend'));
        }
        if (!empty($this->data)) {
            if ($this->BaseWord->save($this->data)) {

                $synId = $this->BaseWord->Word->getIdByNameAndBase($id,
$oldName['BaseWord']['name']);
                if($synId) {
                    $synName = $this->data['BaseWord']['name'];
                    $synStatId = $this->data['BaseWord']['status_id'];
                    $query = "UPDATE `vocalhands`.`words` SET `name` = '$synName',
`status_id`=$synStatId WHERE `words`.`id` =$synId";
                    //print $query;
                    $this->BaseWord->query($query);
                }

                $this->Session->setFlash(__('The BaseWord has been saved', true));
                $this->redirect(array('action' => 'backend'));
            } else {
                $this->Session->setFlash(__('The BaseWord could not be saved. Please, try
again.', true));
            }
        }
        if (empty($this->data)) {
            $this->data = $this->BaseWord->read(null, $id);
            $status = 1;
            if($this->data['BaseWord']['status_id']==3 || $this-
>data['BaseWord']['status_id']==4) $status = $this->data['BaseWord']['status_id'];
            $this->set('status',$status);
        }
        $statuses = $this->BaseWord->Status->find('list');
        $sections = $this->BaseWord->Section->find('list');
        $this->set(compact('statuses','sections'));
    }

    function editAnimation($id = null) {
        if (!$id && empty($this->data)) {
            $this->Session->setFlash(__('Invalid Animation', true));
            $this->redirect(array('action' => 'reservedWords'));
        }
        if (!empty($this->data)) {
            if ($this->BaseWord->save($this->data)) {
                $this->Session->setFlash(__('The BaseWord has been saved', true));
                $this->redirect(array('action' => 'reservedWords'));
            } else {
                $this->Session->setFlash(__('The BaseWord could not be saved. Please, try 
again.', true));
            }
        }
        if (empty($this->data)) {
            $this->data = $this->BaseWord->read(null, $id);
        }
        $statuses = $this->BaseWord->Status->find('list');
        $sections = $this->BaseWord->Section->find('list');
        $this->set(compact('statuses','sections'));
    }

    function approve($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Basic Word', true));
        else if ($this->BaseWord->approve($id)) $this->Session->setFlash(__('Basic Word 
approved', true));
        else $this->Session->setFlash(__('The Basic Word could not be approved. Please, try 
again.', true));
        $this->adminRedirect();
    }

    function cancel($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Basic Word', true));
        else if ($this->BaseWord->cancel($id)) $this->Session->setFlash(__('Basic Word 
cancelled', true));
        else $this->Session->setFlash(__('The Basic Word could not be cancelled. Please, try 
again.', true));
        $this->adminRedirect();
    }

    function reserve($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Basic Word', true));
        else if ($this->BaseWord->reserve($id,$this->Auth->user('id'))) $this->Session-
>setFlash('Basic Word reserved','flash_success');
        else $this->Session->setFlash(__('The Basic Word could not be reserved. Please, try 
again.', true));
        if($this->Auth->user('group_id') != '2') $this->adminRedirect();
        $this->redirect(array('action'=>'openWords'));
    }

    function open($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Basic Word', true));
        else if ($this->BaseWord->free($id)) $this->Session->setFlash(__('Basic Word freed', 
true));
        else $this->Session->setFlash(__('The Basic Word could not be opened. Please, try 
again.', true));
        if($this->Auth->user('group_id') != '2') $this->adminRedirect();
        $this->redirect(array('action'=>'reservedWords'));
    }

    function delete($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for BaseWord', true));
        else if ($this->BaseWord->del($id)) $this->Session->setFlash(__('BaseWord deleted', 
true));
        else $this->Session->setFlash(__('The BaseWord could not be deleted. Please, try again.', 
true));
        $this->adminRedirect();
    }
    private function adminRedirect() {
        if($this->Auth->user('group_id') == '1') $this->redirect(array('action' => 'backend'));
        else $this->redirect(array('controller'=>'asl_administrators', 'action' =>
'viewBaseWordList'));
    }

    function beforeFilter() {
        $this->Auth->loginAction = array('controller' => 'users', 'action' => 'login');
        $this->Auth->loginRedirect = array('controller' => 'pages', 'action' => 'display');
        $this->Auth->authorize = 'controller';
    }

    function isAuthorized() {
        if(($this->Auth->user('group_id') == '2') && ($this->action == 'editAnimation' ||
                $this->action == 'openWords' ||
                $this->action == 'reservedWords' ||
                $this->action == 'open' ||
                $this->action == 'reserve')) return true;
        else if(($this->Auth->user('group_id') == '3' || $this->Auth->user('group_id') == '1') &&
($this->action != 'approve')) return true;
        else if($this->Auth->user('group_id') == '3' && $this->action == 'approve') return true;
        else {
            $this->redirect(array('controller'=>'pages','action'=>'display'));
            return false;
        }
    }

}
?>
```

exam_properties_controller.php

```
<?php

/**
 * @property Exam $Exam
 */

class ExamPropertiesController extends AppController {

    var $name = 'ExamProperties';
    var $helpers = array('Html', 'Form');
    var $components = array('Auth');

    function setProperties() {
        if(!empty($this->data)) {
            Configure::store('ExamProperty','exam_properties',array('numItems'=>$this-
>data['numItems']));
        }
        $nums = array();
        for($i=1;$i<=10;$i++) $nums[$i] = $i;
        $this->set('optionsArray',$nums);
        Configure::load('exam_properties');
        $this->set('selected',Configure::read('ExamProperty.numItems'));
        $this->set('examProperties',$this->ExamProperty->find('all'));
    }

    function flickSwitch($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid Property', true));
        else if ($this->ExamProperty->flickSwitch($id)) $this->Session->setFlash(__('Settings
changed.', true));
        else $this->Session->setFlash(__('Change settings failed.', true));
        $this->redirect(array('action' => 'setProperties'));
    }

    function beforeFilter() {
        $this->Auth->loginAction = array('controller' => 'users', 'action' => 'login');
        $this->Auth->loginRedirect = array('controller' => 'pages', 'action' => 'display');
        $this->Auth->authorize = 'controller';
    }
    function isAuthorized() {
        if($this->Auth->user('group_id') == '1') return true;
        else {
            $this->redirect(array('controller'=>'pages','action'=>'display'));
            return false;
        }
    }
}
?>
<?php
```

exams_controller.php

```
class ExamsController extends AppController {

    var $name = 'Exams';
    var $helpers = array('Html', 'Form', 'Flash');
    var $uses = array('ExamProperty','Section','BaseWord');

    function index() {
        $sections = $this->Section->find('all');
        $sectionArr = array();
        foreach($sections as $section) {
            $sec_id = $section['Section']['id'];
            $countS = $this->BaseWord->find('count',
array('conditions'=>array('Section.id'=>$sec_id)));
            $sectionArr[$sec_id]['id'] = $sec_id;
            $sectionArr[$sec_id]['name'] = $section['Section']['name'];
            $sectionArr[$sec_id]['size'] = $countS;
        }
        $this->set('sections',$sectionArr);
    }

    function startExam($id = null) {
        if(!$id) {
            $this->Session->setFlash('Invalid Difficulty!',true);
            $this->redirect(array('action'=>'index'));
        }
        $this->Session->write('Exam.difficulty',$id);
        $this->Session->write('Exam.index',1);
        $this->Session->write('Exam.score',0);
        $this->redirect(array('action'=>'viewQuestion'));
    }

    function getResult() {
        Configure::load('exam_properties');
        $score = $this->Session->read('Exam.score');
        $total =  Configure::read('ExamProperty.numItems');
        $percent = $this->Session->read('Exam.score') / Configure::read('ExamProperty.numItems');
        if($percent == 1) $message = "Congratulations, you're a sign language expert!";
        else if($percent >= 0.5) $message = "Not bad.";
        else $message = "Boo! Practice more.";
        $this->set(compact('score','total','message'));
    }

    function viewQuestion() {
        Configure::load('exam_properties');
        if($this->Session->read('Exam.index') > Configure::read('ExamProperty.numItems')) $this-
>redirect(array('action'=>'getResult'));
        $formats = $this->ExamProperty->find('all',array('conditions'=>array('enabled'=>true)));
        if(!count($formats)) {
            $this->redirect(array('action'=>'getResult'));
        }
        $rand = rand(0,count($formats)-1);
        for($i=0;$i<count($formats);$i++) {
            if($i == $rand && $formats[$i]['ExamProperty']['enabled']) {
                if($formats[$i]['ExamProperty']['name'] == 'multiple-choice') $this-
>redirect(array('action'=>'viewMultipleChoiceQuestion'));
                if($formats[$i]['ExamProperty']['name'] == 'true-false') $this-
>redirect(array('action'=>'viewTrueFalseQuestion'));
            }
        }
        $this->redirect(array('action'=>'viewQuestion'));
    }

    function viewMultipleChoiceQuestion($answer = null) {
        $this->viewQuestionHelper($answer,3);
    }

    function viewTrueFalseQuestion($answer = null) {
        $this->viewQuestionHelper($answer,2);
    }

    private function viewQuestionHelper($answer = null, $limit = null) {
        $this->evaluateAnswer($answer);
        $wordList = $this->BaseWord-
>find('all',array('conditions'=>array('BaseWord.section_id'=>$this->Session-
>read('Exam.difficulty'), 'Status.name'=>'accepted'),'order'=>'rand()', 'limit'=>$limit));
        $this->Session->write('Exam.currentQuestion',$wordList);
        $this->Session->write('Exam.correctAnswer',rand(0,($limit-1)));
    }

    private function evaluateAnswer($answer = null) {
        if($answer != null) {
            if($this->Session->read('Exam.correctAnswer') == $answer) {
                $currentScore = $this->Session->read('Exam.score');
                $this->Session->write('Exam.score',1+$currentScore);
            }
            $currentIndex = $this->Session->read('Exam.index');
            $this->Session->write('Exam.index',1+$currentIndex);
            $this->redirect(array('action'=>'viewQuestion'));
        }
    }
}

?>
```

grammar_rules_controller.php

```
<?php

/**
 * @property GrammarRule $GrammarRule
 */
class GrammarRulesController extends AppController {

    var $name = 'GrammarRules';
    var $helpers = array('Html', 'Form');
    var $components = array('Auth');
    var $paginate = array(
            'limit'=>10,
            'order'=>array(
                            'GrammarRule.description'=>'asc'
            )
    );

    function index() {
        $this->GrammarRule->recursive = 0;
        $this->set('grammarRules', $this->paginate());
    }

    function view($id = null) {
        if (!$id) {
            $this->Session->setFlash(__('Invalid Rule', true));
            $this->redirect(array('action' => 'index'));
        }
        $this->set('grammarRule', $this->GrammarRule->read(null,$id));
    }

    function translateLfg() {
        $rules = $this->GrammarRule-
>find('all',array('conditions'=>array('Status.name'=>'accepted'),'order'=>array('length(english_r
ule)'=>'desc')));
        $dom = new DOMDocument();
        $dom->preserveWhiteSpace = false;
        $dom->load('java/output.xml');
        $nodeList = $dom->getElementsByTagName('node');
        foreach($rules as $rule) {
            $ruleArray = $this->buildRuleArray($rule['GrammarRule']['english_rule']);
            foreach($nodeList as $currentNode) {
                $chosenArray = $this->doesFollowRule($currentNode, $ruleArray);
                if($chosenArray) {
                    $aslRuleArray = $this->buildRuleArray($rule['GrammarRule']['asl_rule']);
                    $newTree = $this->constructNewTree($dom,$chosenArray,$aslRuleArray);
                    $currentNode->parentNode->replaceChild($newTree,$currentNode);
                }
            }
        }
        $answerString = (string)$nodeList->item(0)->nodeValue;
        $answerString = preg_replace('/\s\s+/', ' ', $answerString);
        $this->Session->write('Sentence.asl',$answerString);
        $this->redirect(array('controller'=>'words','action'=>'outputResult'));
    }

    private function doesFollowRule($aCurrentNode = null, $ruleArray = null) {
        //check if root doesn't follow
        if(!$this->equalValues($this->getValue($aCurrentNode),$ruleArray[0])) return false;
        else {
            $currentNode = $aCurrentNode->cloneNode(true);
            $chosenArray = array();
            $chosenArray[$ruleArray[0]] = $currentNode->cloneNode(true);
            $childrenNodes = $currentNode->childNodes;
            $parentNode = $currentNode->cloneNode(true);
            for($i=1;$i<count($ruleArray);$i++) {
                if($ruleArray[$i]!=']') {
                    $k = 0;
                    foreach($childrenNodes as $node) {
                        if($this->equalValues($this->getValue($node),$ruleArray[$i])) {
                            $parentNode = $currentNode->cloneNode(true);
                            $currentNode = $node->cloneNode(true);
                            $chosenArray[$ruleArray[$i]] = $currentNode->cloneNode(true);
                            $k++;
                        }
                    }
                    if($k==0) return false;
                }
                else $currentNode = $parentNode->cloneNode(true);
                $childrenNodes = $currentNode->childNodes;
            }
        }
        return $chosenArray;
    }

    private function equalValues($nodeString = null, $ruleString = null) {
        if(strlen($ruleString)<2) {
            if(strcasecmp($nodeString,$ruleString)==0) return true;
            else return false;
        }
        else {
            if(strlen($ruleString)<=strlen($nodeString)) {
                if(strcasecmp(substr($nodeString, 0, strlen($ruleString)),$ruleString)==0) {
//ruleString is smaller
                    return true;
                }
            }
            else {
                if(strcasecmp(substr($ruleString, 0, strlen($nodeString)),$nodeString)==0) {
//ruleString is larger
                    return true;
                }
            }
            return false;
        }
    }
    private function getValue($aNode = null) {
        return $aNode->getAttribute('value');
    }

    private function constructNewTree($domObject = null, $toConvert = null, $aslRule = null) {
        $head = $domObject->createElement('node');
        $head->setAttribute('value',$aslRule[0]);
        $currentNode = $head;
        for($i=1;$i<count($aslRule);$i++) {
            if($aslRule[$i]!=']') {
                if($aslRule[$i+1]!=']') {
                    $node = $domObject->createElement('node'); //check if not leaf
                    $node->setAttribute('value',$aslRule[$i]);
                }
                else $node = $toConvert[$aslRule[$i]]->cloneNode(true); //if leaf
                $currentNode->appendChild($node);
                $currentNode = $node;
            }
            else {
                $currentNode = $currentNode->parentNode;
            }
        }
        return $head;
    }

    private function buildRuleArray($ruleX = null) {
        $ruleArray = array();
        $currentTag = '';
        for($i=1;$i<strlen($ruleX);$i++) {
            $aChar = $ruleX[$i];
            if($aChar==']' || $aChar=='[') {
                if(strlen($currentTag)) array_push($ruleArray,$currentTag);
                if($aChar==']') array_push($ruleArray,']');
                $currentTag = '';
            }
            else $currentTag .= $aChar;
        }
        $ruleArray = $this->resolveDuplicates($ruleArray);
        return $ruleArray;
    }

    private function resolveDuplicates($array = null) {
        for($i=0;$i<count($array);$i++) {
            $k = 0;
            for($j=$i+1;$j<count($array);$j++) {
                if($i!=$j) {
                    if($array[$j]==']');
                    else if(strcmp($array[$i], $array[$j])==0) {
                        $tok = $array[$j];
                        $array[$j] = $tok.$k;
                        $k++;
                    }
                }
            }
        }
        return $array;
    }

    function add() {
        if (!empty($this->data)) {
            $this->GrammarRule->create();
            if ($this->GrammarRule->save($this->data)) {
                $this->Session->setFlash(__('The GrammarRule has been saved', true));
                $this->redirect(array('action' => 'index'));
            } else {
                $this->Session->setFlash(__('The GrammarRule could not be saved. Please, try
again.', true));
            }
        }
        $statuses = $this->GrammarRule->Status->find('list');
        $this->set(compact('statuses'));
    }

    function edit($id = null) {
        if (!$id && empty($this->data)) {
            $this->Session->setFlash(__('Invalid GrammarRule', true));
            $this->redirect(array('action' => 'index'));
        }
        if (!empty($this->data)) {
            if ($this->GrammarRule->save($this->data)) {
                $this->Session->setFlash(__('The GrammarRule has been saved', true));
                $this->redirect(array('action' => 'index'));
            } else {
                $this->Session->setFlash(__('The GrammarRule could not be saved. Please, try
again.', true));
            }
        }
        if (empty($this->data)) {
            $this->data = $this->GrammarRule->read(null, $id);
        }
        $statuses = $this->GrammarRule->Status->find('list');
        $this->set(compact('statuses'));
    }

    function approve($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Basic Word', true));
        else if ($this->GrammarRule->approve($id)) $this->Session->setFlash(__('Basic Word
approved', true));
        else $this->Session->setFlash(__('The Basic Word could not be approved. Please, try
again.', true));
        $this->adminRedirect();
    }

    function cancel($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Basic Word', true));
        else if ($this->GrammarRule->cancel($id)) $this->Session->setFlash(__('Basic Word
cancelled', true));
        else $this->Session->setFlash(__('The Basic Word could not be cancelled. Please, try
again.', true));
        $this->adminRedirect();
    }

    function delete($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for GrammarRule', true));
        else if ($this->GrammarRule->del($id)) $this->Session->setFlash(__('GrammarRule deleted',
true));
        else $this->Session->setFlash(__('The GrammarRule could not be deleted. Please, try
again.', true));
        $this->adminRedirect();
    }

    private function adminRedirect() {
        if($this->Auth->user('group_id') == '1') $this->redirect(array('action' => 'index'));
        else $this->redirect(array('controller'=>'asl_administrators', 'action' =>
'viewGrammarRuleList'));
    }

    function beforeFilter() {
        $this->Auth->loginAction = array('controller' => 'users', 'action' => 'login');
        $this->Auth->allow('translateLfg');
        $this->Auth->loginRedirect = array('controller' => 'pages', 'action' => 'display');
        $this->Auth->authorize = 'controller';
    }

    function isAuthorized() {
        if(($this->Auth->user('group_id') == '3' || $this->Auth->user('group_id') == '1') &&
($this->action != 'approve')) return true;
        else if($this->Auth->user('group_id') == '3' && $this->action == 'approve') return true;
        else {
            $this->redirect(array('controller'=>'pages','action'=>'display'));
            return false;
        }
    }

}
?>
```

lessons_controller.php

```
<?php

/**
 * @property Lesson $Lesson
 */
class LessonsController extends AppController {

    var $name = 'Lessons';
    var $helpers = array('Html', 'Form', 'Flash','TinyMce');
    var $components = array('Auth');
    var $paginate = array(
            'limit'=>10,
            'order'=>array(
                            'Lesson.order'=>'asc'
            )
    );

    private function getMax() {
        $maxQuery = $this->Lesson->find('all',array('fields'=>array('MAX(Lesson.order) as
max_order')));
        return $maxQuery[0][0]['max_order'];
    }

    private function getMin() {
        $minQuery = $this->Lesson->find('all',array('fields'=>array('MIN(Lesson.order) as
min_order')));
        return $minQuery[0][0]['min_order'];
    }

    function index() {
        $this->set('menu', $this->buildMenu());
    }

    private function buildMenu() {
        $someArr = array();
        $i = 0;
        foreach($this->Lesson->Section->getAllSections() as $section) {
            $someArr[$i]['name'] = $section['Section']['name'];
            $someArr[$i]['subOptions'] = $this->Lesson-
>find('all',array('conditions'=>array('Section.name'=>$section['Section']['name'],
'Status.name'=>'accepted'), 'fields'=>
array('title','id'),'order'=>array('Lesson.order'=>'asc')));
            $i++;
        }
        return $someArr;
    }

    function view($link = null) {
        if (!$link) {
            $this->Session->setFlash(__('Invalid Lesson', true));
            $this->redirect(array('action' => 'index'));
        }
        $this->Session->write('Lesson.activeLesson',Inflector::humanize($link));
        $id = $this->Lesson->getIdByTitle(Inflector::humanize($link));
        $this->Session->write('Lesson.activeLesson',$id);
        $this->set('lesson', $this->Lesson->read(null, $id));
        $this->set('menu', $this->buildMenu());
    }

    function add() {
        if (!empty($this->data)) {
            $this->Lesson->create();
            if ($this->Lesson->save($this->data)) {
                $this->Session->setFlash(__('The Lesson has been saved', true));
                $this->redirect(array('action' => 'backend'));
            } else {
                $this->Session->setFlash(__('The Lesson could not be saved. Please, try again.', 
true));
            }
        }
        $this->saveJS();
        $statuses = $this->Lesson->Status->find('list');
        $sections = $this->Lesson->Section->find('list');
        $this->set(compact('statuses','sections','wordList'));
        $this->set('new_order', $this->getMax() + 1);
    }

    function saveJS() {
        $output = ''; // Here we buffer the JavaScript code we want to send to the browser.
        $delimiter = "\n"; // for eye candy... code gets new lines
        $this->loadModel('BaseWord');
        $wordList = $this->BaseWord->find('all', 
array('conditions'=>array('BaseWord.status_id'=>2),'order' => array('BaseWord.name'=>'asc')));
        $output .= 'var tinyMCELinkList = new Array(';
        $abspath = "../../words/view";

        foreach($wordList as $word) {
            $output .= $delimiter
                    . '["'
                    . utf8_encode($word['BaseWord']['name'].' ('.$word['BaseWord']['meaning'] . 
') ')
                    . '", "'
                    . utf8_encode($abspath.'/'.$word['BaseWord']['id'])
                    . '"],';
        }
        $output = substr($output, 0, -1); // remove last comma from array item list (breaks some 
browsers)
        $output .= $delimiter;
        $output .= ');';

        $myFile = "js/wordlist.js";
        $fh = fopen($myFile, 'w') or die("can't open file");
        fwrite($fh, $output);
        fclose($fh);
    }

    function edit($id = null) {
        if (!$id && empty($this->data)) {
            $this->Session->setFlash(__('Invalid Lesson', true));
            $this->redirect(array('action' => 'backend'));
        }
        if (!empty($this->data)) {
            if ($this->Lesson->save($this->data)) {
                $this->Session->setFlash(__('The Lesson has been saved', true));
                $this->redirect(array('action' => 'backend'));
            } else {
                $this->Session->setFlash(__('The Lesson could not be saved. Please, try again.', 
true));
            }
        }
        if (empty($this->data)) {
            $this->data = $this->Lesson->read(null, $id);
        }
        $this->saveJS();
        $statuses = $this->Lesson->Status->find('list');
        $sections = $this->Lesson->Section->find('list');
        $this->set(compact('statuses','sections'));
    }

    function approve($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Lesson', true));
        else if ($this->Lesson->approve($id)) $this->Session->setFlash(__('Lesson approved', 
true));
        else $this->Session->setFlash(__('The Lesson could not be approved. Please, try again.', 
true));
        if($this->Auth->user('group_id') == '1') $this->redirect(array('action' => 'backend'));
        else $this->redirect(array('controller'=>'asl_administrators', 'action' =>
'viewLessonList'));
    }

    function cancel($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Lesson', true));
        else if ($this->Lesson->cancel($id)) $this->Session->setFlash(__('Lesson cancelled',
true));
        else $this->Session->setFlash(__('The Lesson could not be cancelled. Please, try again.',
true));
        if($this->Auth->user('group_id') == '1') $this->redirect(array('action' => 'backend'));
        else $this->redirect(array('controller'=>'asl_administrators', 'action' =>
'viewLessonList'));
    }

    function delete($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Lesson', true));
        $q = $this->Lesson->find(array('Lesson.id'=>$id), array('Lesson.order'));
        if ($this->Lesson->del($id)) {
            $this->Lesson->updateOrder($q['Lesson']['order']);
            $this->Session->setFlash(__('Lesson deleted', true));
        }
        else $this->Session->setFlash(__('The Lesson could not be deleted. Please, try again.',
true));
        if($this->Auth->user('group_id') == '1') $this->redirect(array('action' => 'backend'));
        else $this->redirect(array('controller'=>'asl_administrators', 'action' =>
'viewLessonList'));
    }

    function raise_up($id = null) {
        $q = $this->Lesson->find(array('Lesson.id'=>$id), array('Lesson.order'));
        $this->Lesson->raiseUp($id, $q['Lesson']['order']);
        $this->redirect(array('action'=>'backend'));
    }

    function lower_down($id = null) {
        $q = $this->Lesson->find(array('Lesson.id'=>$id), array('Lesson.order'));
        $this->Lesson->lowerDown($id, $q['Lesson']['order']);
        $this->redirect(array('action'=>'backend'));
    }

    function beforeFilter() {
        $this->Auth->loginAction = array('controller' => 'users', 'action' => 'login');
        $this->Auth->allow('index','view');
        $this->Auth->loginRedirect = array('controller' => 'pages', 'action' => 'display');
        $this->Auth->authorize = 'controller';
    }

    function isAuthorized() {
        if(($this->Auth->user('group_id') == '3' || $this->Auth->user('group_id') == '1') &&
($this->action != 'approve')) return true;
        else if($this->Auth->user('group_id') == '3' && $this->action == 'approve') return true;
        else {
            $this->redirect(array('controller'=>'pages','action'=>'display'));
            return false;
        }
    }

    function backend() {
        $this->Lesson->recursive = 0;
        $this->set('lessons', $this->paginate());
        $max = $this->getMax();
        $min = $this->getMin();
        $this->set(compact('min','max'));
    }
}
?>
```

sections_controller.php

```
<?php
```

```
/**
 * @property Section $Section
 */

class SectionsController extends AppController {

    var $name = 'Sections';
    var $helpers = array('Html', 'Form');
    var $paginate = array(
            'order' => array(
                            'Section.order' => 'asc'
            )
    );

    private function getMax() {
        $maxQuery = $this->Section->find('all',array('fields'=>array('MAX(Section.order) as
max_order')));
        return $maxQuery[0][0]['max_order'];
    }

    private function getMin() {
        $minQuery = $this->Section->find('all',array('fields'=>array('MIN(Section.order) as
min_order')));
        return $minQuery[0][0]['min_order'];
    }

    function index() {
        $this->Section->recursive = 0;
        $this->set('sections', $this->paginate());
        $max = $this->getMax();
        $min = $this->getMin();
        $this->set(compact('min','max'));
    }

    function add() {
        if (!empty($this->data)) {
            $this->Section->create();
            if ($this->Section->save($this->data)) {
                $this->Session->setFlash(__('The Section has been saved', true));
                $this->redirect(array('action' => 'index'));
            } else {
                $this->Session->setFlash(__('The Section could not be saved. Please, try again.',
true));
            }
        }
        $this->set('new_order', $this->getMax() + 1);
    }

    function edit($id = null) {
        if (!$id && empty($this->data)) {
            $this->Session->setFlash(__('Invalid Section', true));
            $this->redirect(array('action' => 'index'));
        }
        if (!empty($this->data)) {
            if ($this->Section->save($this->data)) {
                $this->Session->setFlash(__('The Section has been saved', true));
                $this->redirect(array('action' => 'index'));
            } else {
                $this->Session->setFlash(__('The Section could not be saved. Please, try again.',
true));
            }
        }
        if (empty($this->data)) {
            $this->data = $this->Section->read(null, $id);
        }
    }

    function delete($id = null) {
        if (!$id) {
            $this->Session->setFlash(__('Invalid id for Section', true));
            $this->redirect(array('action' => 'index'));
        }
        $q = $this->Section->find(array('Section.id'=>$id), array('Section.order'));
        if ($this->Section->del($id)) {
            $this->Section->updateOrder($q['Section']['order']);
            $this->Session->setFlash(__('Section deleted', true));
            $this->redirect(array('action' => 'index'));
        }
        $this->Session->setFlash(__('The Section could not be deleted. Please, try again.',
true));
        $this->redirect(array('action' => 'index'));
    }

    function raise_up($id = null) {
        $q = $this->Section->find(array('Section.id'=>$id), array('Section.order'));
        $this->Section->raiseUp($id, $q['Section']['order']);
        $this->redirect(array('action'=>'index'));
    }

    function lower_down($id = null) {
        $q = $this->Section->find(array('Section.id'=>$id), array('Section.order'));
        $this->Section->lowerDown($id, $q['Section']['order']);
        $this->redirect(array('action'=>'index'));
    }

}
?>
```

users_controller.php

```
<?php

/**
 * @property User $User
 */
class UsersController extends AppController {

    var $name = 'Users';
    var $helpers = array('Html', 'Form');
    var $components = array('Auth');

    function index() {
        $this->User->recursive = 0;
        $this->set('users', $this->paginate());
        $this->set('user', $this->Session->read('User.group'));
    }

    function add() {
        if (!empty($this->data)) {
            $this->User->create();
            if ($this->User->save($this->data)) {
                $this->Session->setFlash(__('The User has been saved', true));
                $this->redirect(array('action' => 'index'));
            } else {
                $this->Session->setFlash(__('The User could not be saved. Please, try again.',
true));
            }
        }
        $groups = $this->User->Group->find('list');
        $this->set(compact('groups'));
    }

    function edit($id = null) {
        if (!$id && empty($this->data)) {
            $this->Session->setFlash(__('Invalid User', true));
            $this->redirect(array('action' => 'index'));
        }
        if (!empty($this->data)) {
            if ($this->User->save($this->data)) {
                $this->Session->setFlash(__('The User has been saved', true));
                $this->redirect(array('action' => 'index'));
            } else {
                $this->Session->setFlash(__('The User could not be saved. Please, try again.', 
true));
            }
        }
        if (empty($this->data)) {
            $this->data = $this->User->read(null, $id);
        }
        $groups = $this->User->Group->find('list');
        $this->set(compact('groups'));
    }

    function delete($id = null) {
        if (!$id) {
            $this->Session->setFlash(__('Invalid id for User', true));
            $this->redirect(array('action' => 'index'));
        }
        if ($this->User->del($id)) {
            $this->Session->setFlash(__('User deleted', true));
            $this->redirect(array('action' => 'index'));
        }
        $this->Session->setFlash(__('The User could not be deleted. Please, try again.', true));
        $this->redirect(array('action' => 'index'));
    }

    function login() {
    }

    function logout() {
        $this->Session->destroy();
        $this->redirect($this->Auth->logout());
    }

    function beforeFilter() {
        $this->Auth->loginAction = array('controller' => 'users', 'action' => 'login');
        $this->Auth->allow('login','logout');
        $this->Auth->authorize = 'controller';
    }

    function isAuthorized() {
        if($this->Auth->user('group_id') == '4') return true;
        else {
            $this->redirect(array('controller'=>'pages','action'=>'display'));
            return false;
        }
    }
}
?>
```

words_controller.php

```
<?php

/**
 * @property Word $Word
 * @property BaseWord $BaseWord
 */
class WordsController extends AppController {

    var $name = 'Words';
    var $helpers = array('Html', 'Form', 'Flash');
    var $components = array('Auth');

    function index() {
        $this->Word->recursive = 0;
        //$this->set('words', $this->paginate());

        $this->setMenu();
    }

    private function setMenu() {
        $currLetter = $this->Session->read('Word.activeLetter');
        if($currLetter == '')
```

```
            $currLetter = 'A';
        $this->set('currLetter',$currLetter);
        $this->set('activeWords',$this->Word->getWordListForLetter($currLetter));
        $this->set('letterList',$this->buildLetterList());
    }

    private function buildLetterList() {
        $letters = array();
        $allLetters = 
array('A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W'
,'X','Y','Z');
        $wordList = $this->Word->find('all',array('conditions'=>array('Status.name'=>'accepted',
'BaseWord.status_id'=>2),'fields'=>'name'));
        $tempList = '';
        //Get the first letter of each word and add it in tempList
        foreach($wordList as $word)
            $tempList .= substr($word['Word']['name'],0,1);
        for($i=0;$i<count($allLetters);$i++) {
            $letters[$i]['letter'] = $allLetters[$i];
            if(stristr($tempList,$allLetters[$i])!==false)
                $letters[$i]['count'] = 1;
            else
                $letters[$i]['count'] = 0;
        }
        return $letters;
    }

    function search() {
        $this->setMenu();
        $query = $this->data['Word']['searchFor'];
        if($query != null) {
            $this->goSearch($query);
        }
    }

    function goSearch($aToken = null) {
        $this->Session->write('Word.activeLetter',$aToken);
        $this->redirect(array('controller'=>'words','action'=>'index'));
    }

    function translate() {
        $query = $this->data['Word']['translateWhat'];
        if($query != null) {
            $this->Session->write('Word.isTranslating',true);
            $this->Session->write('Sentence.english',$query);
            //create input.txt file and write query to it
            $fh = fopen('java/input.txt','w') or die("can't open file!");
            fwrite($fh,$query);
            fclose($fh);
            //call grammar parser: arg[0] grammar to use; arg[1] input file; arg[2] output file;
            system('java -jar java/Grammar.jar java/englishPCFG.ser.gz java/input.txt 
java/output.xml');
            $this->redirect(array('controller'=>'grammar_rules','action'=>'translateLfg'));
        }
        else {
            $this->Session->write('Word.isTranslating',false);
        }
        $this->set('isTranslating',$this->Session->read('Word.isTranslating'));
    }

    function outputResult() {
        $this->set('toTranslate', $this->Session->read('Sentence.english'));
        $aslSen = $this->Session->read('Sentence.asl');
        $senArr =  array();
        $i = 0;
        foreach(explode(" ",$this->Session->read('Sentence.asl')) as $word) {
            $word = str_replace("\n", "", $word);
            $retrievedWord = $this->Word->find(array('Word.name LIKE'=>$word,
'Word.status_id'=>2, 'BaseWord.status_id'=>2));
            $senArr[$i]['word'] = $word;
            $senArr[$i]['available'] = count($retrievedWord['Word']);
            $senArr[$i]['data'] = $retrievedWord;
            $i++;
        }
        $animation = "";
        $availableWords = array();
        $unavailableWords = array();
        $aCtr = 0;
        $uCtr = 0;
        foreach($senArr as $word) {
            if($word['available']) {
                if($aCtr!=0) $animation .= '|';
                $animation .=
'http://'.$_SERVER['SERVER_NAME'].$word['data']['BaseWord']['animation_link'];
                $availableWords[$aCtr]['word'] = $word['word'];
                $aCtr++;
            }
            else {
                $unavailableWords[$uCtr]['word'] = $word['word'];
                $uCtr++;
            }
        }
        $this->set('translated',$this->Session->read('Sentence.asl'));
        $this->set('availableWords', $availableWords);
        $this->set('unavailableWords', $unavailableWords);
        $this->set('animation', $animation);
    }

    function view($link = null) {
        if (!$link) {
            $this->Session->setFlash(__('Invalid Word', true));
            $this->redirect(array('action' => 'index'));
        }
        $id = $this->Word->getIdByName(Inflector::humanize($link));
        $this->set('word', $this->Word->read(null, $id['Word']['id']));

        $this->setMenu();
    }

    function add() {
        if (!empty($this->data)) {
            $this->Word->create();
            if ($this->Word->save($this->data)) {
                $this->Session->setFlash(__('The Word has been saved', true));
                $this->redirect(array('action' => 'backend'));
            } else {
                $this->Session->setFlash(__('The Word could not be saved. Please, try again.',
true));
            }
        }
        $baseWords = $this->Word->BaseWord-
>find('list',array('order'=>array('BaseWord.name'=>'asc')));
        $statuses = $this->Word->Status->find('list');
        $this->set(compact('statuses','baseWords'));
    }

    function edit($id = null) {
        if (!$id && empty($this->data)) {
            $this->Session->setFlash(__('Invalid Word', true));
            $this->redirect(array('action' => 'backend'));
        }
        if (!empty($this->data)) {
            if ($this->Word->save($this->data)) {
                $this->Session->setFlash(__('The Word has been saved', true));
                $this->redirect(array('action' => 'backend'));
            } else {
                $this->Session->setFlash(__('The Word could not be saved. Please, try again.',
true));
            }
        }
        if (empty($this->data)) {
            $this->data = $this->Word->read(null, $id);
        }
        $baseWords = $this->Word->BaseWord-
>find('list',array('order'=>array('BaseWord.name'=>'asc')));
        $statuses = $this->Word->Status->find('list');
        $this->set(compact('statuses','baseWords'));
    }

    function approve($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Word', true));
        else if ($this->Word->approve($id)) $this->Session->setFlash(__('Word approved', true));
        else $this->Session->setFlash(__('The Word could not be approved. Please, try again.',
true));
        if($this->Auth->user('group_id') == '1') $this->redirect(array('action' => 'backend'));
        else $this->redirect(array('controller'=>'asl_administrators', 'action' =>
'viewWordList'));
    }

    function cancel($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Word', true));
        else if ($this->Word->cancel($id)) $this->Session->setFlash(__('Word cancelled', true));
        else $this->Session->setFlash(__('The Word could not be cancelled. Please, try again.',
true));
        if($this->Auth->user('group_id') == '1') $this->redirect(array('action' => 'backend'));
        else $this->redirect(array('controller'=>'asl_administrators', 'action' =>
'viewWordList'));
    }

    function delete($id = null) {
        if (!$id) $this->Session->setFlash(__('Invalid id for Word', true));
        else if ($this->Word->del($id)) $this->Session->setFlash(__('Word deleted', true));
        else $this->Session->setFlash(__('The Word could not be deleted. Please, try again.',
true));
        if($this->Auth->user('group_id') == '1') $this->redirect(array('action' => 'backend'));
        else $this->redirect(array('controller'=>'asl_administrators', 'action' =>
'viewWordList'));
    }

    function beforeFilter() {
        $this->Auth->loginAction = array('controller' => 'users', 'action' => 'login');
        $this->Auth->allow('index','view','search','translate','goSearch','outputResult');
        $this->Auth->loginRedirect = array('controller' => 'pages', 'action' => 'display');
        $this->Auth->authorize = 'controller';
    }

    function isAuthorized() {
        if(($this->Auth->user('group_id') == '3' || $this->Auth->user('group_id') == '1') &&
($this->action != 'approve')) return true;
        else if($this->Auth->user('group_id') == '3' && $this->action == 'approve') return true;
        else {
            $this->redirect(array('controller'=>'pages','action'=>'display'));
            return false;
        }
    }

    function backend() {
        if(!empty($this->data)) {
            $criteria = $this->data['searchCriteria'];
            $conditionsArr = array('Word.name != BaseWord.name','Word.name'=>$criteria);
        }
        else {
            $conditionsArr = array('Word.name != BaseWord.name');
        }
        $this->Word->recursive = 0;
        $this->paginate = array(
                'limit'=>10,
                'conditions'=>$conditionsArr
        );
        $this->set('words', $this->paginate());
    }
}
?>
```

MODELS

action_request.php
```
<?php

/**
 * @property ActionRequest $ActionRequest
 */

class ActionRequest extends AppModel {
    var $name = 'ActionRequest';
    var $validate = array(
            'sender_id' => array('numeric'),
            'recipient_id' => array('numeric'),
            'title' => array('notempty'),
            'message' => array('notempty'),
            'status_id' => array('numeric')
    );
    var $belongsTo = array(
            'Sender' => array(
                            'className' => 'User',
                            'foreignKey' => 'sender_id'
            ),
            'Recipient' => array(
                            'className' => 'User',
                            'foreignKey' => 'recipient_id'
            )
    );
}

?>
```

article.php
```
<?php

class Article extends AppModel {

    var $name = "Article";
    var $validate = array (
            'title' => array('/[A-Za-z0-9_ ]+/'),
            'content' => array('notempty'),
            'user_id' => array('numeric'),
            'status_id' => array('numeric')
    );

    var $belongsTo = array(
            'Status' => array(
                            'className' => 'Status',
                            'foreignKey' => 'status_id',
            ),
            'Author' => array(
                            'className' => 'User',
                            'foreignKey' => 'user_id'
            )
    );

    function getIdByTitle($aTitle = null) {
        $arr = $this->find(array('title'=>$aTitle), array('id'));
        return $arr['Article']['id'];
    }

    function approve($id = null) {
        $this->read(null, $id);
        return $this->saveField('status_id',2);
    }

    function cancel($id = null) {
        $this->read(null, $id);
        return $this->saveField('status_id',1);
    }
}

?>
<?php

/**
 * @property Word $Word
 * @property Section $Section
 * @property Status $Status
 */
class BaseWord extends AppModel {

    var $name = 'BaseWord';
    var $validate = array(
            'name' => array('notempty'),
            'section_id' => array('numeric')
    );

    //The Associations below have been created with all possible keys, those that are not needed
can be removed
    var $belongsTo = array(
            'Section' => array(
                            'className' => 'Section',
                            'foreignKey' => 'section_id',
            ),
            'Status' => array(
                            'className' => 'Status',
                            'foreignKey' => 'status_id',
            ),
            'Developer' => array(
                            'className' => 'User',
                            'foreignKey' => 'user_id',
            )
    );

    var $hasMany = array(
            'Word' => array(
                            'className' => 'Word',
                            'foreignKey' => 'base_word_id',
                            'dependent' => true,
            )
    );

    function approve($id = null) {
        $this->read(null, $id);
        return $this->saveField('status_id',2);
    }

    function cancel($id = null) {
        $this->read(null, $id);
        return $this->saveField('status_id',1);
    }

    function reserve($id = null, $user_id = null) {
        $this->read(null, $id);
        $this->saveField('status_id',4);
        $this->saveField('user_id',$user_id);
        return true;
    }

    function free($id = null) {
        $this->read(null, $id);
        return $this->saveField('status_id',3);
    }

}
?>
```

exam.php

```
<?php
```

```
/**
 * @property Exam $Exam
 */
class ExamProperty extends AppModel {

    var $name = 'ExamProperty';

    function flickSwitch($id = null) {
        $this->read(null, $id);
        return $this->saveField('enabled',!$this->field('enabled'));
    }
}

?>
```

grammar_rule.php

```
<?php

/**
 * @property GrammarRule $GrammarRule
 */

class GrammarRule extends AppModel {
    var $name = 'GrammarRule';
    var $validate = array(
            'description' => array('notempty'),
            'english_rule' => array('notempty'),
            'asl_rule' => array('notempty'),
            'status_id' => array('numeric')
    );
    var $belongsTo = array(
            'Status' => array(
                            'className' => 'Status',
                            'foreignKey' => 'status_id'
            )
    );

    function approve($id = null) {
        $this->read(null, $id);
        return $this->saveField('status_id',2);
    }

    function cancel($id = null) {
        $this->read(null, $id);
        return $this->saveField('status_id',1);
    }
}

?>
```

group.php

```
<?php
class Group extends AppModel {

        var $name = 'Group';
        var $validate = array(
                'name' => array('notempty')
        );

        //The Associations below have been created with all possible keys, those that are not
needed can be removed
        var $hasMany = array(
                'User' => array(
                        'className' => 'User',
                        'foreignKey' => 'group_id',
                        'dependent' => false,
                )
        );

}
```

```
?>
```

lesson.php

```
<?php

/**
 * @property Lesson $Lesson
 * @property Status $Status
 * @property Section $Section
 */

class Lesson extends AppModel {

    var $name = 'Lesson';
    var $validate = array(
            'title' => array('/[A-Za-z0-9_ ]+/'),
            'description' => array('notempty'),
            'status_id' => array('numeric')
    );

    //The Associations below have been created with all possible keys, those that are not needed
can be removed
    var $belongsTo = array(
            'Status' => array(
                            'className' => 'Status',
                            'foreignKey' => 'status_id',
            ),
            'Section' => array(
                            'className' => 'Section',
                            'foreignKey' => 'section_id',
            )
    );

    function getIdByTitle($aTitle = null) {
        $arr = $this->find(array('title'=>$aTitle), array('id'));
        return $arr['Lesson']['id'];
    }

    function approve($id = null) {
        $this->read(null, $id);
        return $this->saveField('status_id',2);
    }

    function cancel($id = null) {
        $this->read(null, $id);
        return $this->saveField('status_id',1);
    }

    function raiseUp($id = null, $order = null) {
        $high = $order - 1;
        $highArr = $this->find('first',array('conditions'=>array('Lesson.order'=>$high)));
        $this->swapOrder($id,$order,$highArr['Lesson']['id'],$high);
    }

    function lowerDown($id = null, $order = null) {
        $low = $order + 1;
        $lowArr = $this->find('first',array('conditions'=>array('Lesson.order'=>$low)));
        $this->swapOrder($id,$order,$lowArr['Lesson']['id'],$low);
    }

    /**
     *Swaps the order of two records.
     * @param int $firstId id of first record
     * @param int $firstOrder order of first record
     * @param int $secondId id of second record
     * @param int $secondOrder order of second record
     */
    private function swapOrder($firstId = null, $firstOrder = null, $secondId = null,
$secondOrder = null) {
        $this->id = $firstId;
        $this->saveField('order', $secondOrder);
        $this->id = $secondId;
        $this->saveField('order', $firstOrder);
    }

    function updateOrder($order = null) {
        $toUpdate = $this->find('all',array('conditions'=>array('Lesson.order >'=>$order)));
        foreach($toUpdate as $record) {
            $this->id = $record['Lesson']['id'];
            $this->saveField('order',$record['Lesson']['order'] - 1);
        }
    }
}
?>
```

section.php

```
<?php
/**
 * @property Section $Section
 */
class Section extends AppModel {

    var $name = 'Section';
    var $validate = array(
            'name' => array('notempty')
    );

    //The Associations below have been created with all possible keys, those that are not needed
can be removed
    var $hasMany = array(
            'Lesson' => array(
                            'className' => 'Lesson',
                            'foreignKey' => 'section_id',
                            'dependent' => false,
            ),
            'BaseWord' => array(
                            'className' => 'BaseWord',
                            'foreignKey' => 'section_id',
                            'dependent' => false,
            ),
    );

    function getAllSections() {
        return $this->find('all',array('fields'=>'name','order'=>array('Section.order'=>'asc')));
    }

    function raiseUp($id = null, $order = null) {
        $high = $order - 1;
        $highArr = $this->find('first',array('conditions'=>array('Section.order'=>$high)));
        $this->swapOrder($id,$order,$highArr['Section']['id'],$high);
    }

    function lowerDown($id = null, $order = null) {
        $low = $order + 1;
        $lowArr = $this->find('first',array('conditions'=>array('Section.order'=>$low)));
        $this->swapOrder($id,$order,$lowArr['Section']['id'],$low);
    }

    /**
     *Swaps the order of two records.
     * @param int $firstId id of first record
     * @param int $firstOrder order of first record
     * @param int $secondId id of second record
     * @param int $secondOrder order of second record
     */
    private function swapOrder($firstId = null, $firstOrder = null, $secondId = null,
$secondOrder = null) {
        $this->id = $firstId;
        $this->saveField('order', $secondOrder);
        $this->id = $secondId;
        $this->saveField('order', $firstOrder);
    }
    function updateOrder($order = null) {
        $toUpdate = $this->find('all',array('conditions'=>array('Section.order >'=>$order)));
        foreach($toUpdate as $record) {
            $this->id = $record['Section']['id'];
            $this->saveField('order',$record['Section']['order'] - 1);
        }
    }
}
?>
```

status.php

```
<?php
class Status extends AppModel {

    var $name = 'Status';
    var $validate = array(
            'name' => array('notempty')
    );

    //The Associations below have been created with all possible keys, those that are not needed
can be removed
    var $hasMany = array(
            'ActionRequest' => array(
                            'className' => 'ActionRequest',
                            'foreignKey' => 'status_id',
                            'dependent' => false
            ),
            'Lesson' => array(
                            'className' => 'Lesson',
                            'foreignKey' => 'status_id',
                            'dependent' => false
            ),
            'Article' => array(
                            'className' => 'Article',
                            'foreignKey' => 'status_id',
                            'dependent' => false
            ),
            'Word' => array(
                            'className' => 'Word',
                            'foreignKey' => 'status_id',
                            'dependent' => false
            ),
            'BaseWord' => array(
                            'className' => 'BaseWord',
                            'foreignKey' => 'status_id',
                            'dependent' => false
            ),
            'GrammarRule' => array(
                            'className' => 'GrammarRule',
                            'foreignKey' => 'status_id',
                            'dependent' => false
            )
    );

}
?>
```

user.php

```
<?php

/**
 * @property Group $Group
 * @property User $User
 */
class User extends AppModel {

    var $name = 'User';
    var $validate = array(
            'username' => array('notempty','isUnique','/[A-Za-z0-9_]+/'),
            'password' => array('notempty'),
            'group_id' => array('numeric'),
            'email' => array('email','notempty'),
            'full_name' => array('notempty'),
            'location' => array('notempty')
    );

    //The Associations below have been created with all possible keys, those that are not needed
can be removed
    var $belongsTo = array(
            'Group' => array(
                            'className' => 'Group',
                            'foreignKey' => 'group_id'
            )
    );
    var $hasMany = array(
            'MessageSent' => array(
                            'className' => 'ActionRequest',
                            'foreignKey' => 'sender_id'
            ),
            'MessageReceived' => array(
                            'className' => 'ActionRequest',
                            'foreignKey' => 'recipient_id'
            ),
            'Article' => array(
                            'className' => 'Article',
                            'foreignKey' => 'user_id'
            ),
            'Animation' => array(
                            'className' => 'BaseWord',
                            'foreignKey' => 'user_id'
            ),

    );
}
?>
```

word.php

```
<?php

/**
 * @property Word $Word
 * @property BaseWord $BaseWord
 * @property Status $Status
 */
class Word extends AppModel {

    var $name = 'Word';
    var $validate = array(
            'base_word_id' => array('numeric'),
            'name' => array('notempty')
    );

//The Associations below have been created with all possible keys, those that are not needed can
be removed
    var $belongsTo = array(
            'BaseWord' => array(
                            'className' => 'BaseWord',
                            'foreignKey' => 'base_word_id'
            ),

            'Status' => array(
                            'className' => 'Status',
                            'foreignKey' => 'status_id'
            )
    );

    function getWordListForLetter($letter = null) {
        return $this->find('all',
                array('conditions'=>array('Word.name REGEXP'=>'^'.$letter,
'Status.name'=>'accepted', 'BaseWord.status_id'=>2),
                'fields'=>array('Word.name'),
                'order'=>array('Word.name ASC')));
    }

    function getIdByName($aName = null) {
        return $this->find(array('Word.name'=>$aName),array('Word.id'));
    }

    function getIdByNameAndBase($aBase = null, $aName = null) {
        $find = $this->find('first',array('conditions'=>array('Word.base_word_id'=>$aBase,
'Word.name'=>$aName), 'fields'=>array('Word.id')));
        $result = $find['Word']['id'];
        if(isset($result)) return $result;
        else return false;
    }

    function approve($id = null) {
        $this->read(null, $id);
        return $this->saveField('status_id',2);
    }

    function cancel($id = null) {
        $this->read(null, $id);
        return $this->saveField('status_id',1);
    }

}
?>
```

LOADER.FLA

```
var movies:String = this.root.loaderInfo.parameters.movies;
trace("about to load");
var moviesArr:Array=movies.split('|');
var loader:Loader = new Loader();
var cueIndex:int=0;
trace(moviesArr.length);
var aMovieClip:MovieClip;
goButton.addEventListener(MouseEvent.CLICK, startPlaying);
haltButton.addEventListener(MouseEvent.CLICK, stopPlaying);
repeatButton.addEventListener(MouseEvent.CLICK, repeatPlaying);
configureListeners(loader.contentLoaderInfo);
addChildAt(loader,0);
trace(moviesArr[0]);
try {
       loader.load(new URLRequest(moviesArr[0]));
} catch (e:Error) {
       trace("Failed:",e);
}

function configureListeners(dispatcher:IEventDispatcher):void {
       dispatcher.addEventListener(Event.COMPLETE, completeHandler);
}

function completeHandler(e:Event):void {
       var movie:* =loader.content;
       aMovieClip=movie;
       trace(aMovieClip.totalFrames);
       aMovieClip.addEventListener(Event.ENTER_FRAME, checkFrame);
       if (cueIndex==0) {
              aMovieClip.stop();
              goButton.visible = true;
              haltButton.visible = false;
       }
       trace('playing '+moviesArr[cueIndex]);
}

function checkFrame(e:Event):void {
       trace("Current Frame: " + e.target.currentFrame);
```

```
        if(aMovieClip.currentFrame >= aMovieClip.totalFrames) {
                aMovieClip.removeEventListener(Event.ENTER_FRAME, checkFrame);
                nextAnim();
        }
}

function nextAnim() {
        loader.unload();
        cueIndex++;
        if (cueIndex<moviesArr.length) {
                loader.load(new URLRequest(moviesArr[cueIndex]));
        } else {
                cueIndex=0;
                loader.load(new URLRequest(moviesArr[0]));
        }
}

function startPlaying(e:Event):void {
        aMovieClip.play();
        goButton.visible = false;
        haltButton.visible = true;
}

function stopPlaying(e:Event):void {
        aMovieClip.stop();
        goButton.visible = true;
        haltButton.visible = false;
}

function repeatPlaying(e:Event):void {
        aMovieClip.removeEventListener(Event.ENTER_FRAME, checkFrame);
        loader.unload();
        cueIndex = 0;
        loader.load(new URLRequest(moviesArr[0]));
}
```

VIEWS

Action_requests

Compose_message.ctp

```
<?=$javascript->link('tiny_mce/tiny_mce');?>
<script type="text/javascript"
src="/vocalhands/js/tiny_mce/plugins/tinybrowser/tb_tinymce.js.php"></script>
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$html->link('<span>Inbox</span>',array('action'=>'viewInbox'),array(),null,false);?> |
<?=$html->link('<span>Sent Items</span>',array('action'=>'viewSent'),array(),null,false);?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Compose Message </h2>
</div>

<?=$form->create('ActionRequest', array('action' => 'composeMessage'));?>
<?=$tinyMce->init();?>
<table cellpadding="5">
    <fieldset>
        <tr><td><?=$form->label('Title')?></td><td><?=$form->text('title')?></td></tr>
        <tr>
            <td><?=$form->label('Recipient')?></td>
            <td><?=$form->input('recipient_id', array('label'=>false,
'selected'=>$selected))?></td>
        </tr>
        <tr><td><?=$form->label('Message')?></td><td><?=$form-
>textarea('message',array('rows'=>20,'cols'=>40))?></td></tr>
    </fieldset>
    <tr><td colspan="2" align="center"><?=$form->end('Submit');?></td></tr>
</table>
```

```
<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Index.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

Communicate with team members with the messaging system

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('messagemenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

Read_message.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Read Message </h2>
</div>

<table cellspacing="10">

<tr><td><strong>From:</strong></td><td><?=$message['Sender']['full_name']?>&nbsp;&lt;<?=$message[
'Sender']['email']?>&gt;</td></tr>
    <tr><td><strong>Title:</strong></td><td><?=$message['ActionRequest']['title']?></td></tr>
    <tr><td><strong>Date/Time
Sent:</strong></td><td><?=$message['ActionRequest']['datetime']?></td></tr>
    <tr><td><strong>Message:</strong></td><td><?=$message['ActionRequest']['message']?></td></tr>
    <tr><td colspan="2" align="right"><?=$html->link('Reply',
array('action'=>'composeMessage',$message['Sender']['id']))?></td></tr>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('messagemenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

View_inbox.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Inbox </h2>
</div>

<table cellspacing="20">

    <tr>
        <th><?=$paginator->sort('sender_id')?></th>
        <th><?=$paginator->sort('title')?></th>
        <th><?=$paginator->sort('datetime')?></th>
    </tr>
    <?php foreach($in_messages as $msg): ?>
    <tr>
        <td ><?=$msg['Sender']['full_name'];?></td>
        <td ><?=$html->link($msg['ActionRequest']['title'], array('action'=>'readMessage',
$msg['ActionRequest']['id']))?></td>
        <td ><?=$msg['ActionRequest']['datetime'];?></td>
    </tr>
    <?php endforeach; ?>

</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('messagemenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

View_sent.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Sent Items </h2>
</div>

<table cellspacing="20">

    <tr>
        <th><?=$paginator->sort('recipient_id')?></th>
        <th><?=$paginator->sort('title')?></th>
        <th><?=$paginator->sort('datetime')?></th>
    </tr>

    <?php foreach($out_messages as $msg): ?>
    <tr>
        <td ><?=$msg['Recipient']['full_name'];?></td>
        <td ><?=$html->link($msg['ActionRequest']['title'], array('action'=>'readMessage',
$msg['ActionRequest']['id']))?></td>
        <td ><?=$msg['ActionRequest']['datetime'];?></td>
    </tr>
    <?php endforeach; ?>

</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('messagemenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

Articles

```
<?=$javascript->link('tiny_mce/tiny_mce');?>
<script type="text/javascript"
src="/vocalhands/js/tiny_mce/plugins/tinybrowser/tb_tinymce.js.php"></script>
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs"><?=$html->link(__('List Articles', true), array('action' =>
'backend'));?></div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Add New Article </h2>
```

```
</div>
<?=$session->flash() ?>
<div class="articles form">
    <?=$form->create('Article');?>
    <table cellpadding="5">
        <fieldset>
            <?=$tinyMce->init();?>
            <tr><td><?=$form->label('Title')?></td><td><?=$form->text('title',
array('size'=>40))?></td></tr>
            <tr><td><?=$form->label('Content')?></td><td><?=$form-
>textarea('content',array('rows'=>20,'cols'=>80))?></td></tr>
        </fieldset>
        <tr><td colspan="2" rowspan="3" align="center"><?=$form->end('Submit');?></td></tr>
    </table>

</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Backend.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs"><?=$html->link(__('New Article', true), array('action' => 'add')); ?></div>
<?=$this->element('bodycontent.begin')?>
<div class="article-rel-wrapper">
    <h2 class="contentheading"> Articles </h2>
</div>

<div class="articles index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?></p>
    <table cellpadding="10" cellspacing="10">
        <tr>
            <th><?=$paginator->sort('Title','title');?></th>
            <th><?=$paginator->sort('Modified','date_time_modified');?></th>
            <th><?=$paginator->sort('Status','status_id');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php
        $i = 0;
        foreach ($articles as $article):
            $class = null;
            if ($i++ % 2 == 0) {
                $class = ' class="altrow"';
            }
            ?>
        <tr<?=$class;?>>
            <td><?=$article['Article']['title']; ?></td>
            <td><?=$article['Article']['date_time_modified']; ?></td>
            <td><?=$article['Status']['name']; ?></td>
            <td class="actions">
                    <?=$html->link(__('View', true), array('action' => 'view',
$article['Article']['id'])); ?>
                    <?=$html->link(__('Edit', true), array('action' => 'edit',
$article['Article']['id'])); ?>

            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="4" align="center">
                <table>
```

```
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Edit.ctp

```
<?=$javascript->link('tiny_mce/tiny_mce');?>
<script type="text/javascript"
src="/vocalhands/js/tiny_mce/plugins/tinybrowser/tb_tinymce.js.php"></script>
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs">
    <?=$html->link(__('List Articles', true), array('action' => 'backend'));?>
</div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Edit Article </h2>
</div>
<?=$session->flash() ?>
<div class="articles form">
    <?=$form->create('Article');?>
    <table cellpadding="5">
        <fieldset>
            <?=$tinyMce->init();?>
            <tr><td><?=$form->label('Title')?></td><td><?=$form->text('title',
array('size'=>40))?></td></tr>
            <tr><td><?=$form->label('Content')?></td><td><?=$form-
>textarea('content',array('rows'=>20,'cols'=>80))?></td></tr>
        </fieldset>
        <tr><td colspan="2" rowspan="3" align="center"><?=$form->end('Submit');?></td></tr>
    </table>

</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Index.ctp

```
<?=$this->element('style.single')?>
<?=$html->addCrumb('Articles',array('controller'=>'articles','action'=>'index'));?>
<?=$this->element('main.begin');?>
<?=$this->element('maincol.begin')?>
<?=$this->element('breadcrumbs.default')?>

<?php foreach($articles as $article): ?>

<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> <?=$article['Article']['title'];?> </h2>
</div>
<div class="article-info-surround">
    <div class="iteminfo">
        <div class="article-info-left">
            Modified&nbsp;
            <?=$article['Article']['date_time_modified']?>
            &nbsp;by&nbsp;
            <?=$article['Author']['full_name']?>
            <div class="clr"></div>
        </div>
    </div>
</div>
<?=str_replace("../../", "../", $article['Article']['content']);?>

<?=$this->element('bodycontent.end')?>
<br /><br />
<?php endforeach; ?>
<div class="paging">
    <table align="right">
    <tr><td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
    <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td></tr>
    </table>
</div>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

View.ctp

```
<?=$this->element('style.single')?>
<?=$html->addCrumb($article['Article']['title'],array('controller'=>'articles','action'=>'view',
Inflector::slug($article['Article']['title'])));?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('breadcrumbs.default')?>
<a href="javascript: history.go(-1)">Back</a>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> <?=$article['Article']['title'];?> </h2>
</div>
<div class="article-info-surround">
    <div class="iteminfo">
        <div class="article-info-left">
            <div class="clr"></div>
        </div>
    </div>
</div>
<?=$article['Article']['content'];?>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Asl_administrators

Asl_manager.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

Approve and cancel all ASL-related topics

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('aslmanagermenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

View_article_list.ctp

```
<?=$this->element('style.dual')?>
```

```
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Articles </h2>
</div>
<?=$session->flash()?>
<div class="articles index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?></p>
    <table cellpadding="10" cellspacing="10">
        <tr>
            <th><?=$paginator->sort('Title','title');?></th>
            <th><?=$paginator->sort('Author','user_id');?></th>
            <th><?=$paginator->sort('Modified','date_time_modified');?></th>
            <th><?=$paginator->sort('Status','status_id');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php
        $i = 0;
        foreach ($articles as $article):
            $class = null;
            if ($i++ % 2 == 0) {
                $class = ' class="altrow"';
            }
            ?>
        <tr<?=$class;?>>
            <td><?=$article['Article']['title']; ?></td>
            <td><?=$article['Author']['full_name']; ?></td>
            <td><?=$article['Article']['date_time_modified']; ?></td>
            <td><?=$article['Status']['name']; ?></td>
            <td class="actions">
                    <?=$html->link(__('View', true), array('controller'=>'articles', 'action' =>
'view', $article['Article']['id'])); ?>
                    <?php if($article['Article']['status_id'] != 2): ?>
                        <?=$html->link(__('Approve', true),
array('controller'=>'articles','action' => 'approve', $article['Article']['id'])); ?>
                    <?php endif; ?>
                    <?php if($article['Article']['status_id'] != 1): ?>
                        <?=$html->link(__('Cancel', true),
array('controller'=>'articles','action' => 'cancel', $article['Article']['id'])); ?>
                    <?php endif; ?>
                    <?=$html->link(__('Delete', true), array('controller'=>'articles','action' =>
'delete', $article['Article']['id']), null, sprintf(__('Are you sure you want to delete # %s?',
true), $article['Article']['id'])); ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="4" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
```

```
<?=$this->element('leftmenu.begin')?>
<?=$this->element('aslmanagermenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

View_base_word_list.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Animations </h2>
</div>

<div class="baseWords index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?></p>
    <table cellpadding="10" cellspacing="10">
        <tr>
            <th><?=$paginator->sort('name');?></th>
<th><?=$paginator->sort('meaning');?></th>
            <th><?=$paginator->sort('Developer','user_id');?></th>
            <th><?=$paginator->sort('status_id');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php
        $i = 0;
        foreach ($baseWords as $baseWord):
            $class = null;
            if ($i++ % 2 == 0) {
                $class = ' class="altrow"';
            }
            ?>
        <tr<?=$class;?>>
            <td>
                    <?=$baseWord['BaseWord']['name']; ?>
            </td>
            <td>
                    <?=$baseWord['BaseWord']['meaning']; ?>
            </td>
            <td>
                    <?php if($baseWord['BaseWord']['status_id']==4): ?>
                        <?=$baseWord['Developer']['full_name']; ?>
                    <?php endif; ?>
            </td>
            <td>
                    <?=$baseWord['Status']['name']; ?>
            </td>
            <td class="actions">
                    <?=$html->link(__('View', true), array('controller'=>'words', 'action' =>
'view', Inflector::slug($baseWord['BaseWord']['name']))); ?>
                    <?php if($baseWord['BaseWord']['status_id'] == 4): ?>
                        <?=$html->link(__('Free', true), array('controller'=>'baseWords','action'
=> 'open', $baseWord['BaseWord']['id'])); ?>
                    <?php endif; ?>
                    <?php if($baseWord['BaseWord']['status_id'] == 1): ?>
                        <?=$html->link(__('Approve', true),
array('controller'=>'baseWords','action' => 'approve', $baseWord['BaseWord']['id'])); ?>
                    <?php endif; ?>
                    <?php if($baseWord['BaseWord']['status_id'] == 2): ?>
                        <?=$html->link(__('Cancel', true),
array('controller'=>'baseWords','action' => 'cancel', $baseWord['BaseWord']['id'])); ?>
                    <?php endif; ?>
                    <?=$html->link(__('Delete', true), array('controller'=>'baseWords','action'
=> 'delete', $baseWord['BaseWord']['id']), null, sprintf(__('Are you sure you want to delete #
%s?', true), $baseWord['BaseWord']['id'])); ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="4" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('aslmanagermenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

View_grammar_rule_list.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Grammar Rules </h2>
</div>
<?=$session->flash()?>
<div class="Grammar Rules index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?></p>
    <table cellpadding="10" cellspacing="10">
        <tr>
            <th><?=$paginator->sort('description');?></th>
            <th><?=$paginator->sort('english_rule');?></th>
            <th><?=$paginator->sort('asl_rule');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php foreach ($grammarRules as $grammarRule): ?>
        <tr>
            <td>
                    <?=$grammarRule['GrammarRule']['description']; ?>
            </td>
            <td>
                    <?=$grammarRule['GrammarRule']['english_rule']; ?>
            </td>
            <td>
                    <?=$grammarRule['GrammarRule']['asl_rule']; ?>
            </td>
            <td>
                    <?=$grammarRule['Status']['name']; ?>
            </td>
            <td class="actions">
                    <?php if($grammarRule['GrammarRule']['status_id'] != 2): ?>
```

```
                        <?=$html->link(__('Approve', true),
array('controller'=>'grammarRules','action' => 'approve', $grammarRule['GrammarRule']['id'])); ?>
                    <?php endif; ?>
                    <?php if($grammarRule['GrammarRule']['status_id'] != 1): ?>
                        <?=$html->link(__('Cancel', true),
array('controller'=>'grammarRules','action' => 'cancel', $grammarRule['GrammarRule']['id'])); ?>
                    <?php endif; ?>
                    <?=$html->link(__('Delete', true),
array('controller'=>'grammarRules','action' => 'delete', $grammarRule['GrammarRule']['id']),
null, sprintf(__('Are you sure you want to delete # %s?', true),
$grammarRule['GrammarRule']['id'])); ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="4" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('aslmanagermenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

View_lesson_list.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Lessons </h2>
</div>
<?=$session->flash()?>
<div class="lessons index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?></p>
    <table cellpadding="10" cellspacing="10">
        <tr>
            <th><?=$paginator->sort('title');?></th>
            <th><?=$paginator->sort('section_id');?></th>
            <th><?=$paginator->sort('status_id');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php
        $i = 0;
        foreach ($lessons as $lesson):
            $class = null;
            if ($i++ % 2 == 0) {
                $class = ' class="altrow"';
            }
            ?>
        <tr<?=$class;?>>
```

```
            <td>
                    <?=$lesson['Lesson']['title']; ?>
            </td>
            <td>
                    <?=$lesson['Section']['name']; ?>
            </td>
            <td>
                    <?=$lesson['Status']['name']; ?>
            </td>
            <td class="actions">
                    <?=$html->link(__('View', true), array('controller'=>'lessons', 'action' =>
'view', $lesson['Lesson']['id'])); ?>
                    <?php if($lesson['Lesson']['status_id'] != 2): ?>
                        <?=$html->link(__('Approve', true),
array('controller'=>'lessons','action' => 'approve', $lesson['Lesson']['id'])); ?>
                    <?php endif; ?>
                    <?php if($lesson['Lesson']['status_id'] != 1): ?>
                        <?=$html->link(__('Cancel', true), array('controller'=>'lessons','action'
=> 'cancel', $lesson['Lesson']['id'])); ?>
                    <?php endif; ?>
                    <?=$html->link(__('Delete', true), array('controller'=>'lessons','action' =>
'delete', $lesson['Lesson']['id']), null, sprintf(__('Are you sure you want to delete # %s?',
true), $lesson['Lesson']['id'])); ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="4" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('aslmanagermenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
<?php
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
*/

?>
```

View_word_list.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Complex Words </h2>
</div>

<div class="words index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?></p>
    <table cellpadding="10" cellspacing="10">
        <tr>
            <th><?=$paginator->sort('Word','name');?></th>
            <th><?=$paginator->sort('Maps to','base_word_id');?></th>
            <th><?=$paginator->sort('status_id');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php
        $i = 0;
        foreach ($words as $word):
            $class = null;
            if ($i++ % 2 == 0) {
                $class = ' class="altrow"';
            }
            ?>
        <tr<?=$class;?>>
            <td>
                    <?=$word['Word']['name']; ?>
            </td>
            <td>
                    <?=$word['BaseWord']['name']; ?>
            </td>
            <td>
                    <?=$word['Status']['name']; ?>
            </td>
            <td class="actions">
                    <?php if($word['Word']['status_id'] != 2): ?>
                        <?=$html->link(__('Approve', true), array('controller'=>'words','action'
=> 'approve', $word['Word']['id'])); ?>
                    <?php endif; ?>
                    <?php if($word['Word']['status_id'] != 1): ?>
                        <?=$html->link(__('Cancel', true), array('controller'=>'words','action'
=> 'cancel', $word['Word']['id'])); ?>
                    <?php endif; ?>
                    <?=$html->link(__('Delete', true), array('controller'=>'words','action' =>
'delete', $word['Word']['id']), null, sprintf(__('Are you sure you want to delete # %s?', true),
$word['Word']['id'])); ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="4" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('aslmanagermenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

Base_words

Add.ctp

```
<script type="text/javascript"
src="/vocalhands/js/tiny_mce/plugins/tinybrowser/tb_standalone.js.php"></script>
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs">
    <?=$html->link(__('List Basic Words', true), array('action' => 'backend'));?>
</div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Add New Basic Word </h2>
</div>

<?=$form->create('BaseWord');?>
<table cellpadding="5">
    <fieldset>
        <tr><td><?=$form->label('Name');?></td><td colspan="2"><?=$form-
>text('name');?></td></tr>
        <tr><td><?=$form->label('Meaning');?></td><td colspan="2"><?=$form->text('meaning',
array('size'=>50));?></td></tr>
        <tr><td><?=$form->label('Part of Speech');?></td><td colspan="2"><?=$form-
>text('part_of_speech', array('size'=>20));?></td></tr>
        <tr><td><?=$form->label('Movement Description');?></td><td colspan="2"><?=$form-
>textarea('movement_description');?></td></tr>
        <tr>
            <td><?=$form->label('Image');?></td>
            <td><?=$form->text('image_link');?>
                <input name="Browse" value="Browse" type="button"
onclick="tinyBrowserPopUp('image','BaseWordImageLink')" /></td>
        </tr>
        <tr><td><?=$form->label('Section');?></td><td colspan="2"><?=$form->input('section_id',
array('label'=>false));?></td></tr>
        <?=$form->input('status_id',array('type'=>'hidden', 'value'=>3));?>
    </fieldset>
    <tr><td colspan="3" align="center"><?=$form->end('Submit');?><td></tr>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Backend.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs"><?=$html->link(__('New Basic Word', true), array('action' => 'add'));
?></div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Basic Words </h2>
</div>

<div class="baseWords index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?></p>
    <table cellpadding="10" cellspacing="10">
        <tr>
            <th><?=$paginator->sort('name');?></th>
            <th><?=$paginator->sort('meaning');?></th>
            <th><?=$paginator->sort('part_of_speech');?></th>
            <th><?=$paginator->sort('section_id');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php
        $i = 0;
        foreach ($baseWords as $baseWord):
            $class = null;
            if ($i++ % 2 == 0) {
                $class = ' class="altrow"';
            }
            ?>
        <tr<?=$class;?>>
            <td>
                    <?=$baseWord['BaseWord']['name']; ?>
            </td>
            <td>
                    <?=$baseWord['BaseWord']['meaning']; ?>
            </td>
            <td>
                    <?=$baseWord['BaseWord']['part_of_speech']; ?>
            </td>
            <td>
                    <?=$baseWord['Status']['name']; ?>
            </td>
            <td class="actions">
                    <?=$html->link(__('Edit', true), array('action' => 'edit',
$baseWord['BaseWord']['id'])); ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="4" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>


<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('wordmanagermenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

Edit.ctp

```
<script type="text/javascript"
src="/vocalhands/js/tiny_mce/plugins/tinybrowser/tb_standalone.js.php"></script>
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs"><?=$html->link(__('List Basic Words', true), array('action' =>
'backend'));?></div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Edit Basic Word </h2>
</div>

<?=$form->create('BaseWord');?>
<table cellpadding="5">
    <fieldset>
        <tr><td><?=$form->label('Name');?></td><td colspan="2"><?=$form-
>text('name');?></td></tr>
```

```
        <tr><td><?=$form->label('Meaning');?></td><td colspan="2"><?=$form->text('meaning',
array('size'=>50));?></td></tr>
        <tr><td><?=$form->label('Part of Speech');?></td><td colspan="2"><?=$form-
>text('part_of_speech', array('size'=>20));?></td></tr>
        <tr><td><?=$form->label('Movement Description');?></td><td colspan="2"><?=$form-
>textarea('movement_description',array('cols'=>50));?></td></tr>
        <tr>
            <td><?=$form->label('Image');?></td>
            <td><?=$form->text('image_link');?>
                <input name="Browse" value="Browse" type="button"
onclick="tinyBrowserPopUp('image','BaseWordImageLink')" /></td>
        </tr>
        <tr><td><?=$form->label('Section');?></td><td colspan="2"><?=$form->input('section_id',
array('label'=>false));?></td></tr>
        <?=$form->input('status_id',array('type'=>'hidden', 'value'=>$status));?>
    </fieldset>
    <tr><td colspan="3" align="center"><?=$form->end('Submit');?><td></tr>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Edit_animation.ctp

```
<script type="text/javascript"
src="/vocalhands/js/tiny_mce/plugins/tinybrowser/tb_standalone.js.php"></script>
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs"><?=$html->link(__('Back to reserved words', true), array('action' =>
'reservedWords'));?></div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Upload Animation </h2>
</div>

<?=$form->create('BaseWord', array('action'=>'editAnimation'));?>
<table cellpadding="5">
    <fieldset>
        <tr><td><?=$form->label('Name');?></td><td colspan="2"><?=$form->text('name',
array('readonly'=>true));?></td></tr>
        <tr><td><?=$form->label('Meaning');?></td><td colspan="2"><?=$form->text('meaning',
array('size'=>50, 'readonly'=>true));?></td></tr>
        <tr><td><?=$form->label('Part of Speech');?></td><td colspan="2"><?=$form-
>text('part_of_speech', array('size'=>20, 'readonly'=>true));?></td></tr>
        <tr>
            <td><?=$form->label('Animation');?></td>
            <td><?=$form->text('animation_link');?></td>
            <td><input name="Browse" value="Browse" type="button"
onclick="tinyBrowserPopUp('media','BaseWordAnimationLink')" /></td>
        </tr>
        <?=$form->input('status_id',array('type'=>'hidden', 'value'=>1));?>
    </fieldset>
    <tr><td colspan="3" align="center"><?=$form->end('Submit');?><td></tr>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Open_words.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$session->flash()?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
```

```
    <h2 class="contentheading"> Free Words </h2>
</div>

<div class="baseWords index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?></p>
    <table cellpadding="10" cellspacing="10">
        <tr>
            <th><?=$paginator->sort('name');?></th>
            <th><?=$paginator->sort('meaning');?></th>
            <th><?=$paginator->sort('Part of Speech','part_of_speech');?></th>
            <th><?=$paginator->sort('section_id');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php
        $i = 0;
        foreach ($baseWords as $baseWord):
            $class = null;
            if ($i++ % 2 == 0) {
                $class = ' class="altrow"';
            }
            ?>
        <tr<?=$class;?>>
            <td><?=$baseWord['BaseWord']['name']; ?></td>
            <td><?=$baseWord['BaseWord']['meaning']; ?></td>
            <td><?=$baseWord['BaseWord']['part_of_speech']; ?></td>
            <td><?=$baseWord['Section']['name']; ?></td>
            <td class="actions">
                    <?=$html->link(__('Reserve', true), array('action' => 'reserve',
$baseWord['BaseWord']['id'])); ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="4" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Base_words.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Reserved Words </h2>
</div>

<div class="baseWords index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?></p>
    <table cellpadding="10" cellspacing="10">
        <tr>
            <th><?=$paginator->sort('name');?></th>
            <th><?=$paginator->sort('meaning');?></th>
            <th><?=$paginator->sort('Part of Speech','part_of_speech');?></th>
            <th><?=$paginator->sort('section_id');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php
        $i = 0;
        foreach ($baseWords as $baseWord):
            $class = null;
            if ($i++ % 2 == 0) {
                $class = ' class="altrow"';
            }
            ?>
        <tr<?=$class;?>>
            <td><?=$baseWord['BaseWord']['name']; ?></td>
            <td><?=$baseWord['BaseWord']['meaning']; ?></td>
            <td><?=$baseWord['BaseWord']['part_of_speech']; ?></td>
            <td><?=$baseWord['Section']['name']; ?></td>
            <td class="actions">
                    <?=$html->link(__('Upload', true), array('action' => 'editAnimation',
$baseWord['BaseWord']['id'])); ?>
                    <?=$html->link(__('Free', true), array('action' => 'open',
$baseWord['BaseWord']['id'])); ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="4" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Exam_properties

Set_properties.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Set Exam Properties </h2>
</div>

<div class="examProperties index">
    <table cellpadding="10" cellspacing="10">
        <?=$form->create(false,array('action'=>'setProperties')) ?>
        <?php foreach ($examProperties as $property): ?>
        <tr>
```

```
            <td><?=$property['ExamProperty']['description']?></td>
            <td class="actions">
                    <?php $url = array('action' =>'flickSwitch',
$property['ExamProperty']['id']); ?>
                    <?php if($property['ExamProperty']['enabled']): ?>
                        <?=$html->link(__('Enabled', true), $url); ?>
                    <?php else: ?>
                        <?=$html->link(__('Disabled', true),$url); ?>
                    <?php endif; ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr><td>Number of items per exam</td><td><?=$form-
>select('numItems',$optionsArray,$selected,null,false)?></td><td align="right"><?=$form-
>end('Submit')?></td></tr>
    </table>
</div>


<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Exams

Get_result.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Exam Result </h2>
</div>

<h4>You scored <strong><?=$score?>/<?=$total?></strong>. <?=$message?></h4>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Index.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Take Exam </h2>
</div>

<h4>Choose exam difficulty:</h4>

<ul>
<?php foreach($sections as $section):?>
    <?php if($section['size']>=3): ?>
    <li><?=$html->link($section['name'],array('action'=>'startExam',$section['id']))?></li>
    <?php endif; ?>
<?php endforeach; ?>
</ul>
<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

View_multiple_choice_question.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
```

```
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Question # <?=$session->read('Exam.index')?> </h2>
</div>

<?php $questions = $session->read('Exam.currentQuestion'); ?>

Which of these is the sign language for <strong><?=$questions[$session-
>read('Exam.correctAnswer')]['BaseWord']['name']?></strong>?

<table cellspacing="15">
    <?php $i = 0 ?>
    <?php foreach($questions as $option): ?>
    <tr>
        <td><?=$html->link(($i+1).')',array('action'=>'viewMultipleChoiceQuestion',$i))?></td>
        <td>
            <?=$flash-
>renderSwf('flash/loader.swf',450,320,false,array('flashvars'=>array('movies'=>$option['BaseWord'
]['animation_link'])))?>
        </td>
    </tr>
        <?php $i++; ?>
    <?php endforeach; ?>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

View_true_false_question.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Question # <?=$session->read('Exam.index')?> </h2>
</div>

<?php $questions = $session->read('Exam.currentQuestion'); ?>

<table cellspacing="15">
    <tr>
        <td align="center">
            <?=$flash-
>renderSwf('flash/loader.swf',450,320,false,array('flashvars'=>array('movies'=>$questions[0]['Bas
eWord']['animation_link'])))?>
        </td>
    </tr>
    <tr>
        <td align="center">
            This is sign language for
            <?php if($session->read('Exam.correctAnswer')): ?>
            <strong><?=$questions[0]['BaseWord']['name']?></strong>
            <?php else: ?>
            <strong><?=$questions[1]['BaseWord']['name']?></strong>
            <?php endif; ?>.
            True or False?
            <?=$html->link('True',array('action'=>'viewTrueFalseQuestion',1))?>
            <?=$html->link('False',array('action'=>'viewTrueFalseQuestion',0))?>
        </td>
    </tr>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Grammar_rules

Add.ctp

```
<script type="text/javascript">
    function bracketCount(id) {
        if(id==0) var text = document.getElementById('GrammarRuleEnglishRule').value;
        else var text = document.getElementById('GrammarRuleAslRule').value;
        var open = 0;
        var closed = 0;
        var i= 0;
        for(i=0;i<text.length;i++) {
            if(text.charAt(i)=='[') open++;
            if(text.charAt(i)==']') closed++;
        }
        document.getElementsByName('opencount')[id].value = open;
        document.getElementsByName('closedcount')[id].value = closed;
    }

</script>
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs"><?=$html->link(__('List Grammar Rules', true), array('action' =>
'index'));?></div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Add New Grammar Rule </h2>
</div>
<p><a href="http://nlp.stanford.edu:8080/parser/" target="_blank">Click here for sample
grammar</a></p>
<table>
    <?=$form->create('GrammarRule');?>
    <tr>
        <td><?=$form->label('Description');?></td>
        <td><?=$form->text('description', array('size'=>30));?></td>
    </tr>
    <tr>
        <td><?=$form->label('English Rule');?></td>
        <td><?=$form->text('english_rule', array('onkeyup'=>'bracketCount(0)'));?></td>
        <td colspan="2">
            Open brackets:
            <input type="text" class="readonly" name="opencount" size="2" readonly="readonly"
value="0" />
            Closed brackets:
            <input type="text" class="readonly" name="closedcount" size="2" readonly="readonly"
value="0" />
            </td>
    </tr>

    <tr>
        <td><?=$form->label('ASL Rule');?></td>
        <td><?=$form->text('asl_rule', array('onkeyup'=>'bracketCount(1)'));?></td>
        <td colspan="2">
            Open brackets:
            <input type="text" class="readonly" name="opencount" size="2" readonly="readonly"
value="0" />
            Closed brackets:
            <input type="text" class="readonly" name="closedcount" size="2" readonly="readonly"
value="0" />
        </td>
    </tr>

    <?=$form->input('status_id',array('type'=>'hidden', 'value'=>1));?>
    <tr><td colspan="4" align="center"><?=$form->end('Submit');?><td></tr>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Edit.ctp

```
<script type="text/javascript">
    function bracketCount(id) {
        if(id==0) var text = document.getElementById('GrammarRuleEnglishRule').value;
        else var text = document.getElementById('GrammarRuleAslRule').value;
        var open = 0;
        var closed = 0;
        var i= 0;
        for(i=0;i<text.length;i++) {
            if(text.charAt(i)=='[') open++;
            if(text.charAt(i)==']') closed++;
        }
        document.getElementsByName('opencount')[id].value = open;
        document.getElementsByName('closedcount')[id].value = closed;
    }

</script>
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs"><?=$html->link(__('List Grammar Rules', true), array('action' =>
'index'));?></div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Edit Grammar Rule </h2>
</div>
<p><a href="http://nlp.stanford.edu:8080/parser/" target="_blank">Click here for sample
grammar</a></p>
<table>
    <tr>
        <td>

            <?=$form->create('GrammarRule');?>
            <table cellpadding="5">
                <tr>
                    <td><?=$form->label('Description');?></td>
                    <td><?=$form->text('description', array('size'=>30));?></td>
                </tr>
                <tr>
                    <td><?=$form->label('English Rule');?></td>
                    <td>
                        <?=$form->text('english_rule', array('onkeyup'=>'bracketCount(0)'));?>
                    </td>
                    <td colspan="2">
                        Open brackets:
                        <input type="text" class="readonly" name="opencount" size="2"
readonly="readonly" value="0" />
                        Closed brackets:
                        <input type="text" class="readonly" name="closedcount" size="2"
readonly="readonly" value="0" />
                    </td>
                </tr>

                <tr>
                    <td><?=$form->label('ASL Rule');?></td>
                    <td><?=$form->text('asl_rule', array('onkeyup'=>'bracketCount(1)'));?></td>
                    <td colspan="2">
                        Open brackets:
                        <input type="text" class="readonly" name="opencount" size="2"
readonly="readonly" value="0" />
                        Closed brackets:
                        <input type="text" class="readonly" name="closedcount" size="2"
readonly="readonly" value="0" />
                    </td>
                </tr>

                <?=$form->input('status_id',array('type'=>'hidden', 'value'=>1));?>
                <tr><td colspan="4" align="center"><?=$form->end('Submit');?><td></tr>
            </table>
        </td>

    </tr>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Index.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs"><?=$html->link(__('New Grammar Rule', true), array('action' => 'add'));
?></div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Grammar Rules </h2>
</div>

<div class="grammarRules index">
    <p>
        Grammar Rules tell Vocalhands how to translate a group of english text to its ASL
equivalent.
        It allows you to rearrange words or to remove unneeded ones.
        A rule is defined as a group of <a
href="http://www.comp.leeds.ac.uk/ccalas/tagsets/brown.html">Brown Corpus Tags</a>
        in Labeled Bracket Notation. Remember to use square brackets.
    <table cellpadding="10" cellspacing="10">
        <tr>
            <th><?=$paginator->sort('description');?></th>
            <th><?=$paginator->sort('english_rule');?></th>
            <th><?=$paginator->sort('ASL Rule','asl_rule');?></th>
            <th><?=$paginator->sort('status_id');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php
        $i = 0;
        foreach ($grammarRules as $grammarRule): ?>
        <tr>
            <td>
                    <?=$grammarRule['GrammarRule']['description']; ?>
            </td>
            <td>
                    <?=$grammarRule['GrammarRule']['english_rule']; ?>
            </td>
            <td>
                    <?=$grammarRule['GrammarRule']['asl_rule']; ?>
            </td>
            <td>
                    <?=$grammarRule['Status']['name']; ?>
            </td>
            <td class="actions">
                    <?=$html->link(__('Edit', true), array('action' => 'edit',
$grammarRule['GrammarRule']['id'])); ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="4" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Lessons

Add.ctp

```
<?=$javascript->link('tiny_mce/tiny_mce');?>
<script type="text/javascript"
src="/vocalhands/js/tiny_mce/plugins/tinybrowser/tb_tinymce.js.php"></script>
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs">
    <?=$html->link(__('List Lessons', true), array('action' => 'backend'));?> |
    <?=$html->link(__('List Sections', true), array('controller' => 'sections', 'action' =>
'index')); ?> |
    <?=$html->link(__('New Section', true), array('controller' => 'sections', 'action' =>
'add')); ?>
</div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Add New Lesson </h2>
</div>
<?=$session->flash() ?>
<?=$tinyMce->init(array(

'external_link_list_url'=>'http://'.$_SERVER['SERVER_NAME'].DS.'vocalhands'.DS.'js'.DS.'wordlist.
js'
));?>
<div class="lessons form">
    <?=$form->create('Lesson');?>
    <?=$form->hidden('status_id', array('value'=>'1'));?>
    <table cellpadding="5">
        <?=$form->input('order',array('type'=>'hidden','value'=>$new_order));?>
        <tr><td><?=$form->label('Title')?></td><td><?=$form->text('title')?></td></tr>
        <tr><td><?=$form->label('Description')?></td><td><?=$form-
>text('description',array('size'=>50))?></td></tr>
        <tr><td><?=$form->label('Content')?></td><td><?=$form-
>textarea('content',array('rows'=>20,'cols'=>80))?></td></tr>
        <tr><td><?=$form->label('Section')?></td><td><?=$form->input('section_id',
array('label'=>false))?></td></tr>

        <tr><td colspan="2" align="center"><?=$form->end('Submit');?></td></tr>
    </table>

</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Backend.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs">
    <?=$html->link(__('New Lesson', true), array('action' => 'add')); ?> |
    <?=$html->link(__('List Sections', true), array('controller'=>'sections', 'action' =>
'index')); ?>
</div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
```

```
    <h2 class="contentheading"> Lessons </h2>
</div>

<div class="lessons index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?></p>
    <table cellpadding="10" cellspacing="10">
        <tr>
            <th><?=$paginator->sort('title');?></th>
            <th><?=$paginator->sort('order');?></th>
            <th><?=$paginator->sort('section_id');?></th>
            <th><?=$paginator->sort('status_id');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php foreach ($lessons as $lesson):?>
        <tr>
            <td><?=$lesson['Lesson']['title']; ?></td>
            <td>
                    <?=$lesson['Lesson']['order']; ?>
                    <?php if($lesson['Lesson']['order']!=$min): ?>
                        <?=$html->link($html->image("frontend/button-
up.png"),array('action'=>'raise_up', $lesson['Lesson']['id']),array('escape'=>false)); ?>
                    <?php endif; ?>
                    <?php if($lesson['Lesson']['order']!=$max): ?>
                        <?=$html->link($html->image("frontend/button-
down.png"),array('action'=>'lower_down', $lesson['Lesson']['id']),array('escape'=>false)); ?>
                    <?php endif; ?>
            </td>
            <td><?=$lesson['Section']['name']; ?></td>
            <td><?=$lesson['Status']['name']; ?></td>
            <td class="actions">
                    <?=$html->link(__('View', true), array('action' => 'view',
$lesson['Lesson']['id'])); ?>
                    <?=$html->link(__('Edit', true), array('action' => 'edit',
$lesson['Lesson']['id'])); ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="5" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Edit.ctp

```
<?=$javascript->link('tiny_mce/tiny_mce');?>
<script type="text/javascript"
src="/vocalhands/js/tiny_mce/plugins/tinybrowser/tb_tinymce.js.php"></script>
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs">
```

```
    <?=$html->link(__('List Lessons', true), array('action' => 'backend'));?> |
    <?=$html->link(__('List Sections', true), array('controller' => 'sections', 'action' =>
'index')); ?> |
    <?=$html->link(__('New Section', true), array('controller' => 'sections', 'action' =>
'add')); ?>
</div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Edit Lesson </h2>
</div>

<div class="lessons form">
    <?=$form->create('Lesson');?>
    <fieldset>
        <?=$tinyMce->init(array(

'external_link_list_url'=>'http://'.$_SERVER['SERVER_NAME'].DS.'vocalhands'.DS.'js'.DS.'wordlist.
js'
        ));?>
        <table cellpadding="5">
            <tr><td><?=$form->label('Title')?></td><td><?=$form->text('title')?></td></tr>
            <tr><td><?=$form->label('Description')?></td><td><?=$form-
>text('description',array('size'=>50))?></td></tr>
            <tr><td><?=$form->label('Content')?></td><td><?=$form-
>textarea('content',array('rows'=>20,'cols'=>80))?></td></tr>
            <tr><td><?=$form->label('Section')?></td><td><?=$form->input('section_id',
array('label'=>false))?></td></tr>
            <?=$form->hidden('status_id', array('value'=>'1'));?>
            <tr><td colspan="2" align="center"><?=$form->end('Submit');?></td></tr>
        </table>
    </fieldset>

</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Index.ctp

```
<?=$this->element('style.dual')?>
<?=$html->addCrumb('Tutorials',array('controller'=>'lessons','action'=>'index'));?>
<?=$this->element('main.begin');?>
<?=$this->element('maincol.begin')?>
<?=$this->element('breadcrumbs.default')?>
<?=$this->element('bodycontent.begin')?>

Welcome to the Tutorials page, where you will learn lots of stuff about ASL.

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('lessonmenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

View.ctp

```
<?=$this->element('style.dual')?>
<?=$html->addCrumb($lesson['Lesson']['title'],

array('controller'=>'lessons','action'=>'view/'.Inflector::slug($lesson['Lesson']['title'])));?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('breadcrumbs.default')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> <?=$lesson['Lesson']['title'];?> </h2>
</div>
```

```
<div class="article-info-surround">
    <div class="iteminfo">
        <div class="article-info-left">
            <?=$lesson['Lesson']['description']?>
            <div class="clr"></div>
        </div>
    </div>
</div>
<?=$lesson['Lesson']['content'];?>
<a href="javascript: history.go(-1)">Back</a>
<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('lessonmenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

Sections

Add.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs"><?=$html->link(__('List Sections', true), array('action' =>
'index'));?></div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Add New Section </h2>
</div>
<?=$form->create('Section');?>
<table cellspacing="5">
    <fieldset>
        <?=$form->input('order',array('type'=>'hidden','value'=>$new_order));?>
        <tr><td><?=$form->label('Name')?></td><td><?=$form->input('name',
array('label'=>false));?></td><td colspan="2"><?=$form->end('Submit');?></td></tr>
    </fieldset>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Edit.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs">
    <?=$html->link(__('Delete', true), array('action' => 'delete', $form->value('Section.id')),
null, sprintf(__('Are you sure you want to delete # %s?', true), $form->value('Section.id'))); ?>
|
    <?=$html->link(__('List Sections', true), array('action' => 'index'));?>
</div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Edit Section </h2>
</div>
<?=$form->create('Section');?>
<table cellspacing="5">
    <fieldset>
        <tr><td><?=$form->label('Name')?></td><td><?=$form->input('name',
array('label'=>false));?></td><td colspan="2"><?=$form->end('Submit');?></td></tr>
    </fieldset>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Index.ctp

```
<?=$javascript->link('tiny_mce/tiny_mce');?>
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs"><?=$html->link(__('New Section', true), array('action' => 'add')); ?></div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Sections </h2>
</div>

<div class="sections index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?>
    </p>
    <table cellpadding="10" cellspacing="1">
        <tr>
            <th><?=$paginator->sort('name');?></th>
            <th><?=$paginator->sort('order');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php foreach ($sections as $section): ?>
        <tr>
            <td><?=$section['Section']['name']; ?></td>
            <td>
                    <?=$section['Section']['order']; ?>
                    <?php if($section['Section']['order']!=$min): ?>
                        <?=$html->link($html->image("frontend/button-
up.png"),array('action'=>'raise_up', $section['Section']['id']),array('escape'=>false)); ?>
                    <?php endif; ?>
                    <?php if($section['Section']['order']!=$max): ?>
                        <?=$html->link($html->image("frontend/button-
down.png"),array('action'=>'lower_down', $section['Section']['id']),array('escape'=>false)); ?>
                    <?php endif; ?>
            </td>
            <td class="actions" align="center">
                    <?=$html->link(__('Edit', true), array('action' => 'edit',
$section['Section']['id'])); ?>
                    <?=$html->link(__('Delete', true), array('action' => 'delete',
$section['Section']['id']), null, sprintf(__('Are you sure you want to delete # %s?', true),
$section['Section']['id'])); ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="4" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Users

Add.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Create New User </h2>
</div>

<div class="users form">
    <?=$form->create('User');?>
    <table cellspacing="5">
        <fieldset>
            <tr><td><?=$form->label('Username')?></td><td><?=$form->input('username',
array('label'=>false));?></td></tr>
            <tr><td><?=$form->label('Password')?></td><td><?=$form->input('password',
array('label'=>false));?></td></tr>
            <tr><td><?=$form->label('Email Address')?></td><td><?=$form->input('email',
array('label'=>false));?></td></tr>
            <tr><td><?=$form->label('Group')?></td><td><?=$form->input('group_id',
array('label'=>false));?></td></tr>
            <tr><td><?=$form->label('Full Name')?></td><td><?=$form->input('full_name',
array('label'=>false));?></td></tr>
            <tr><td><?=$form->label('Location')?></td><td><?=$form->input('location',
array('label'=>false));?></td></tr>
        </fieldset>
        <tr><td colspan="2" align="center"><?=$form->end('Submit');?></td></tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Edit.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs">
    <?=$html->link(__('Delete', true), array('action' => 'delete', $form->value('User.id')),
null, sprintf(__('Are you sure you want to delete # %s?', true), $form->value('User.id'))); ?>
</div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Create New User </h2>
</div>

<div class="users form">
    <?=$form->create('User');?>
    <table cellspacing="5">
        <fieldset>
            <tr><td><?=$form->label('Username')?></td><td><?=$form->input('username',
array('label'=>false));?></td></tr>
            <tr><td><?=$form->label('Password')?></td><td><?=$form->input('password',
array('label'=>false, 'value'=>''));?></td></tr>
            <tr><td><?=$form->label('Email Address')?></td><td><?=$form->input('email',
array('label'=>false));?></td></tr>
            <tr><td><?=$form->label('Group')?></td><td><?=$form->input('group_id',
array('label'=>false));?></td></tr>
            <tr><td><?=$form->label('Full Name')?></td><td><?=$form->input('full_name',
array('label'=>false));?></td></tr>
            <tr><td><?=$form->label('Location')?></td><td><?=$form->input('location',
array('label'=>false));?></td></tr>
        </fieldset>
        <tr><td colspan="2" align="center"><?=$form->end('Submit');?></td></tr>
    </table>
</div>


<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Index.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs"><?=$html->link(__('New User', true), array('action' => 'add')); ?></div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Users </h2>
</div>

<?=$user?>
<div class="users index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?></p>
    <table cellpadding="5" cellspacing="5">
        <tr>
            <th><?=$paginator->sort('username');?></th>
            <th><?=$paginator->sort('email');?></th>
            <th><?=$paginator->sort('group_id');?></th>
            <th><?=$paginator->sort('full_name');?></th>
            <th><?=$paginator->sort('location');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php
        $i = 0;
        foreach ($users as $user):
            $class = null;
            if ($i++ % 2 == 0) {
                $class = ' class="altrow"';
            }
            ?>
        <tr<?=$class;?>>
            <td>
                    <?=$user['User']['username']; ?>
            </td>
            <td>
                    <?=$user['User']['email']; ?>
            </td>
            <td>
                    <?=$user['Group']['name']; ?>
            </td>
            <td>
                    <?=$user['User']['full_name']; ?>
            </td>
            <td>
                    <?=$user['User']['location']; ?>
            </td>
            <td class="actions">
                    <?=$html->link(__('Edit', true), array('action' => 'edit',
$user['User']['id'])); ?>
                    <?=$html->link(__('Delete', true), array('action' => 'delete',
$user['User']['id']), null, sprintf(__('Are you sure you want to delete # %s?', true),
$user['User']['id'])); ?>
            </td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="7" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Login.ctp

```
<?=$this->element('style.single');?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Administrators </h2>
</div>
<?php
if($session->check('Message.auth')) $session->flash('auth');
?>
<table cellpadding="3">
    <?=$form->create('User', array('action' => 'login'))?>
    <tr><td><?=$form->label('Username');?></td><td><?=$form->input('username',
array('label'=>false));?></td></tr>
    <tr><td><?=$form->label('Password');?></td><td><?=$form->input('password', array('type' =>
'password', 'label'=>false));?></td></tr>
    <tr><td colspan="2" rowspan="2" valign="bottom"><?=$form->end('Login');?></td></tr>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Words

Add.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs"><?=$html->link(__('List Words', true), array('action' =>
'backend'));?></div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Add New Word </h2>
</div>

<div class="words form">
    <?=$form->create('Word');?>
    <table cellspacing="5">
        <fieldset>
            <tr><td><?=$form->label('Name')?></td><td><?=$form->input('name',
array('label'=>false))?></td></tr>
            <tr><td><?=$form->label('Base Word')?></td><td><?=$form->input('base_word_id',
array('label'=>false))?></td></tr>
            <?=$form->input('status_id', array('type'=>'hidden', 'value'=>1))?>
        </fieldset>
        <tr><td colspan="2" align="center"><?=$form->end('Submit');?></td></tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Backend.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs">
    <?=$html->link(__('New Word', true), array('action' => 'add')); ?> |
    <?=$html->link(__('List Sections', true), array('controller'=>'sections','action' =>
'index')); ?>
</div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Complex Words </h2>
</div>

<div class="words index">
    <p>
        <?php
        echo $paginator->counter(array(
        'format' => __('Page %page% of %pages%, showing %current% records out of %count% total,
starting on record %start%, ending on %end%', true)
        ));
        ?></p>
    <table cellpadding="10" cellspacing="10">
        <tr>
            <th><?=$paginator->sort('Word','name');?></th>
            <th><?=$paginator->sort('Maps to','base_word_id');?></th>
            <th><?=$paginator->sort('status_id');?></th>
            <th class="actions"><?php __('Actions');?></th>
        </tr>
        <?php
        $i = 0;
        foreach ($words as $word):
            $class = null;
            if ($i++ % 2 == 0) {
                $class = ' class="altrow"';
            }
            ?>
        <tr<?=$class;?>>
            <td>
                    <?=$word['Word']['name']; ?>
            </td>
            <td>
                    <?=$word['BaseWord']['name']; ?>
            </td>
            <td>
                    <?=$word['Status']['name']; ?>
            </td>
            <td class="actions">
                    <?=$html->link(__('View', true), array('action' => 'view',
$word['Word']['id'])); ?>
                    <?=$html->link(__('Edit', true), array('action' => 'edit',
$word['Word']['id'])); ?></td>
        </tr>
        <?php endforeach; ?>
        <tr>
            <td colspan="4" align="center">
                <table>
                    <tr>
                        <td><?=$paginator->prev(__('previous', true), array(), null,
array('class'=>'disabled'));?></td>
                        <td><?=$paginator->next(__('next', true), array(), null, array('class' =>
'disabled'));?></td>
```

```
                    </tr>
                </table>
            </td>
        </tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('wordmanagermenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

Edit.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<div id="breadcrumbs">
    <?=$html->link(__('List Words', true), array('action' => 'backend'));?>
</div>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Edit Word </h2>
</div>

<div class="words form">
    <?=$form->create('Word');?>
    <table cellspacing="5">
        <fieldset>
            <tr><td><?=$form->label('Name')?></td><td><?=$form->input('name',
array('label'=>false));?><td></td></tr>
            <tr><td><?=$form->label('Basic Word')?></td><td><?=$form->input('base_word_id',
array('label'=>false));?></td></tr>
            <?=$form->input('status_id',array('type'=>'hidden','value'=>'1'))?>
        </fieldset>
        <tr><td colspan="2" align="center"><?=$form->end('Submit');?></td></tr>
    </table>
</div>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Index.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin');?>
<?=$this->element('maincol.begin');?>
<?=$this->element('breadcrumbs.letterlist')?>
<?=$this->element('bodycontent.begin')?>

Welcome to the ASL Dictionary.

<?=$this->element('bodycontent.end');?>
<?=$this->element('maincol.end');?>
<?=$this->element('leftmenu.begin')?>
<?=$this->element('wordmenu')?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end')?>
```

Output_result.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
```

```
    <h2 class="contentheading"> Translate </h2>
</div>

<table cellspacing="15" align="center">
    <tr><td>English:</td> <td><?=$toTranslate?></td></tr>
    <tr><td>ASL:</td> <td><?=$translated?></td></tr>
    <?php if(strlen($animation)>0): ?>
    <tr><td colspan="2" align="center"><?=$flash-
>renderSwf('flash/loader.swf',450,320,false,array('flashvars'=>array('movies'=>$animation)))?></t
d></tr>
    <tr>
        <td colspan="2" align="center">
            <strong>
                    <?php foreach($availableWords as $word): ?>
                        <?='['.$word['word'].']'?>
                    <?php endforeach; ?>
            </strong>
        </td>
    </tr>
    <?php endif; ?>
    <?php if(count($unavailableWords)>0): ?>
    <tr>
        <td colspan="2" align="center" class="translation_error">
            <p>The following words are unavailable for signing:</p>
                <?php $i=0; ?>
                <?php foreach($unavailableWords as $word): ?>
                    <?php if($i!=0): ?>,<?php endif;?>
                    <?=$word['word']?>
                    <?php $i++; ?>
                <?php endforeach; ?>
        </td>
    </tr>
    <?php endif; ?>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Search.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('breadcrumbs.letterlist')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Word Search </h2>
</div>
<?=$form->create('Word',array('action'=>'search'));?>
<table align="center">
    <tr><td>Look for: </td><td><?=$form->input('searchFor',
array('label'=>false))?></td><td><?=$form->end('Search');?></td></tr>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Translate.ctp

```
<?=$this->element('style.single')?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<div class="article-rel-wrapper">
    <h2 class="contentheading"> Translate </h2>
</div>
<table align="center">
```

```
    <?php if(!$isTranslating): ?>

        <?=$form->create('Word',array('action'=>'translate'));?>
    <tr><td>Enter a sentence that you want to translate. </td></tr>
    <tr><td><?=$form->input('translateWhat', array('type' => 'textArea',
'label'=>false));?></td></tr>
    <tr><td align="right"><?=$form->end('Translate');?></td></tr>

    <?php endif; ?>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

View.ctp

```
<?=$this->element('style.dual')?>
<?=$this->element('main.begin');?>
<?=$this->element('maincol.begin');?>
<?=$this->element('breadcrumbs.letterlist')?>
<?=$this->element('bodycontent.begin')?>
<div class="article-rel-wrapper">
    <h2 class="contentheading"> <?=$word['Word']['name'];?> </h2>
</div>
<table cellspacing="10" align="center">
    <?php $rowspan = 1 ?>
    <?php if($word['BaseWord']['name']!=$word['Word']['name']): ?>
    <?php $rowspan = 2 ?>
    <tr>
        <td align="right">Also the same as </td><td><?=$word['BaseWord']['name']?></td>
        <td align="center" rowspan="<?=$rowspan ?>" valign="center">
            <img src="<?=$word['BaseWord']['image_link'];?>" width="200"
alt="<?=$word['BaseWord']['image_link'];?>" />
        </td>
    </tr>
    <?php endif; ?>
    <tr>
        <td align="right">It means
</td><td><strong><?=$word['BaseWord']['meaning']?></strong></td>
    </tr>
    <tr>
        <td align="center" colspan="3">
            <?=$flash-
>renderSwf('flash/loader.swf',450,320,false,array('flashvars'=>array('movies'=>$word['BaseWord'][
'animation_link'])))?>
        </td>
    </tr>
    <tr>
        <td align="center" colspan="3">
            <?=$word['BaseWord']['movement_description']?>
        </td>
    </tr>
    <tr>
        <td align="left" colspan="2"><a href="javascript: history.go(-1)">Back</a></td>
    </tr>
</table>
<?=$this->element('bodycontent.end');?>
<?=$this->element('maincol.end');?>
<!--End Main Column (col1wrap)-->
<!--Begin Left Column (col2)-->
<?=$this->element('leftmenu.begin')?>
<?=$this->element('wordmenu');?>
<?=$this->element('leftmenu.end')?>
<?=$this->element('main.end');?>
```

Pages

Home.ctp

```
<?=$this->element('style.single');?>
```

```
<?=$html->addCrumb('Home','/vocalhands');?>
<?=$this->element('main.begin')?>
<?=$this->element('maincol.begin')?>
<?=$this->element('bodycontent.begin')?>

<table align="center">
    <tr><td><img src="img/Sign_Language_2_by_larah88.png" alt="sign language"/></td></tr>
    <tr><td align="right">by ~larah88</td></tr>
</table>

<?=$this->element('bodycontent.end')?>
<?=$this->element('maincol.end')?>
<?=$this->element('main.end')?>
```

Layouts

Default.ctp

```
<?=$html->docType('xhtml-strict');?>
<head>
    <?=$html->charset('UTF-8');?>
    <?=$html->meta('icon');?>
    <?=$html->css(array('frontend'));?>
    <?=$javascript->link(array('mootools','swfobject'))?>
    <title>Vocalhands</title>
    <style type="text/css">
        div.wrapper { margin: 0 auto; width: 982px;padding:0;}
        body { min-width:982px;}
        #inset-block-left { width:0px;padding:0;}
        #inset-block-right { width:0px;padding:0;}
        #maincontent-block { margin-right:0px;margin-left:0px;}

        .s-c-s .colmid { left:0px;}
        .s-c-s .colright { margin-left:-280px;}
        .s-c-s .col1pad { margin-left:280px;}
        .s-c-s .col2 { left:280px;width:0px;}
        .s-c-s .col3 { width:280px;}

        .s-c-x .colright { left:0px;}
        .s-c-x .col1wrap { right:0px;}
        .s-c-x .col1 { margin-left:0px;}
        .s-c-x .col2 { right:0px;width:0px;}

        .x-c-s .colright { margin-left:-280px;}
        .x-c-s .col1 { margin-left:280px;}
        .x-c-s .col3 { left:280px;width:280px;}
    </style>
    <script type="text/javascript">
        window.addEvent('load', function() {
            new Fusion('ul.menutop', {
                pill: 1,
                effect: 'slide and fade',
                opacity: 1,
                hideDelay: 500,
                tweakInitial: {'x': -12, 'y': 2},
                tweakSubsequent: {'x': -12, 'y': -14},
                menuFx: {duration: 400, transition: Fx.Transitions.Quint.easeOut},
                pillFx: {duration: 400, transition: Fx.Transitions.Quint.easeOut}
            });
        });
    </script>
</head>
<body id="ff-infuse" class="f-infuse style1 full infuse-home iehandle">
    <!--Begin Header-->
    <div id="header">
        <div class="wrapper">
            <div class="padding">
                <!--Begin Logo-->
                <div class="logo-module">
                    <a href="/vocalhands" id="logo"></a>
                </div>
                <!--End Logo-->
                <div id="top-right-surround">
                    <div id="top-right">
                        <div class="moduletable">
                            <ul class="menu-nav">
                                <?php
                                if($session->read('Auth.User') != null) echo '<li>'.$html-
>link('<span>Logout</span>', array('controller'=>'users', 'action'=>'logout'), null, null,
false).'</li>';
                                else echo '<li>'.$html->link('<span>Administrators</span>',
array('controller'=>'users','action'=>'login'), null, null, false).'</li>';
                                ?>
                                <li><?=$html->link('<span>About Vocalhands</span>',
array('controller'=>'articles','action'=>'view','about_vocalhands'), null, null, false)?></li>
                            </ul>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
    <!--End Header-->
    <div class="wrapper">
        <!--Begin Showcase-->
        <div class="show-tm">
            <div class="show-tl"></div>
            <div class="show-tr"></div>
        </div>
        <div class="show-m">
            <div class="show-l">
                <div class="show-r">
                    <!--Begin Horizontal Menu-->
                    <div id="horiz-menu" class="fusion" align="center">
                        <div class="wrapper" align="center">
                            <div class="padding" align="center">
                                <div id="horizmenu-surround" align="center">
                                    <!-- Main Menu goes here -->
                                    <?=$projectHtml->createMainMenu($session-
>read('Auth.User.group_id'))?>
                                </div>
                                <div class="clr"></div>
                            </div>
                        </div>
                    </div>
                    <!--End Horizontal Menu-->
                    <!--Begin Showcase Modules-->
                    <div id="showmodules" class="spacer w99"></div>
                    <!--End Showcase Modules-->
                </div>
            </div>
        </div>
        <div class="show-bm"><div class="show-bl"></div><div class="show-br"></div></div>
        <!--End Showcase-->
        <!--Begin Main Body-->

        <?=$content_for_layout; ?>

        <!--End Main Body-->
    </div>
    <!--Begin Footer-->
    <div id="footer-bg">
        <div class="wrapper">
            <div id="footer">
                <!--Begin Copyright Section-->
                <div class="copyright-block">
                    <div class="footer-div"></div>
                    <a href="http://www.rockettheme.com/" title="RocketTheme Joomla Templates
Club" id="rocket"></a>
                    <div id="copyright">
                        &copy; Copyright 2010, All Rights Reserved
                    </div>
                    <div id="top-button"><a href="#" id="top-scroll" class="top-button-desc">Back
to Top</a></div>
                </div>
                <!--End Copyright Section-->
            </div>
            <div id="footer-bg2"></div><div id="footer-bg3"></div>
        </div>
    </div>
    <!--End Footer-->
</body>
```

Helpers

Project_html_helper.php

```
<?php

/**
 * @property MainMenu $MainMenu
 */

class ProjectHtmlHelper extends HtmlHelper {

    function createMainMenu($aGroupId = null) {
        $homeClass = 'orphan item bullet';
        $itemClass = 'daddy item bullet subtext';

        //Identifies if consultant; 1 stands for consultant
        if($aGroupId == 1) $menuArr = $this->__buildConsultant($homeClass, $itemClass);
        //Identifies if consultant; 2 stands for flash developer
        else if($aGroupId == 2) $menuArr = $this->__buildFlashDeveloper($homeClass, $itemClass);
        //Identifies if ASL administrator; 3 stands for ASL administrator
        else if($aGroupId == 3) $menuArr = $this->__buildAslAdministrator($homeClass,
$itemClass);
        //Identifies if site administrator; 4 stands for site administrator
        else if($aGroupId == 4) $menuArr = $this->__buildSiteAdministrator($homeClass,
$itemClass);

        //Else, ordinary user
        else $menuArr = $this->__buildFrontEnd($homeClass, $itemClass);

        $returnTag = '<ul class="menutop level1">';
        $i = 0;
        foreach($menuArr as $menuItem) {
            if($i == 0)
                $returnTag .= '<li class="item1 active root">'.$menuItem.'</li>';
            else {
                $returnTag .= '<li class="item53 parent root">';
                $returnTag .= $menuItem;
                $returnTag .= '<div class="fusion-submenu-wrapper level2 columns2"><div
class="drop-top"></div></div>';
                $returnTag .= '</li>';
            }
            $i++;
        }
        $returnTag .= '</ul>';
        return $returnTag;
    }

    function __buildConsultant($homeClass = null, $itemClass = null) {
        $i = 0;
        $menuArr = array();
        $menuArr[$i] = $this->link('<span>Home</span>',
                array('controller'=>'pages', 'action'=>'display'),
                array('class'=>$homeClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>News & Articles<em>Customize Articles</em></span>',
                array('controller'=>'articles', 'action'=>'backend'),
                array('class'=>$itemClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Lessons<em>Customize Lessons</em></span>',
                array('controller'=>'lessons', 'action'=>'backend'),
                array('class'=>$itemClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Words<em>Manage Dictionary</em></span>',
                array('controller'=>'words', 'action'=>'backend'),
                array('class'=>$itemClass), null, false);
        $i++;
        $menuArr[$i] = $this->link('<span>Translator<em>Modify Grammar Rules</em></span>',
                array('controller'=>'grammar_rules', 'action'=>'index'),
                array('class'=>$itemClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Exams<em>Set Exam Options</em></span>',
                array('controller'=>'exam_properties', 'action'=>'setProperties'),
                array('class'=>$itemClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Messages<em>Inter-Communicate</em></span>',
                array('controller'=>'action_requests', 'action'=>'index'),
                array('class'=>$itemClass),null,false);
        $i++;
        return $menuArr;
    }

    function __buildFlashDeveloper($homeClass = null, $itemClass = null) {
        $i = 0;
        $menuArr = array();
        $menuArr[$i] = $this->link('<span>Home</span>',
                array('controller'=>'pages', 'action'=>'display'),
                array('class'=>$homeClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Free Words<em>Words with no developer</em></span>',
                array('controller'=>'baseWords', 'action'=>'openWords'),
                array('class'=>$itemClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Reserved Words<em>Words you reserved</em></span>',
                array('controller'=>'baseWords', 'action'=>'reservedWords'),
                array('class'=>$itemClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Messages<em>Inter-Communicate</em></span>',
                array('controller'=>'action_requests', 'action'=>'index'),
                array('class'=>$itemClass),null,false);
        $i++;
        return $menuArr;
    }

    function __buildAslAdministrator($homeClass = null, $itemClass = null) {
        $i = 0;
        $menuArr = array();
        $menuArr[$i] = $this->link('<span>Home</span>',
                array('controller'=>'pages', 'action'=>'display'),
                array('class'=>$homeClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Checklist<em>Approve ASL-Related Actions</em></span>',
                array('controller'=>'asl_administrators', 'action'=>'aslManager'),
                array('class'=>$itemClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Site Information<em>Customize About</em></span>',
                array('controller'=>'articles', 'action'=>'edit', 1),
                array('class'=>$itemClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Messages<em>Inter-Communicate</em></span>',
                array('controller'=>'action_requests', 'action'=>'index'),
                array('class'=>$itemClass),null,false);
        $i++;
        return $menuArr;
    }

    function __buildSiteAdministrator($homeClass = null, $itemClass = null) {
        $i = 0;
        $menuArr = array();
        $menuArr[$i] = $this->link('<span>Home</span>',
                array('controller'=>'pages', 'action'=>'display'),
                array('class'=>$homeClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Users<em>Manage Users</em></span>',
                array('controller'=>'users', 'action'=>'index'),
                array('class'=>$itemClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Messages<em>Inter-Communicate</em></span>',
                array('controller'=>'action_requests', 'action'=>'index'),
                array('class'=>$itemClass),null,false);
        $i++;
        return $menuArr;
    }

    function __buildFrontEnd($homeClass = null, $itemClass = null) {
        $i = 0;
        $menuArr = array();
        $menuArr[$i] = $this->link('<span>Home</span>',
                array('controller'=>'pages', 'action'=>'display'),
                array('class'=>$homeClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>News & Articles<em>Know What\'s Hot</em></span>',
                array('controller'=>'articles', 'action'=>'index'),
                array('class'=>$itemClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Tutorials<em>Take Lessons</em></span>',
                array('controller'=>'lessons', 'action'=>'index'),
                array('class'=>$itemClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Dictionary<em>Learn ASL Words</em></span>',
                array('controller'=>'words', 'action'=>'index'),
                array('class'=>$itemClass),null,false);
        $i++;
        $menuArr[$i] = $this->link('<span>Word Search<em>Look For Words</em></span>',
                array('controller'=>'words', 'action'=>'search'),
                array('class'=>$itemClass), null, false);
        $i++;
        $menuArr[$i] = $this->link('<span>Translate<em>Turn English to ASL</em></span>',
                array('controller'=>'words', 'action'=>'translate'),
                array('class'=>$itemClass), null, false);
        $i++;
        $menuArr[$i] = $this->link('<span>Exams<em>Assess Yourself</em></span>',
                array('controller'=>'exams', 'action'=>'index'),
                array('class'=>$itemClass), null, false);
        $i++;
        return $menuArr;
    }
}

?>
```

Tiny_mce_helper.php

```
<?php
class TinyMceHelper extends HtmlHelper {
    var $helpers = array('Javascript');
    var $defaults = array('mode'=>'textareas',
            'theme'=>'advanced',
            'skin'=>'o2k7',

'plugins'=>'tinybrowser,pagebreak,style,layer,table,save,advhr,advimage,advlink,emotions,iespell,
insertdatetime,preview,media,searchreplace,print,contextmenu,paste,directionality,fullscreen,none
ditable,visualchars,nonbreaking,xhtmlxtras,template,inlinepopups,autosave',
            // Theme options

'theme_advanced_buttons1'=>"save,newdocument,|,bold,italic,underline,strikethrough,|,justifyleft,
justifycenter,justifyright,justifyfull,styleselect,formatselect,fontselect,fontsizeselect",

'theme_advanced_buttons2'=>"cut,copy,paste,pastetext,pasteword,|,search,replace,|,bullist,numlist
,|,outdent,indent,blockquote,|,undo,redo,|,link,unlink,anchor,image,cleanup,help,code,|,insertdat
e,inserttime,preview,|,forecolor,backcolor",
'theme_advanced_buttons3'=>"tablecontrols,|,hr,removeformat,visualaid,|,sub,sup,|,charmap,emotion
s,iespell,media,advhr,|,print,|,ltr,rtl,|,fullscreen",

'theme_advanced_buttons4'=>"insertlayer,moveforward,movebackward,absolute,|,styleprops,|,cite,abb
r,acronym,del,ins,attribs,|,visualchars,nonbreaking,template,pagebreak,restoredraft",
            'theme_advanced_toolbar_location'=>"top",
            'theme_advanced_toolbar_align'=>"left",
            'theme_advanced_statusbar_location'=>"bottom",
            'theme_advanced_resizing'=>true,
            'content_css'=>'css/content.css',
            'file_browser_callback'=>'tinyBrowser',
    );

    function init($anOption=array()) {
        $options = array_merge($this->defaults, $anOption);
        $code = 'tinyMCE.init('.json_encode($options).');';
        return $this->Javascript->codeBlock($code);
    }
}
?>
```

Elements

Aslmanagermenu.ctp

```
<div class="side-style-h3">
    <h3 class="module-title">ASL Manager</h3>
</div>
<div class="module-inner">
    <ul class="menu level1" >
        <li class="item65 parent active" >
            <?=$html-
>link('<span>Articles</span>',array('action'=>'viewArticleList'),array(),null,false);?>
        </li>
        <li class="item65 parent active" >
            <?=$html-
>link('<span>Lessons</span>',array('action'=>'viewLessonList'),array(),null,false);?>
        </li>
        <li class="item65 parent active" >
            <?=$html->link('<span>Complex
Words</span>',array('action'=>'viewWordList'),array(),null,false);?>
        </li>
        <li class="item65 parent active" >
            <?=$html->link('<span>Simple
Words</span>',array('action'=>'viewBaseWordList'),array(),null,false);?>
        </li>
        <li class="item65 parent active" >
            <?=$html->link('<span>Grammar
Rules</span>',array('action'=>'viewGrammarRuleList'),array(),null,false);?>
        </li>
    </ul>
</div>
```

Bodycontentbegin.ctp

```
<div class="bodycontent">
    <div id="maincontent-block">
        <div class="">
            <div id="page" class="full-article">
                <div class="module-tm"><div class="module-tl"></div><div class="module-
tr"></div></div>
                <div class="module-inner">
```

Bodycontentend.ctp

```
                        <div class="article-ratings">
                        </div>
                    </div>
                <div class="module-bm"><div class="module-bl"></div><div class="module-
br"></div></div>
```

```
            </div>
        </div>
    </div>
</div>
```

Breadcrumbs.default.ctp

```
<div id="breadcrumbs">
    <?=$html->link('','/',array('id'=>'breadcrumbs-home'));?>
    <span class="breadcrumbs pathway">
        <?=$html->getCrumbs();?>
    </span>
</div>
```

Breadcrumbs.letterlist.ctp

```
<div id="breadcrumbs">
    <?=$html->link('','/',array('id'=>'breadcrumbs-home'));?>
    <span class="breadcrumbs pathway">
        <?php
        foreach($letterList as $letter) {
            if($letter['count']>0) {
                echo $html->link($letter['letter'],
                array('controller'=>'words','action'=>'goSearch'.'/'.$letter['letter']),
                array('class'=>'pathway'));
            }
            else { ?> <span class="no-link"><?=$letter['letter'];?></span> <?php }
        }
        ?>
    </span>
</div>
```

Leftmenu.begin.ctp

```
<div class="col2">
    <div id="leftcol">
        <div class="sidecol-tm"><div class="sidecol-tl"></div><div class="sidecol-
tr"></div></div>
        <div class="sidecol-m"><div class="sidecol-l"><div class="sidecol-r">
                    <div class="">
                        <div class="moduletable">
```

Leftmenu.end.ctp

```
                        </div>
                    </div>
                </div>
            </div>
        </div>
        <div class="sidecol-bm"><div class="sidecol-bl"></div><div class="sidecol-
br"></div></div>
    </div>
</div>
```

Lessonmenu.ctp

```
<div class="side-style-h3">
    <h3 class="module-title">Tutorials</h3>
</div>
<div class="module-inner">
    <ul class="menu level1" >
        <?php foreach($menu as $menuOption): ?>
            <?php if($menuOption['subOptions']): ?>
        <li class="lessonCategory" >
            <span><?=$menuOption['name'] ?></span>
        </li>
        <div class="fusion-submenu-wrapper level2 columns2">
            <div class="drop-top"></div>
            <ul class="lessonSubCategories">
                        <?php
                        foreach($menuOption['subOptions'] as $subOptions) {
                            if($subOptions['Lesson']['id']==$session-
>read('Lesson.activeLesson')) echo '<li class="item66 active">';
                            else echo '<li class="item66">';

                            echo $html->link('<span>'.$subOptions['Lesson']['title'].'</span>',
                            array('controller'=>'lessons', 'action'=>'view',
$subOptions['Lesson']['id']),
                            array('class'=>'orphan item bullet'),null,false);

                            echo '</li>';
                        }
                        ?>
            </ul>
        </div>
            <?php endif; ?>
        <?php endforeach; ?>
    </ul>
</div>
```

Main.begin.ctp

```
<div class="main-tm"><div class="main-tl"></div><div class="main-tr"></div></div>
<div class="main-m">
    <div class="main-l">
        <div class="main-r">
            <div id="main-body">
                <div id="main-content" class="s-c-x">
                    <div class="colmask leftmenu "><div class="wrapper">
                            <div class="colmid">
                                <div class="colright">
```

Main.end.ctp

```
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>
<div class="main-bm"><div class="main-bl"></div><div class="main-br"></div></div>
```

Maincol.begin.ctp

```
<div class="col1wrap">
    <div class="col1pad">
        <div class="col1">
            <div id="maincol">
```

Maincol.end.ctp

```
                <div class="clr"></div>
            </div>
        </div>
    </div>
</div>
```

Messagemenu.ctp

```
<div class="side-style-h3">
    <h3 class="module-title">Messages</h3>
</div>
<div class="module-inner">
    <ul class="menu level1" >
        <li class="item65 parent active" >
            <?=$html->link('<span>Compose
Message</span>',array('action'=>'composeMessage'),array(),null,false);?>
        </li>
        <li class="item65 parent active" >
            <?=$html-
>link('<span>Inbox</span>',array('action'=>'viewInbox'),array(),null,false);?>
        </li>
        <li class="item65 parent active" >
            <?=$html->link('<span>Sent
Items</span>',array('action'=>'viewSent'),array(),null,false);?>
        </li>
    </ul>
</div>
```

Style.dual.ctp

```
<style type="text/css">
    <!--

    div.wrapper { margin: 0 auto; width: 982px;padding:0;}
    body { min-width:982px;}
    #inset-block-left { width:0px;padding:0;}
    #inset-block-right { width:0px;padding:0;}
    #maincontent-block { margin-right:0px;margin-left:0px;}

    .s-c-s .colmid { left:280px;}
    .s-c-s .colright { margin-left:-280px;}
    .s-c-s .col1pad { margin-left:280px;}
    .s-c-s .col2 { left:0px;width:280px;}
    .s-c-s .col3 { width:0px;}

    .s-c-x .colright { left:280px;}
    .s-c-x .col1wrap { right:280px;}
    .s-c-x .col1 { margin-left:280px;}
    .s-c-x .col2 { right:280px;width:280px;}

    .x-c-s .colright { margin-left:-0px;}
    .x-c-s .col1 { margin-left:0px;}
    .x-c-s .col3 { left:0px;width:0px;}

    -->
</style>
```

Style.single.ctp

```
<style type="text/css">
    <!--

    div.wrapper { margin: 0 auto; width: 982px;padding:0;}
    body { min-width:982px;}
    #inset-block-left { width:0px;padding:0;}
    #inset-block-right { width:0px;padding:0;}
    #maincontent-block { margin-right:0px;margin-left:0px;}

    .s-c-s .colmid { left:200px;}
    .s-c-s .colright { margin-left:-200px;}
    .s-c-s .col1pad { margin-left:50px;}
    .s-c-s .col2 { left:0px;width:200px;}
    .s-c-s .col3 { width:0px;}

    .s-c-x .colright { left:50px;}
    .s-c-x .col1wrap { right:50px;}
    .s-c-x .col1 { margin-left:0px;}
    .s-c-x .col2 { right:280px;width:200px;}

    .x-c-s .colright { margin-left:-0px;}
    .x-c-s .col1 { margin-left:0px;}
    .x-c-s .col3 { left:0px;width:0px;}

    -->
</style>
```

Wordmanagermenu.ctp

```
<div class="side-style-h3">
    <h3 class="module-title">Word Manager</h3>
</div>
<div class="module-inner">
    <ul class="menu level1" >
        <li class="item65 parent active" >
            <?=$html->link('<span>Complex
Words</span>',array('controller'=>'words','action'=>'backend'),array(),null,false);?>
        </li>
        <li class="item65 parent active" >
            <?=$html->link('<span>Simple
Words</span>',array('controller'=>'base_words','action'=>'backend'),array(),null,false);?>
        </li>
    </ul>
</div>
<?=$form->create(false,array('action'=>'backend'))?>
<table align="center">
    <tr><td><?=$form->text('searchCriteria',array('size'=>15))?></td><td><?=$form-
>end('Search')?></td></tr>
</table>
```

Wordmenu.ctp

```
<div class="side-style-h3">
    <h3 class="module-title"><?=$currLetter?></h3>
</div>
<div class="module-inner">
    <ul class="menu level1" >
        <?php if(count($activeWords)==0): ?>
        <li class="item65 parent active">
            Search results returned none.
        </li>
        <?php endif;
        foreach($activeWords as $word): ?>
        <li class="item65 parent active">
                <?=$html->link('<span>'.$word['Word']['name'].'</span>',
                        array('controller'=>'words',
'action'=>'view/'.Inflector::slug($word['Word']['name'])),
                        array('class'=>'daddy item bullet subtext'),
                        null,false);?>
        </li>
        <?php endforeach; ?>
    </ul>
</div>
```

XIII. ACKNOWLEDGEMENTS

It has actually been around a year and four months since I have presented and defended this Special Problem in front of my mentors. After graduating, I earned a job at FBM, Aldrich Co's company and have been working there since. The tight work schedule has been a factor in delaying me from finishing this paper, along with a generous amount of my procrastination. But finally, I am now writing the acknowledgements section of my paper and I feel proud to get this stage.

Even though is really has been long while, conceptualizing and developing this special problem has been truly an unforgettable experience and I remember all the sleepless nights and the hard work I have put in this undertaking until now. I can also never forget a lot of things about my college life, which has been by far the most colorful part of my life. Lessons have been learned, about life and about the career I pursued. For all these things, there were countless people whom I have interacted with and I want to thank everyone, from a random RVC person who helped my enroll my subjects to by best friends and close mentors.

First of all, I thank the almighty God for His never-ending guidance and protection. When facing some rough times, just give Him a call and He will be right there beside you. I cannot count the times when I felt astray from reaching my aspirations and doing what I have to do. God eventually sets things right and you may not know it right away.

I thank my parents, my Mom and Pop who has always been there for me and supported me all the way. They always set me straight when I get bent down. I also thank my sister Abi who is always my comrade. And because I feel that I am his idol, I am inspired to be a good example to her. Hopefully I did so haha.

I thank my Alma Mater, UP Manila for basically letting me in here. I knew I was smart when I was young but I never expected to be a UP alumnus. You have taught most of the things I need to know about my career. I am really proud to have graduated here. I thank all of my esteemed mentors who made it an effort to provide us with high quality education.

I thank and salute my SP adviser, Sir Geoffrey Solano for all his support and effort in helping me finish this project. I thank him for not giving up on me when we were reaching stumbling blocks or when we were running out of ideas on his to make this project become a reality. I also thank him for allowing me be a part of his deaf class lessons once. It is really for them that I dedicate this system.

My college friends! Oh how I miss you guys. It has been more than a year and I haven’t seen a lot you even once! We should have a reunion of sort sometime. I thank you guys for letting me being part of the closely-knitted block we are. I believe our block is a rare gem in UP Manila because of how bonded we are. I consider you guys my family. I won’t forget all our Valentine’s and Christmas parties and how we ruled every DPSM event. With such overflowing talent, I think some of you should have taken the film industry and have been successful too. Good job everyone.

I thank the stat pips for all the group studies, overnights and laugh trips in and out of the classroom. I thank you for helping with stuff I don’t understand, most especially about the stat subjects. I believe all you guys will be successful in life because you are simply smart guys. Gessie, Izza, Mara, Joy, and Maita, the girls who are collectively known as 4f, thank you for being there for me every time. The sisters and didn’t have. May we see and laugh with each other again.

Do I really need to mention GS here? Jenzen, Siegfrid, AJ, Ronmark, Onil, Ahmad, GJ, Yanyan, Sam. The greatest peer group anyone can have. They are the perfect mix of dramatic, romantic and air headedness. I thank you guys for all the cries and laughs we shared and will continue to share. For carrying me up when I’m down, which is quite often I must say, and for accepting me regardless of whatever mess I have been in. I don’t need to flatter you much here. But you guys really deserve all the praise. See you guys later then.