

Software Design

Specification

**Terminology Creation**

*Thao Thai, Bikram Pokhrel, and Mary Hogan*

*Revision 1.0*

**Table of Contents**

1. Components…………………………………………………………………………………….3

1.1 Database..…………………………………………………………………………………3

1.2 Algorithm...…………….…………………………………………………………………3

1.3 Website…………….……….……………….…………………………………………….3

2. Third Party Technologies...…………………………………………………………………….3

3. Timeline………………………………………………………………………………………...4

**1. Components**

**1.1 Database**

We are using PHP to manipulate our SQL database. The database is a table that stores a list of technical terms in English and their translations. It uses the output of our algorithm to populate the table.

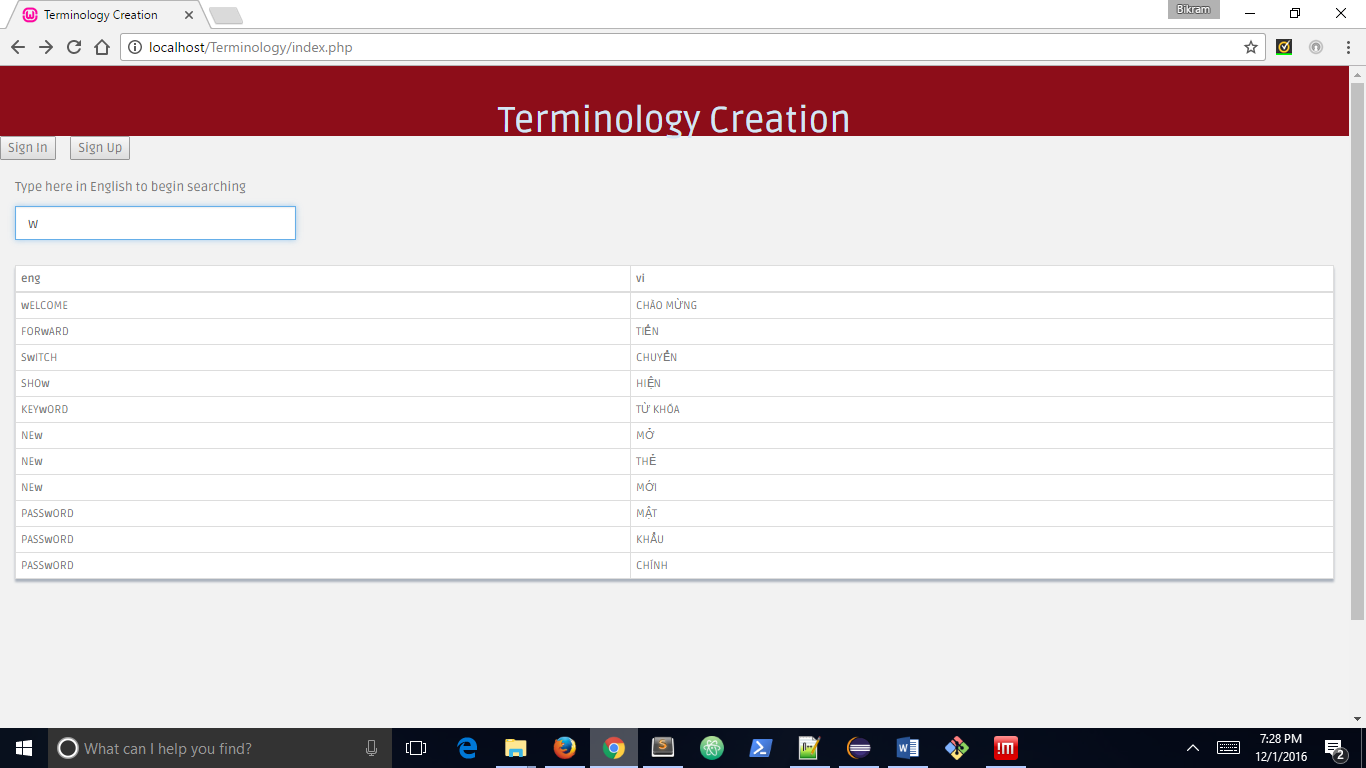
We will also be using a database to store the user/password information for our website.

**1.2 Algorithm**

Our algorithm is written in python. It takes as input language po files, and determines the probability that two terms are a translation of each other. The language files are a list of sentences in English and another language (i.e., “Go back one page,” “set desktop background”). We started by using PMI to calculate probability by determining how often two terms appear in the sentence translations together. However, this was not reliable and was very inaccurate for words that appear often (“the,” “and,” etc.). Hence, we are trying to incorporate Moses. Moses is a statistical machine translation software that is much more accurate than PMI. Also, we are working on ways to make PMI more reliable. For example, if a single word translation exists in the po file, we have a special case that recognizes this instead of using PMI on the term. The output of the algorithm is a file that lists the English term and the translation.

**1.2 Website**

We will be using javascript for our website. The user will select a language, and the website will display a list of the technical terms in English and the translations in the given language. If no translation exists, the space will be left blank. Additionally, users with accounts can login and update the database by editing/adding translations.



**2. Third Party Technologies**

We are using Moses and polib for our algorithm. As described above, Moses is a translation software that is significantly more accurate than PMI. Polib is a python library for editing and creating gettext files (i.e. po files). Lastly, we are using an open source language library that contains po files for 60+ languages.

**3. Timline**

By the end of this semester, we would either like to have Moses working or have a more accurate method of calculating probability. To do this, we can ignore common words that would result in a very high probability. Additionally, we can consider groups of more than one word because, for many languages, one English term may translate into a term of multiple words. We will also create our username/password database and a working front-end to our website.

By midterm of next semester, we would like to evaluate the accuracy of our algorithm by comparing it against a dictionary and to work on debugging and testing the application. Lastly, by the end of next semester, we want to have a fully functional, bug-free application.

The pair of words is added to the list as a translation pair

Open language po files

[Yes]

[No]

Create a list for English and translated terms

Calculate probability of English and translated words appearing in same sentence

[Check if PMI is >= 0.5]