# LinguisticDB Final Project Report

By Maya Messinger, Alex Angel, John Benhart, Ryan Piersma

## Description of the Application

LinguisticBD is a website that provides an extensive search interface for all 30,000+ books and 12,000+ authors of the Project Gutenberg database including a comparison algorithm that compares books based on linguistic features such as word sequences. Our website also supports the ability for users to create accounts, rate, and comment on books.

The website has numerous features. The homepage (the “Search” page) provides a simple search as well as an advanced search feature via the drop-down button with 11 different fields the user can specify. The user can click the chosen book title in the results list to be re-directed to the book page for that book. A Project Gutenberg link is also provided in the search results if the user wishes to read the book. (Note: Our website does not store the actual book content.)

The “About” page provides general information about our website, including the fact that we are not affiliated with the Project Gutenberg.

The “Statistics” page provides aggregate information for the user to better understand our dataset, like the numbers of books, users, and reviews on our website.

The “Sign In” page which lets the user create an account or sign in to an existing account. Being signed in is necessary to be able to comment or rate books on their respective book pages.

The “Profile” page lets the user change the account password, view the titles for the books that have been rated using that account and the corresponding rating, and view the titles for the books that have been commented on. The links in the “Profile” page will redirect the user to the book pages.

The individual book pages include summary information about each book, including the following:

* Title
* Book uid (Note: The uids on LinguisticDB and the uids on Project Gutenberg match, which makes it easy for the user to retrieve a book from the Project Gutenberg website by uid)
* Author name and birthyear
* “Date released” is the date the book was added to Project Gutenberg
* The word count for the book
* Average sentence length (in words)
* Average word length (in characters)
* Number of downloads on the Project Gutenberg website (rolling 30 days value)
* Average user rating on LinguisticDB
* Number of ratings on LinguisticDB
* “Rate this book” feature if the user is logged in
* Top 5 most common words in the book (words with length > 3)
* Top 5 most popular word sequences (words of all lengths)
* List of comments with username and timestamp
* Ability to comment if the user is logged in

On the right hand-side of the book page, the user will find the book titles for the five books that are most common to the book the user is viewing (using our cosine similarity algorithm).

## E/R Diagram

The E/R diagram is in a PDF file in the same folder as this report.

## Assumptions about the data

## Schema

The schema is as follows:

Authors (name VARCHAR(256) NOT NULL PRIMARY KEY,

birthdate INTEGER CHECK(birthdate < date\_part('year', current\_date)));

Books (uid INTEGER NOT NULL PRIMARY KEY,

title VARCHAR(512) NOT NULL,

date\_published VARCHAR(256) NOT NULL,

link\_to\_book VARCHAR(256) NOT NULL);

Writes (uid INTEGER NOT NULL REFERENCES Books(uid) PRIMARY KEY,

name VARCHAR(256) NOT NULL REFERENCES Authors(name));

BookWordAggregates (uid INTEGER NOT NULL REFERENCES Books(uid) PRIMARY KEY,

per\_sentence REAL NOT NULL,

total\_count REAL NOT NULL,

avg\_word\_length REAL NOT NULL);

CommonWords

(uid INTEGER NOT NULL REFERENCES Books(uid),

word VARCHAR(256) NOT NULL,

frequency INTEGER NOT NULL,

PRIMARY KEY(uid, word));

CREATE TABLE Downloads

(uid INTEGER NOT NULL REFERENCES Books(uid) PRIMARY KEY,

download INTEGER NOT NULL);

CREATE TABLE Sequences

(uid INTEGER NOT NULL REFERENCES Books(uid),

word VARCHAR(256) NOT NULL,

next\_word VARCHAR(256) NOT NULL,

times\_appear REAL NOT NULL,

PRIMARY KEY(uid, word, next\_word));

CREATE TABLE Users

(username VARCHAR(256) NOT NULL PRIMARY KEY,

email VARCHAR(256) NOT NULL,

password VARCHAR(256) NOT NULL);

CREATE TABLE UserRatings

(username VARCHAR(256) NOT NULL REFERENCES Users(username),

book\_id INTEGER NOT NULL REFERENCES Books(uid),

rating INTEGER NOT NULL, -- CHECK(rating > 0) AND (rating < 11),

timestamp TIMESTAMP NOT NULL,

PRIMARY KEY(username, book\_id));

CREATE TABLE UserReview

(username VARCHAR(256) NOT NULL REFERENCES Users(username),

book\_id INTEGER NOT NULL REFERENCES Books(uid),

review VARCHAR(256) NOT NULL,

timestamp TIMESTAMP NOT NULL,

PRIMARY KEY(username, book\_id));

CREATE TABLE CosineSimilarity

(uid1 INTEGER NOT NULL REFERENCES Books(uid),

uid2 INTEGER NOT NULL REFERENCES Books(uid),

cos\_similarity REAL NOT NULL,

rank INTEGER NOT NULL,

PRIMARY KEY(uid1, rank));

CREATE TABLE AuthorSimilarity

(author1 VARCHAR(256) NOT NULL REFERENCES Authors(name),

author2 VARCHAR(256) NOT NULL REFERENCES Authors(name),

cos\_similarity REAL NOT NULL,

rank INTEGER NOT NULL,

PRIMARY KEY(author1, rank));

## Indexes

## Implementation

Google Cloud

HTML

JavaScript

Vue