Programmer’s Manual

For

Language Learners United over 9000

Michael Cottrell

Riley Tucker

Project functions:

The major functions are as follows: A flash card game for the language of the user’s choice in which they can learn top words of a language. Ability for the user to find and message users to help facilitate learning. Ability for the user to see facts about the language of the user’s choice and countries, cultures, or the people who speak the language of choice.

**Table of Contents**

[1 Introduction 3](#_Toc58184396)

[1.1 Purpose 3](#_Toc58184397)

[2 Getting Started 3](#_Toc58184398)

[2.1 Source Code 3](#_Toc58184399)

[2.2 Installation (Windows Machine) 3](#_Toc58184400)

[2.3 Installation General 4](#_Toc58184401)

[3.1 Front-End Code 4](#_Toc58184402)

[3.2 Back-End Code 4](#_Toc58184403)

[3.3 Database Code 5](#_Toc58184404)

[3.1 Front-End Code 5](#_Toc58184405)

[3.1.1 Views 5](#_Toc58184406)

[3.1.2 Scripts 5](#_Toc58184407)

[3.2 Back-End Code 6](#_Toc58184408)

[3.1.1 App.js File 6](#_Toc58184409)

[3.1.2 Private Scripts 6](#_Toc58184410)

# 1 Introduction

## 1.1 Purpose

The purpose of this document is to explain the program code and programming practices of the Language Learners over 9000 (LLUO) web application. It will be used to aid developers in debugging and developing future features of the application.

1.1 Purpose

The purpose of this system is to help people learn languages, about the people, culture, and countries that speak the languages of interest, and to be able to communicate with and help other language learners.

# 2 Getting Started

## 2.1 Source Code

* + 1. The source code is available via a public repository [here](https://github.com/runksmania/LanguageLearnersUnitedOver9000).
       1. You can use git to download it into a directory of your choice on your machine.

## 2.2 Installation (Windows Machine)

* + 1. It is recommended that you do not have PostgreSQL already installed on your computer.
    2. Go [here](https://www.enterprisedb.com/postgresql-tutorial-resources-training?cid=437) and move the installer into the database subfolder of the project.
    3. Run setupDatabase.bat inside the Database subfolder of the project.
    4. Follow the instructions on the batch script.
    5. Go to step 6 of Installation General.

## 2.3 Installation General

* + 1. Grab PostgreSQL from [here](https://www.enterprisedb.com/downloads/postgres-postgresql-downloads) and install the latest version for your operating system.
    2. Once installed add psql to your bash profile or path environment variable.
    3. Create the file pgpass.conf in your home directory or %APPDATA%\Roaming\postgresql\ on a windows machine.
    4. In pgpass.conf enter the following line:
       1. localhost:5432:LLUO:postgres:[pwd]
       2. Where [pwd] is your password you used to setup PostgreSQL.
    5. Now run the resetDatabase.bat script in the database subfolder.
    6. Now to login to the database and to start the webserver use Database/loginToDatabase.bat and Server/startWebServer.bat

3 Code Organization

## 3.1 Front-End Code

1. CSS files will be located in Server/public/css.
2. Javascript files will be located in Server/public/scripts.
3. Any graphical assets will be in Server/public/graphics.
4. All .ejs views file will be located in Server/views.

## 3.2 Back-End Code

1. All Javascript classes and functions to help app.js run the server will be located in Server/private.
2. All logic for routing, requests, and sending of web application data will go through app.js.

## 3.3 Database Code

1. All database code will be located in Database/
2. All scripts for creating tables, triggers, and initial population of the database will be in Database/creationScripts

4 Code Overview

## 3.1 Front-End Code

### 3.1.1 Views

The code in this folder contains all ejs template files that make up the basic framework for the sitemap contained in the SDS.

### 3.1.2 Scripts

1. flashCardGame.js
   1. Logic for making the flashcard game work properly.
2. languageFacts.js
   1. Logic for ensuring Wikipedia links work properly.
3. loginFailed.js
   1. Logic for animating failed login attempts and if caps lock is on when typing a password.
4. logout.js
   1. Logic for ensuring logout links work properly.
5. resetPassword.js
   1. Logic for validating a password on registering for an account, and for resetting your password.
6. searchUsers.js
   1. Logic for requesting a new search of users from the webserver and updating the response.
7. webmail.js
   1. Logic for sending, writing, replying, updating inbox, and updating messages in order for the webmail to work.

## 3.2 Back-End Code

### 3.1.1 App.js File

This file is the backbone of the web server. It takes care of all routing between the user and the web application. It also logic that links the web application with the database.

### 3.1.2 Private Scripts

1. Constants.js
   1. Class to handle where the web server is being hosted.
2. DatabaseHandler.js
   1. Class for handling all database queries needed for the web application.
3. hash.js
   1. Functions to handle hashing and salting passwords for security reasons.
4. logger.js
   1. Configuration file for Logger class to handle logging of debug, standard, and error messages.
5. updateWordsList.js
   1. Function for handling the updating the words table of a language in the database.
6. User.js
   1. Class for easy access to persistent user information upon logging in.
7. wikipediaAPIRequest.js
   1. Function for handling requests to the Wikipedia API to get data for specific language.
8. wordsHttpRequest.js
   1. Function for parsing information on the website where the words for a language.