Advanced Databases Report

Radu Lefter

Heroku link:

Github link:

Introduction

My project is trying to simplify the way people, either linguists or language learners, can access and compare the first 200 common concepts across languages. The linguist Morris Swadesh came up with a list of basic concepts that can be found in every language. His work has been expanded through the years and there are many lists and collections of lists available now. I have taken a database of about 1200 languages and created an application that lets the user access any of these languages or compare them two by two. The inspiration came from my passion for learning languages. Although most lists are freely available online, applications that allow an easy visualisation are scarce, and even more so those that allow comparing them. Such an application is useful because it saves time, it allows collaborative work and it promotes language learning and understanding.

System overview

The application is a simple website that allows all the basic CRUD functions. The home page allows the user to see a selection of the first 12 languages in the database in alphabetical order. There is a simple navigation at the bottom that allows the user to browse through the languages in sets of 12. The user can search for a specific language from the search field in the header. That will display any languages that match the query . There is a static About page available that contains more information about the application and about the concept of a Swadesh List. From the navigation, the user can also access a Compare Languages page, where he can input two languages of his choice and see them side by side. The words should align perfectly, so they can be easily compared. The site also has basic user identification implemented. Any user can access the languages and compare them, but only registered and logged in users can edit or delete languages.

I used a database of 1200 languages provided for free by the [Wolfram Data Repository](https://datarepository.wolframcloud.com/resources/Swadesh-Lists). The database was in json format and it contained strings and arrays of strings.

Key design decisions

Front-end design

I decided to keep the front-end part very simple as the focus of the project was on the back-end. A lot of information is concentrated in the home page, where the user can see and browse through all the languages available. I wanted this to give a quick general idea of the number of languages and of the size of the data being used, as there are over a thousand languages to choose from. When comparing languages, they are displayed in a mirror image so that words as well as their English translation easily come into the field of view of the user. There is also an About page that thoroughly explains the concepts upon which the app is based.

The plan for the key views and interfaces can be seen here:

Graphical user interface

Description automatically generated

Fig 1 Swapp front page

Graphical user interface

Description automatically generated with low confidence

Fig 2 About page

Graphical user interface, application

Description automatically generated

Fig 3 Compare page

Graphical user interface, application

Description automatically generated

Fig 4 Login/Register page

Database design

The database is a NOSQL json file. It contains an array on which every object represents a language. The fields of each language object are the string “language” which contains the language name, the object “words” which contains the English words as keys and a string or an array of strings as the translations, the string “family” which contains the family of the language, the “iso693” string which contains the language code, the string “description” which contains a short description of the language, the string “region” which indicates the geographical location where the language is most widely spread, the string “status” which indicates if the language is alive, in danger or extinct, and the number “speakers” which indicates approximatively how many people still speak the language.

I had to heavily alter the procured database it in order to make it suitable for my purposes. Firstly, I added fields to capture details about the language, not just the words in the list. Secondly, the actual list of words for every language was deeply nested inside multiple arrays, so I extracted it and recreated it in a simple object. The keys on this object are the English words of a Swadesh list and the values are the translations in the target language. The word lists were not complete, and they were also not consistent, that is, some word lists had more keys than others. I created an array with all the words I wanted every word list to contain, and I transformed it into an object where the keys were the values in the array and all the new values were empty. I then merged every word list for every language in the database with this template object and replaced the word list with the newly created object. In this way, even if not all word lists have the complete number of values, all the keys are the same and are in the same order.

Security and scalability

The is basic security in place, as not anyone can modify the database. All users have access to the views and can view and compare languages at their pleasure. However, I implemented basic authentication so that, if users want to modify the languages they need to register first and then sign-in into their account. This will permit then not only to add new language and modify existing one, but also to delete any language in the database. The system could be made more secure so that only an admin can delete a language, but that regular users can only modify them.

The system is scalable as any number of languages can be added as well as any number of users. Whenever a new language is added, it will be available from the front page and will be searchable from the navigation bar. Also, the users can submit more than one translation for a word, as the key for every word in the database accepts an array. There is basic user-input implemented, so that only valid information can be used when registering and when creating a new language only strings up to a certain number of characters are accepted.

The front-end is decoupled from the back-end, following the rules of the MVC architecture, and that also contributes to security and scalability. The database can be manipulated independently and even completely replaced, only requiring a few tweaks in the views and controllers. The views can also be enhanced with better styling and slicker displays and it would still work with the current database.

Conclusion and reflection

This project was very easy to work with and a great learning experience. This was due to the excellent guidance offered and the very well-structured learning materials. The database I worked with was a fairly simple one, and yet it did provide me with challenges and opportunities to understand the importance of structuring data before using it in an application. Because my project was so small, I couldn’t use all the benefits of a NOSQL database, such as scalability, but I could, on the other hand appreciate how it was easier to design than an SQL database. I struggled a bit to update an object that contained other objects, but once I’ve done it a few times, the logic fell into place. The MVC pattern was already familiar to me, so I found it straight-forward to map the requests to the controller and then the controller to the model and the views. Error handling was also made easier by having the validation messages inside the model. I had some familiarity with the Express framework but none with Mongoose. Using both of them helped and sped up my work considerably and also provided me with a better understanding of the whole process of creating an app. The project can still be improved in many ways, such as allowing multiple languages to be compared at once, by implementing better security and a better design of the main interface, aspects that I am planning working on. All in all, this project made me a more confident developer and gave me the impetus to strive forwards in my computer science journey.