WordSet

A Dictionary and a Language Learning App with GPT API

Gebze Technical University - Computer Engineering Department

Undergraduate

Yusuf Arslan

https://wordset.netlify.app/

Abstract

WordSet is a breakthrough in the domain of computer engineering, revolutionizing the language learning adventure. Traditional methods of memorizing new words often lead to forgetfulness, overwhelming lists, and time-consuming study routines. This innovative web application simplifies the challenging process of learning a new language by addressing these issues. WordSet efficiently compiles words that the user asks the dictionary in everyday life and automatically transforms them into engaging study sets through the use of technology. By employing different and up-to-date technologies together, WordSet follows the user's learning stages with algorithms tailored to their progress. Its unique approach and technological integration mark a significant advancement in computer engineering, making language learning more efficient, enjoyable, and accessible for learners worldwide.

Introduction

WordSet is a remarkable web application offering a solution for individuals navigating the complexities of learning a new language. This introduction seeks to present WordSet's purpose within the realm of language acquisition and its notable significance in the domain of computer engineering.

Problem Statement:

The journey of acquiring new words in a foreign language often poses daunting challenges. Conventional methods, like using dictionaries or creating manual study sets, often lead to forgetfulness and overwhelm as word lists grow. WordSet emerges as a new technology that overcomes these challenges by automating the creation of study sets, thereby transforming the language learning experience.

The primary aim of WordSet is to alleviate the struggles encountered while learning new words by employing technology. By seamlessly curating word lists and crafting interactive study sets, the application aims to simplify vocabulary acquisition, making language learning an engaging and enjoyable pursuit.

Background

Language Learning Domain Overview:

Learning a new language presents inherent challenges, particularly in retaining and effectively utilizing new vocabulary. Traditional methods, such as dictionaries or manually created study sets, often fall short in ensuring long-term retention and user engagement. WordSet, however, stands out by introducing a dynamic and automated system that simplifies the process of mastering new words.

Existing Solutions and Limitations:

Traditional tools like dictionaries or static study sets (e.g. *Quizlet* is a one of the examples) have limitations in adapting to users' evolving needs. They often lack interactivity and fail to cater to diverse learning preferences. WordSet's introduction of automatic study set creation addresses these limitations by offering a user-friendly, interactive platform for language learners.

Relevance of WordSet:

WordSet emerges as a solution to bridge the gaps in existing language learning methodologies. By harnessing technology to create personalized study sets and interactive learning experiences, it presents a viable alternative to conventional methods. This dynamic approach not only simplifies vocabulary acquisition but also enhances user engagement and retention.

Methodology and Design

Development Tools and Technologies:

WordSet, as a web application, harnesses modern technologies to create a seamless and dynamic language learning experience. Below are the key technologies and tools used in its development:

Frontend Development

- **React:** Known for its dynamic and responsive user interfaces, React is utilized to build the frontend of WordSet. This framework enables the application to provide real-time updates and interactive features that enhance the user experience.
- **Ant Design:** To complement React's capabilities, WordSet incorporates the Ant Design system. Ant Design is a comprehensive design system that comes with a set of

high-quality React components out of the box. It is known for its enterprise-level products and allows for the creation of complex layouts and patterns with ease.

Backend Development

- **Apollo/GraphQL:** The backend leverages Apollo/GraphQL for efficient data management and retrieval. This combination allows for precise and fast queries, which is essential for the real-time functionality that WordSet offers.
- **PostgreSQL and Prisma:** For database management, WordSet uses PostgreSQL with Prisma. This setup ensures robust data integrity and reliability, which is crucial for maintaining the user's progress and study sets.

Integration with AI

- **GPT API:** To augment the user experience, WordSet integrates the GPT API for generating precise responses to dictionary queries. This AI-powered feature provides users with accurate and contextually relevant language information, aiding in their learning journey.

Deployment

For the deployment of WordSet, Netlify and Railway are used.

- **Netlify**, a cloud-based platform, is ideal for deploying the frontend of WordSet, a React application. It offers features like continuous deployment, global CDN, security, and instant rollbacks.
- **Railway**, a platform for deploying and managing backend services (PostgreSQL for our app), is used for the backend of WordSet. It simplifies deployment, provides scalability, manages different environments, and supports integrated services.

The Technology that Allows Me to Use All Technologies Together: RedwoodJS

RedwoodJS is a full-stack JavaScript framework that combines the best of React, GraphQL, Prisma, and more into a single toolchain. It's designed to help developers build scalable, maintainable web applications with less boilerplate code.

Software Design Principles and Architectural Patterns:

The application's design revolves around creating a user-centric experience. WordSet employs intuitive interfaces and interactive elements to facilitate effortless creation and utilization of study sets. The architecture emphasizes scalability and modularity, enabling seamless updates and additions to functionalities. Furthermore, the choice of React and Apollo/GraphQL allows for flexibility and adaptability, crucial in catering to diverse user needs.

Implementation and Application Development

Development Overview:

The development of WordSet involves the use of Git for version control, ensuring organized and systematic tracking of changes. GitHub serves as a repository, facilitating secure storage and collaboration. By creating issues on GitHub, the project's progress is effectively monitored, allowing for structured development.

Functionalities and Features:

WordSet integrates an intuitive feature wherein every dictionary query made by a user is automatically added to the user's WordSet. This aids in creating personalized word collections for further learning. Additionally, users possess the ability to organize these sets into folders, enabling seamless management and customization.

The application comes pre-equipped with essential word sets, including the Oxford 3000–5000-word lists, offering users readily available study materials. Users can clone these sets, working on them at their own pace and convenience.

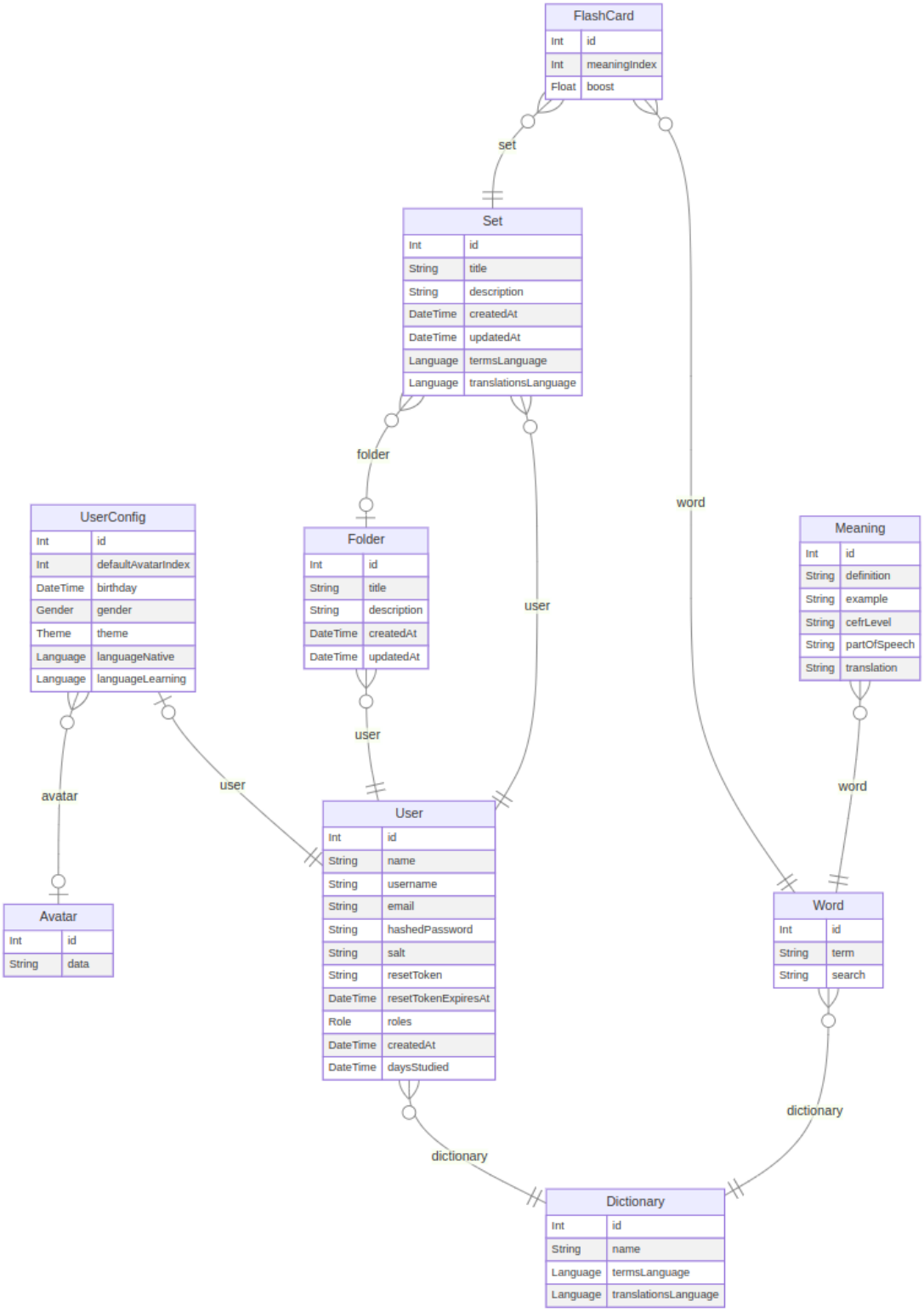
Moreover, users can explore, and access sets created by other users within the application. By cloning these sets, users can actively engage with diverse study materials shared by the community, fostering collaborative learning experiences.

Challenges Faced During Development:

Collecting and compiling ready-made word lists and converting them into sets using the GPT API will be one of the big challenges of this application. Other than that, since the application is still under development and I haven't encountered and solved any major problems yet, but in the normal course of writing code there have been/will continue to be some bugs.
It would be pointless to list them.

Class Diagrams:

The Class diagram you see below will give you an idea about the structure of the application.



Results and Findings

Development Stage:

As WordSet remains in the developmental phase, conclusive results and comprehensive findings regarding its performance and user feedback are yet to be obtained. Given its current status, the application's evaluation metrics, user response, and assessment criteria are pending, awaiting the completion of the development process and subsequent user interaction.

Discussion and Analysis

Comparison with Existing Solutions:

In comparison to traditional methods, WordSet showcases a distinct advantage by automating the generation of study sets and adapting to user queries in real-time. The application's dynamic nature and user-centric design set it apart from static, traditional study materials commonly used in language learning.

Application's Contributions and Strengths:

WordSet's strength lies in its integration of modern technologies to streamline language learning. The utilization of React, Apollo/GraphQL, and GPT API demonstrates its commitment to offering a responsive, interactive, and comprehensive learning platform.

Future Improvements and Enhancements:

To fortify WordSet's position as a robust language learning tool, continual enhancements are imperative. Future developments may include refining user interfaces for enhanced usability, expanding the repository of study sets, and conducting rigorous user testing to ensure optimal performance.

Conclusion

Recap of Objectives and Achievements:

In conclusion, WordSet, a novel web application designed to simplify language learning, stands poised to redefine the landscape of vocabulary acquisition. The application's primary goal of automating the creation of study sets and fostering an interactive learning environment has been a focal point throughout its development phase.

Significance in Computer Engineering:

Within the realm of computer engineering, WordSet represents a significant leap forward in leveraging technology to aid language learners. Its utilization of React, Apollo/GraphQL, and GPT API exemplifies its commitment to harnessing modern tools for enhancing the language learning experience.

Future Directions and Recommendations:

Looking ahead, WordSet's journey continues with a focus on further enhancements and refinements. Future endeavors can involve expanding its repository of study sets, refining user interfaces for enhanced accessibility.

Although it is beyond the scope of my current application, word groups can be clustered using Natural Language Processing (NLP) techniques during the grouping sets. Likewise, dictionaries can be used instead of the GPT API with web scrapping methods.

In summary, WordSet's development journey has highlighted its potential to simplify language learning through technology. When completed, it is a candidate to be an effective tool for those who are learning a new language or who want to improve themselves in a language.

References

1. React documentation. (n.d.). React - A JavaScript library for building user interfaces. Retrieved from https://reactjs.org/docs/getting-started.html
2. Ant Design documentation. (n.d.). Ant Design - The world's second most popular React UI framework. Retrieved from https://ant.design/docs/react/introduce
3. Apollo GraphQL documentation. (n.d.). Apollo GraphQL Docs. Retrieved from https://www.apollographql.com/docs/
4. PostgreSQL documentation. (n.d.). PostgreSQL: The world's most advanced opensource relational database. Retrieved from https://www.postgresql.org/docs/
5. Prisma documentation. (n.d.). Prisma - Next-generation Node.js and TypeScript ORM for Databases. Retrieved from https://www.prisma.io/docs/
6. OpenAI API documentation. (n.d.). OpenAI API. Retrieved from https://beta.openai.com/docs/
7. GitHub documentation. (n.d.). GitHub Docs. Retrieved from https://docs.github.com/en
8. Oxford University Press. (n.d.). Oxford 3000 and 5000. Retrieved from https://www.oxfordlearnersdictionaries.com/wordlist/english/oxford3000/
9. Quizlet. (n.d.). Quizlet - Learning tools & flashcards, for free. Retrieved from https://quizlet.com/
10. Netlify documentation. (n.d.) from https://docs.netlify.com/
11. Railway documentation. (n.d.) from https://docs.railway.app/

12. RedwoodJS documentation. (n.d.)

from https://redwoodjs.com/docs