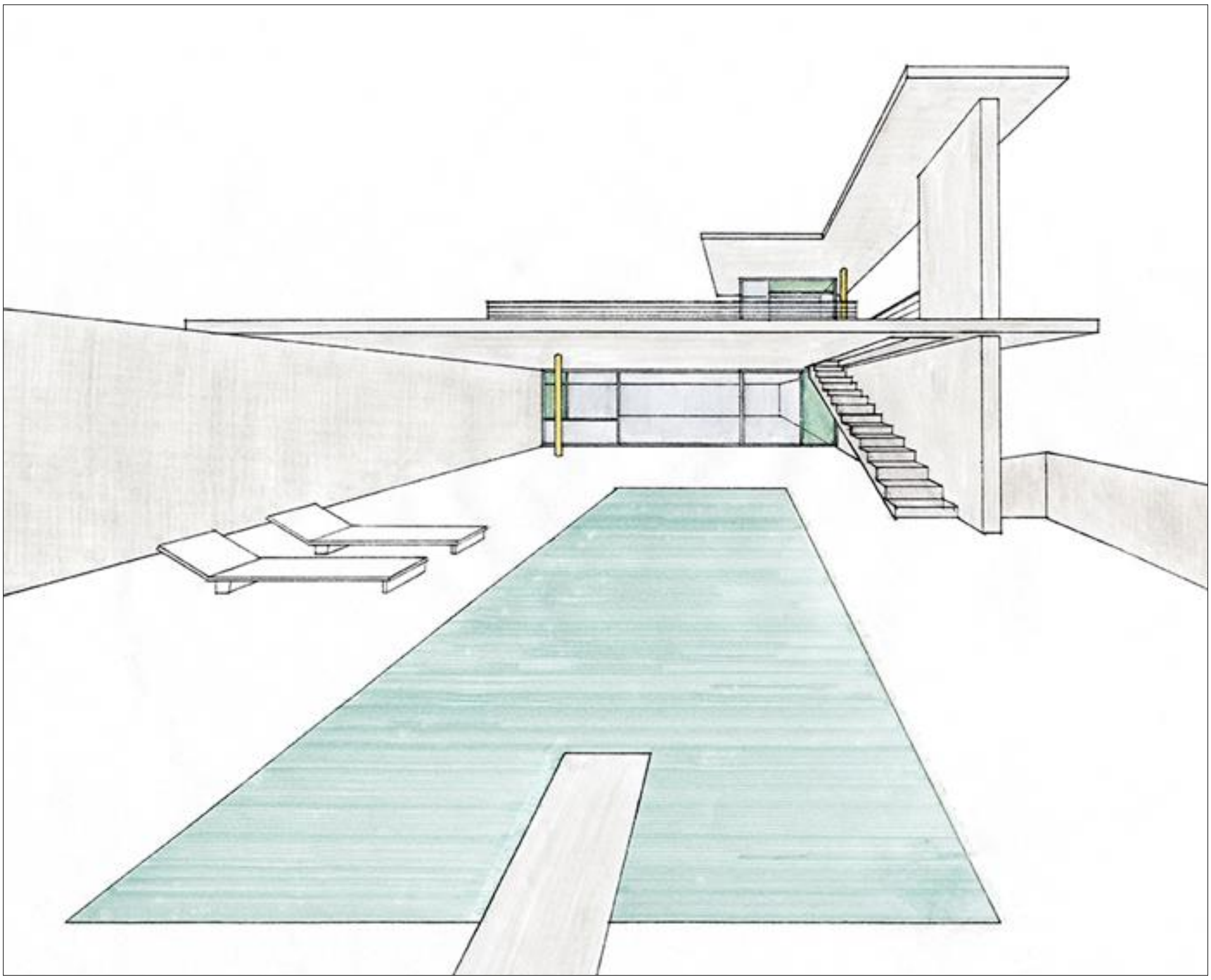


ABD Dictionary

A Language Learner-Oriented Project

BMIS Group 8

October 2023

CONTENTS

1. Summary of the business case

1.1 Introduction

Dictionaries have always been essential tools for language learners, particularly for Chinese students. However, there are often frustrating limitations when using traditional dictionaries and language software. Slow response times, excessive memory consumption, and the need for an internet connection can hinder the learning process. Moreover, the intrusion of advertisements and product promotions can disrupt students' concentration. To address these issues, we propose the development of an innovative and user-friendly dictionary software project that will revolutionize the way students and language learners' access and utilize linguistic resources. Our goal is to offer a seamless and efficient experience that empowers users to look up words, phrases, and translate sentences with ease and accuracy.

1.2 Background

During their English language courses, students frequently encounter unfamiliar and intricate words. While there are numerous dictionaries and software available to assist them, these resources have several significant drawbacks. They tend to consume a large amount of memory, resulting in sluggish response times. Additionally, they often require online queries, further delaying the word lookup process. Furthermore, these dictionaries and software are plagued by intrusive advertisements and irrelevant product promotions. This not only lengthens the software startup times but also disrupts the students' concentration and hampers their learning experiences. Consequently, students end up wasting a substantial amount of time navigating through the dictionary software instead of focusing on their language development.

1.3 Proposal

To optimize this situation, it is imperative to enhance existing dictionary software. This can be achieved by minimizing memory consumption, improving response times, and ensuring offline functionality for a seamless word lookup experience. Furthermore, removing irrelevant advertisements and unnecessary product promotions would alleviate distractions and streamline the learning process. By addressing these issues, students can maximize their learning potential and dedicate more time to mastering the English language, rather than being hindered by inefficient and disruptive dictionary software.

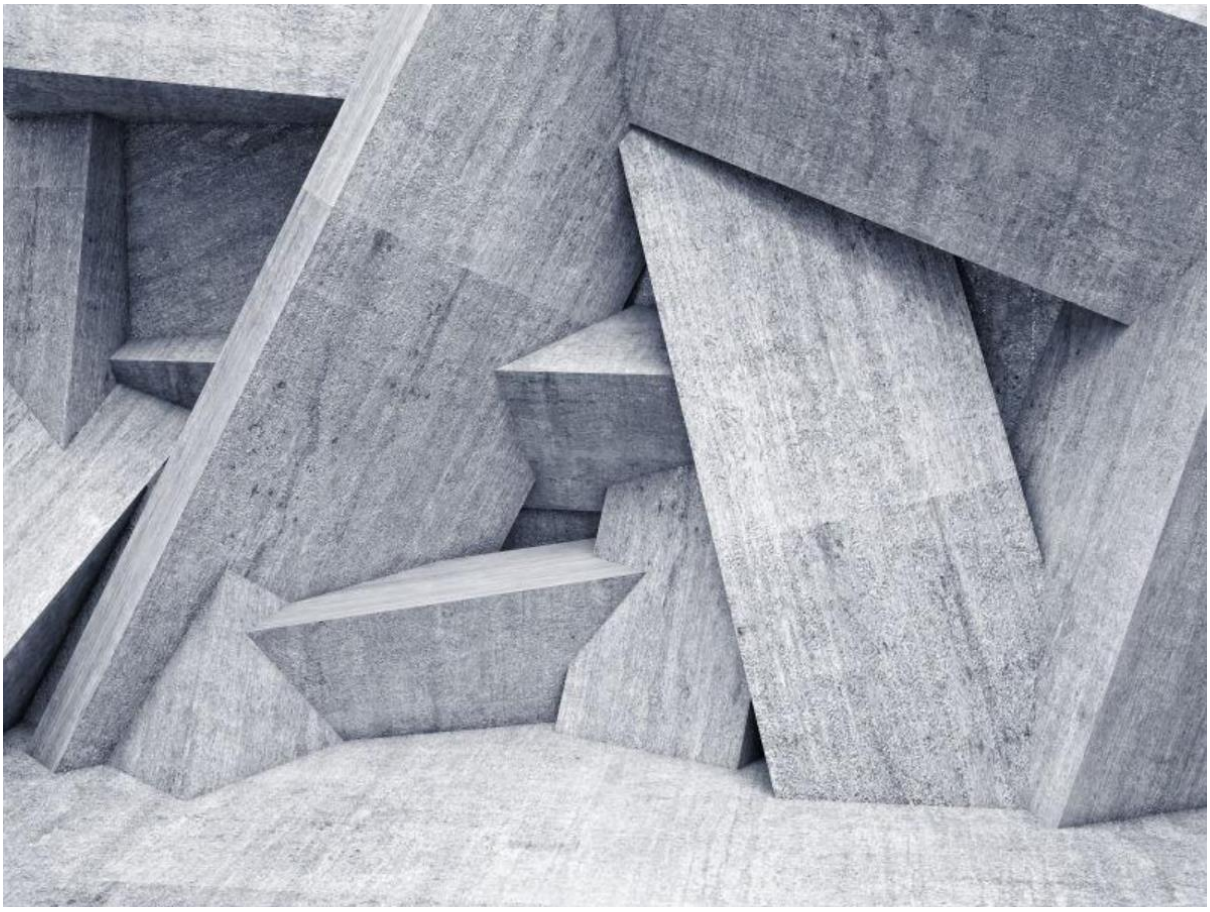
1.4 Key Features

We propose the development of a dictionary software that integrates a comprehensive vocabulary database, encompassing a vast array of words and their meanings. This extensive database will empower users to access precise and reliable word definitions, synonyms, antonyms, and contextual examples, all at their fingertips. Importantly, our software will offer offline functionality, ensuring that users can access the dictionary even in areas with limited internet connectivity, providing uninterrupted learning experiences.

To ensure optimal performance and an exceptional user experience, we will implement a set of advanced query algorithms. These algorithms will enable rapid and efficient searches, delivering instant results and minimizing response times. Users will no longer have to wait for the software to process their queries, allowing them to swiftly delve into their language studies and maximize their learning potential.

In addition to its core features, our dictionary software will prioritize user experience by eliminating intrusive advertisements and unrelated product promotions. This intentional design choice will foster an uninterrupted and focused learning environment, enabling users to concentrate solely on expanding their vocabulary and language skills.

Through this pure dictionary software project, we aspire to provide a reliable and efficient tool that empowers students and language enthusiasts alike. By harnessing cutting-edge technology, comprehensive vocabulary resources, and a commitment to user experience, our software will redefine the way individuals engage with language learning, facilitating their journey towards linguistic proficiency.



2. Market research

2.1 Do you think the current dictionary software ads interfere with your use of the dictionary?

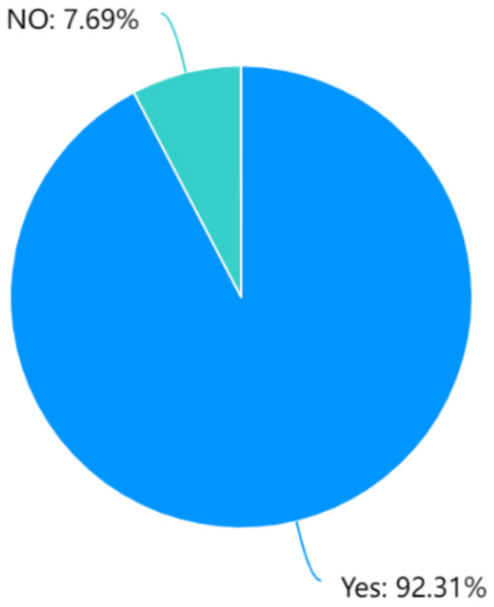


Figure 1: Preference surveys about whether or not to be subjected to AD pop-ups

Analysis: Based on the depicted graph, it is evident that a notable proportion of individuals, specifically 7.69%, express the belief that advertising does not exert any influence on their utilization of dictionary queries. Conversely, a significant majority of 92.31% of people believe that advertising holds a substantial impact on their query experience. These market analysis findings unequivocally demonstrate the immense value attributed to an advertisement-free and unadulterated dictionary.

2.2 Do you think the query interface of a dictionary should be simple and clear or varied and complex?

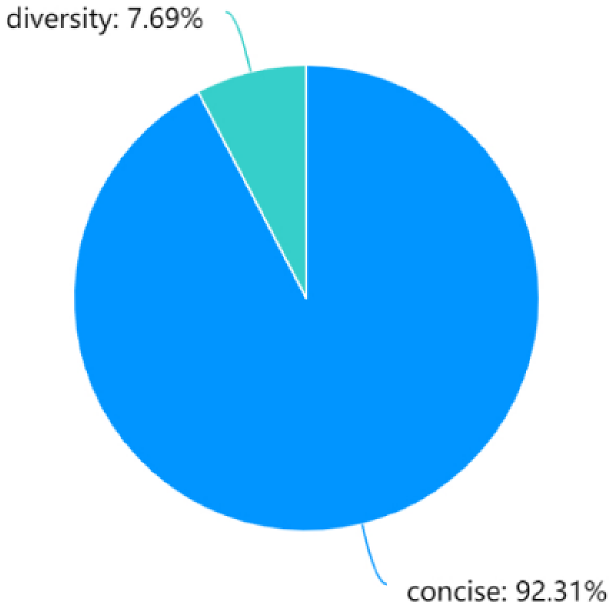


Figure 2: Survey results on interface display mode preferences

Analysis: Upon careful examination of the presented chart, it becomes apparent that a mere 7.69% of individuals advocate for a diverse range of features within the query interface. In stark contrast, an overwhelming majority of 92.31% opine that the query interface ought to be straightforward and unadorned. This insightful market analysis impels us to conclude that the development of a dictionary boasting an uncluttered and user-friendly query interface is imperative to cater to the predominant preferences of the target audience.

2.3 What additional features do you want the translation software to provide to enhance the user experience?

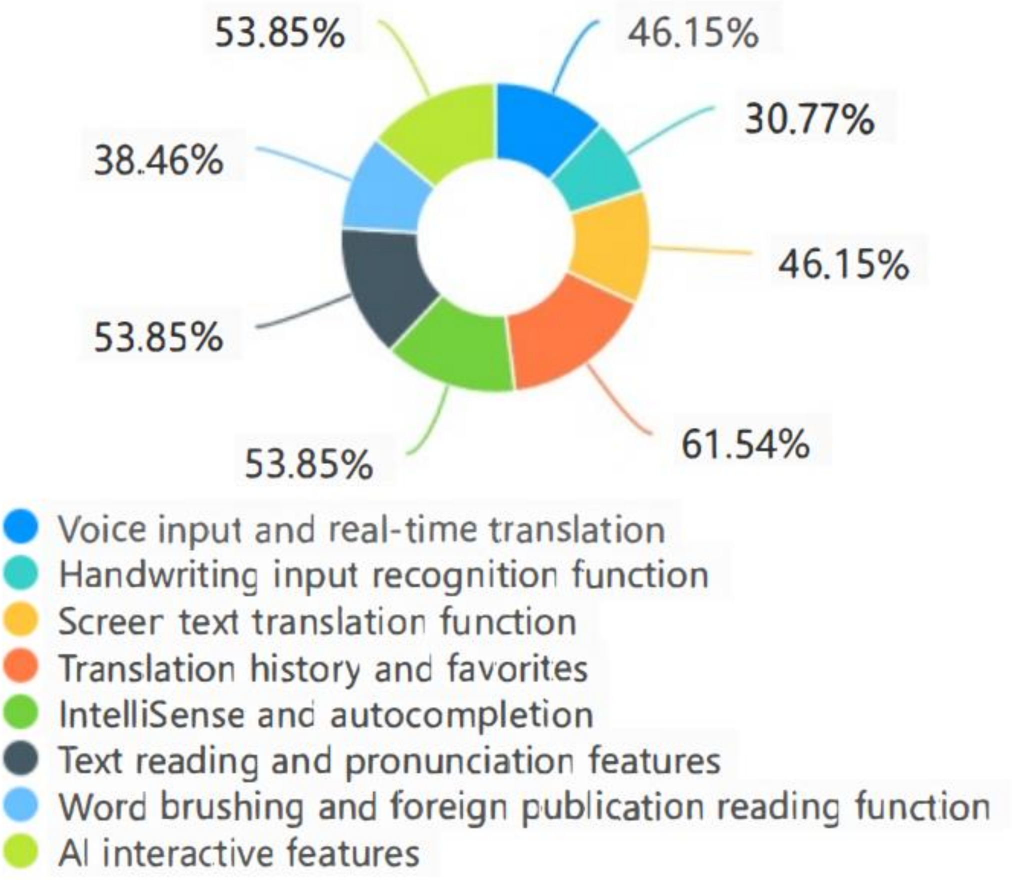


Figure 3: Results of a survey of requirements for extended functionality

Analysis: Based on the figure, it is evident that the survey results regarding this issue exhibit a diverse range of responses. Notably, the functions of favorites and history query surpass 60% in popularity. Autocompletion, text reading and pronunciation, as well as AI interaction, all exceed 50% in user preference. It is clear that each option has garnered a substantial user base. Considering our constrained development resources, we have made the decision to prioritize the development of favorites, history queries, and AI interaction features. The enhancement of other features will be progressively addressed in future iterations of the software upgrade.

3. Functional requirements.

3.1 Use binary search to match words for users

3.1.1 Source of requirement:

As we all know, providing users with matching words is the basic function of every dictionary, and our dictionary naturally needs to have such a function

3.1.2 Requirement description:

When a user has a word that needs to be translated, they enter the app, the front end will display the function for the user to select, when the user selects 1, the front end will ask the user to enter the word and input the word to the back end. The back end introduces the csv library as the word library, and creates a list to import the words in the word library into the list in order. Then the program will use the binary search method to match the lower-case word with the word in the lexicon, and the corresponding translation will be printed if it matches.

3.1.3Requirement priority:

First priority

Abnormal condition:

If the user typed “exit”, the program returns to the selection screen

If the user enters a word with uppercase, convert it to lowercase and then match it

If the program does not find an exact match for the word in the dictionary, try using fuzzy matching

3.1.5Input and output:

Input: word

Output: The corresponding translation of the word

3.2 Error word query, similar words reselect.

3.2.1Source of requirement：

For a variety of reasons, it is inevitable that there will be mistakes when we enter words. If our dictionary has error correction and fuzzy matching methods, it can save users' time or help users clarify the correct spelling of words.

3.2.2Requirement description：

If there is no perfect match for the word, the program will enter a close match, use the algorithm to match similar words for the user and display all of them to the user for selection, and match the new word entered by the user again, and return if it matches. If there are no similar words, the corresponding result is reported back to the user.

3.2.3Requirement priority:

Second priority

3.2.4Abnormal condition:

If the user enters a selection that does not match any close matches, The program would inform the user that the word is not found in the dictionary

3.2.5Input and Output:

Input: Words that don't match exactly

Output: Close matches or Corresponding information

3.2.6Associations with other requirements:

This requirement is a derivative requirement of the word translation requirement and needs to be established in the case that the word query system has been fully developed

3.3 Use youdao interface to realize translation of paragraphs or sentences.

3.3.1Source of requirement:

In our life and work, the translation of sentences is essential, because it can not only improve our efficiency, but also increase our understanding of a language.

3.3.2Requirement description:

When the user needs to translate a sentence or paragraph, type "3" in the input interface to enter the corresponding interface. At this time, the user enters the language and the content that needs to be translated successively according to the requirements, and the program will import these contents into the back-end. The backend uses Youdao api interface, introduces related libraries and sets corresponding parameters at the same time to achieve accurate translation effect, and finally returns the content to the user.

3.3.3Requirement priority:

Second priority

3.3.4Abnormal condition:

If the length of the sentence entered by the user is more than 20, it is divided into smaller chunks of 10, and then analyzed

3.3.5Input and Output

Input: The language selection entered by the user and the paragraph to be translated

Output: Exact translation of the corresponding language

3.4 Front-End interactive code implementation

3.4.1Source of requirement:

As we all know, a translation software needs to be able to interact with users well and give users a good experience, so we need to implement a good front-end interactive system through the program.

3.4.2Requirement description:

First of all, import the dictionary library, and then print the number for the user to choose, the user chooses different numbers corresponding to different functions. Each method is then implemented by connecting the front and back ends through calls to the class.

3.4.3Requirement priority:

Third priority

3.4.4Associations with other requirements:

As the front end of the system, the realization of interactive functions naturally depends on the improvement of the back-end code. Therefore, the above three requirements for translation must be realized first, and then this requirement can be considered.

3.5 History query function

3.5.1 Source of requirement:

When we use a dictionary, we can't look up all the words at once, so if there is a function that can record the historical words we look up, we can quickly continue our work.

3.5.2 Requirement description:

Enter the corresponding number on the front page of the system to enter the dictionary history page, where the user can choose a history record he wants to query, this history record can be a word or a phrase. When the user queries a history, the back end of the system will give the corresponding translation example sentence to the user.

3.5.3 Requirement priority:

First priority

3.5.4 Abnormal condition:

The history is not stored locally indefinitely, in this system, the history will only save the contents of the query within three days, and the query after three days will be deleted.

3.6 Favorites function

3.6.1 Source of requirement:

Dictionaries are often used as tools for learning a language, and learning a language requires memorizing words, so we need to have a favorites function for different users to save the words they need.

3.6.2 Requirement description:

Similarly, users can enter the corresponding number in the front end to enter the favorites interface to view their own favorites words, there are also two options to display words and show the meaning of words in the favorites, so as to assist us to better memorize words. If we want to bookmark a word, we only need to do a specific interaction on the front end, and the back end will put the corresponding word into a list, which will be used as a bookmark.

3.6.3 Requirement priority:

Second priority

3.6.4 Input and output

Input: Interact with the corresponding word on the front end

Output: Corresponding words appear in Favorites.

3.6.5 Associations with other requirements:

Favorites function is established in the case that the query function of the dictionary has been perfected.

3.7 Switch dictionary function

3.7.1 Source of requirement:

As we all know, different dictionaries have different translations and interpretations for different words. In our life scenes, we sometimes tend to prefer one dictionary. For this reason, the function of dictionary switching is essential for our system.

3.7.2 Requirement description:

Enter the corresponding number in the front end, the system will jump to the switch dictionary interface, the system will show the user several dictionary choices, the user can enter the corresponding serial number to select the required dictionary. When the user makes a selection, the back end of the system will automatically switch the dictionary library to the dictionary library corresponding to the user's dictionary.

3.7.3 Requirement priority:

Second priority

3.7.4 Abnormal condition:

If the user enters an unrecognized number when selecting the dictionary, the system will return the corresponding error message and ask the user to enter the correct number again.

3.7.5 Input and output

Input: The sequence number entered by the user

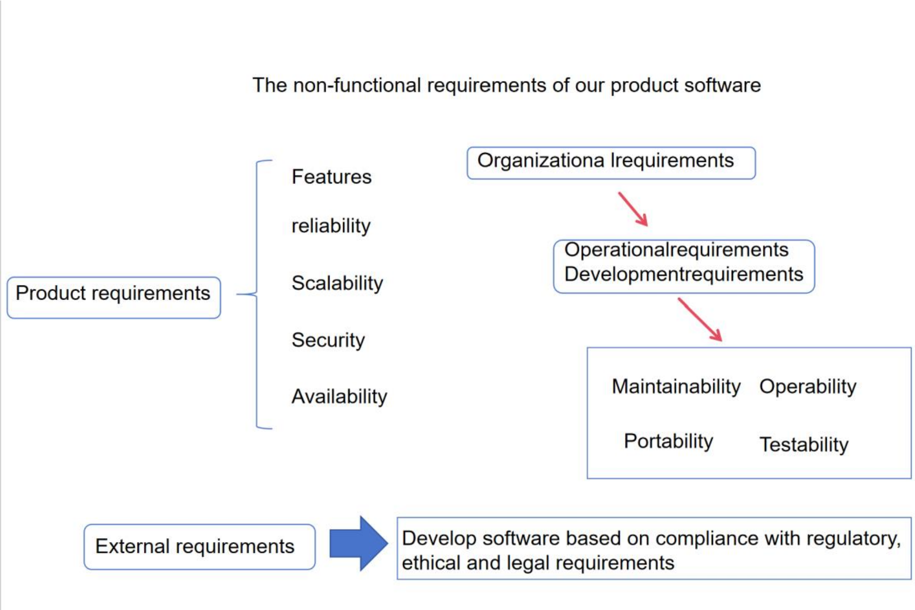
Output: Dictionary base replacement

4. Non-functional requirements.

Non-functional requirements usually specify constrains on the characteristics of the system as a whole. They may relate to emergent system properties such as reliability, response time, and memory use.

Non-functional requirements are often more critical than individual functional requirements. The reason is failing to meet a non-functional requirement can mean that the whole system is unusable.

Non-functional requirements refer to the requirements related to the performance, security, reliability, usability, maintainability, and scalability of a system or software. They do not directly affect the functionality of the system, but they have a significant impact on the overall quality and user experience. Non-functional requirements describe the performance metrics, constraints, and quality standards that the system must meet. Examples of non-functional requirements include response time, concurrent user count, system reliability, data security, etc.


The non-functional requirements of our product software

Product requirements

Features

reliability

Scalability

Security

Availability

Organizationa lrequirements

Operationalrequirements
Developmentrequirements

Maintainability Operability

Portability Testability

External requirements

Develop software based on compliance with regulatory, ethical and legal requirements

Figure 4: The non-functional requirements of our product software

4.1 Performance

Optimize data structures and algorithms, such as using hash tables or balanced binary trees to implement dictionaries, to improve the efficiency of lookup, insert, and delete operations. In addition, our product can be performance tested and optimized to ensure good performance even when dealing with large amounts of data or high concurrency. For example, when the user enters a lot of content or when the user enters a lot of content, our dictionary can still remain efficient

4.2 Reliability

By implementing the exception handling mechanism, transaction processing and data backup mechanism, to ensure the reliability of dictionary operations. The product can catch and handle exceptions to avoid system crashes or data loss. In addition, Those words that are stored in the dictionary or personalized content about themselves will upload data periodically to prevent loss.

4.3 Scalability

The product can use scalable architecture and design patterns, such as distributed architecture, microservices architecture, or modular design, to support the scalability requirements of dictionaries. By splitting dictionaries into multiple independent modules or services and using techniques such as message queuing or distributed caching, it is possible to scale horizontally or vertically to handle larger data volumes and user requests. Make the dictionary more stable and can make the user's request can be answered faster

4.4 Security

The product can implement user authentication and authorization mechanisms to ensure that only authorized users can access the dictionary. In addition, the product can use encryption algorithms to protect the privacy and security of user data, such as encrypt sensitive data stored by users in the dictionary to prevent leakage

4.5 Usability

The product can make dictionaries available by providing a user-friendly interface, good error handling and status alerts, and monitoring and automated operations tools. By implementing load balancing and fault recovery mechanisms, as well as monitoring the health status of the system, you can ensure that the dictionary can run normally at any time and provide reliable services.

4.6 Maintainability

The product can adopt a clear and readable code style, using comments and documentation to explain the functionality and design of the code. In addition, the products can adopt modular and object-oriented design principles, dividing the functions of dictionaries into independent modules or classes for easy maintenance and upgrading. Reduce the user confusion caused by dictionary updates.

4.7 Operability

The product can provide a user-friendly interface or command line interface for users to configure and manage dictionaries. In addition, The product can provide flexible configuration options, Allows users can customize the Settings

4.8 Portability

The product can adopt cross-platform technologies, such as using cross-platform programming languages or frameworks, to achieve portability of dictionaries across different operating systems and environments. In addition, The product should follow common coding specifications and standards to improve the portability of the products. Allow users to use the same dictionary for different platforms, so that users do not have to adapt to a new dictionary

4.9 Testability

The product can use unit testing, integration testing, performance testing and other methods to test the dictionary comprehensively. By writing testable code and using automated testing tools, the quality and stability of dictionaries can be ensured. ensure user experience

Through the above practice, our products should be able to meet the dictionary performance, reliability, scalability, security, availability, maintainability, operability, portability and testability and other non-functional requirements.

5. Use Case Diagram And Description Of Use Cases

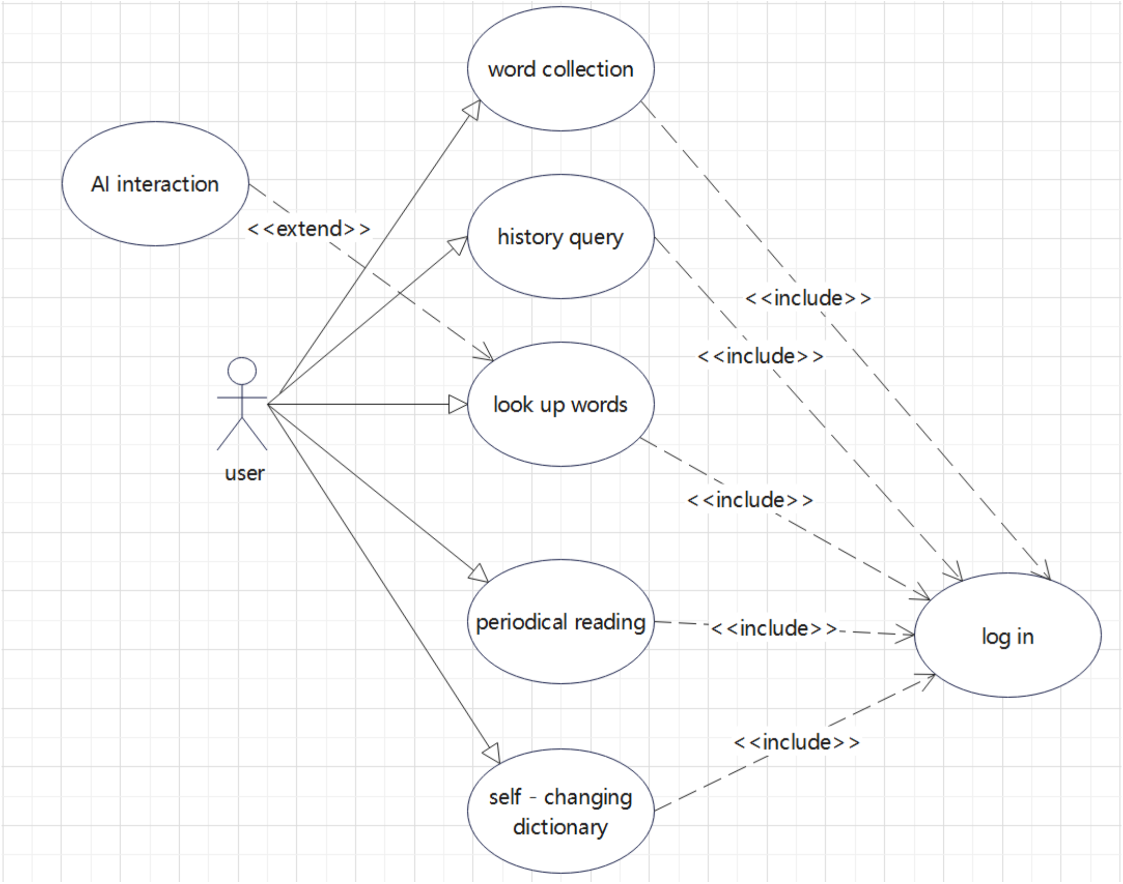


Figure 5: A use-case diagram describing the user and the dictionary system.

This is the use case diagram for the ABD dictionary. As shown in the diagram, users are associated with five functions: word collection, historical record search, word search, periodical reading, and dictionary change. All of these functions include user login operations, which means that users can access these five functions after logging into the dictionary software. Additionally, in the word query function, we have expanded the functionality to include AI interaction, allowing users to optionally utilize AI interaction for an enhanced query experience.

6. Context Model

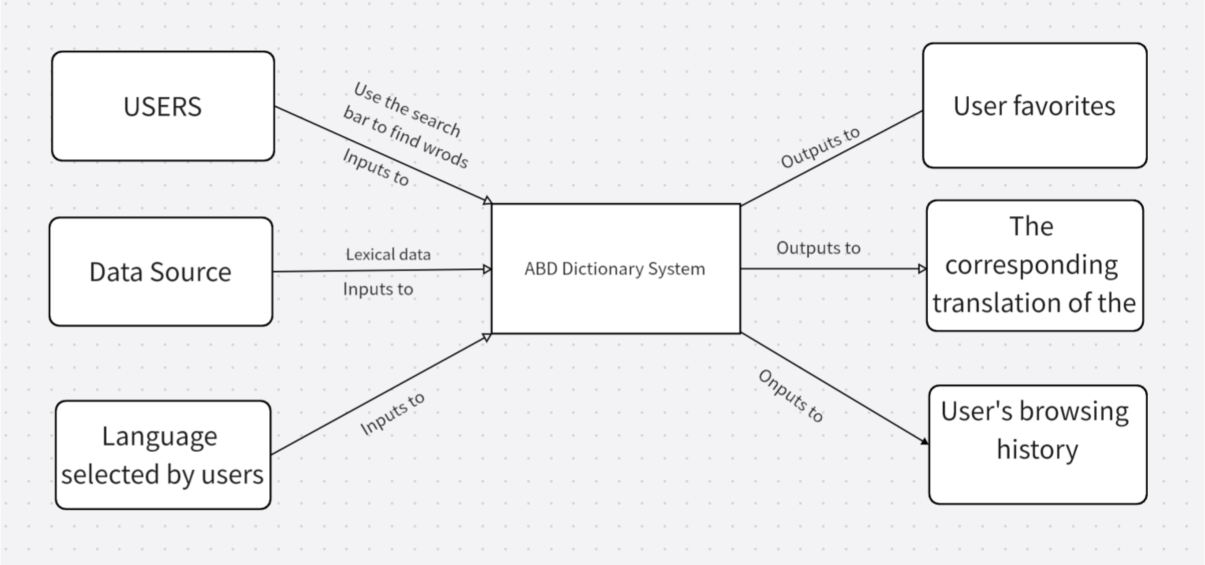


Figure 6: A Context diagram describing the user and the dictionary system.

The ABD Dictionary system in intended to translate text entered by the user in the language of the user's choice. The user base for the ABD Dictionary system is diverse, and users have a wide range of different system requirements, so the system collects user data and generates query history in order to better provide more accurate translations for different users. And the system will find an accurate or similar translation of the word in the database according to the words entered by the user, and then output it.

7. Business Process Models

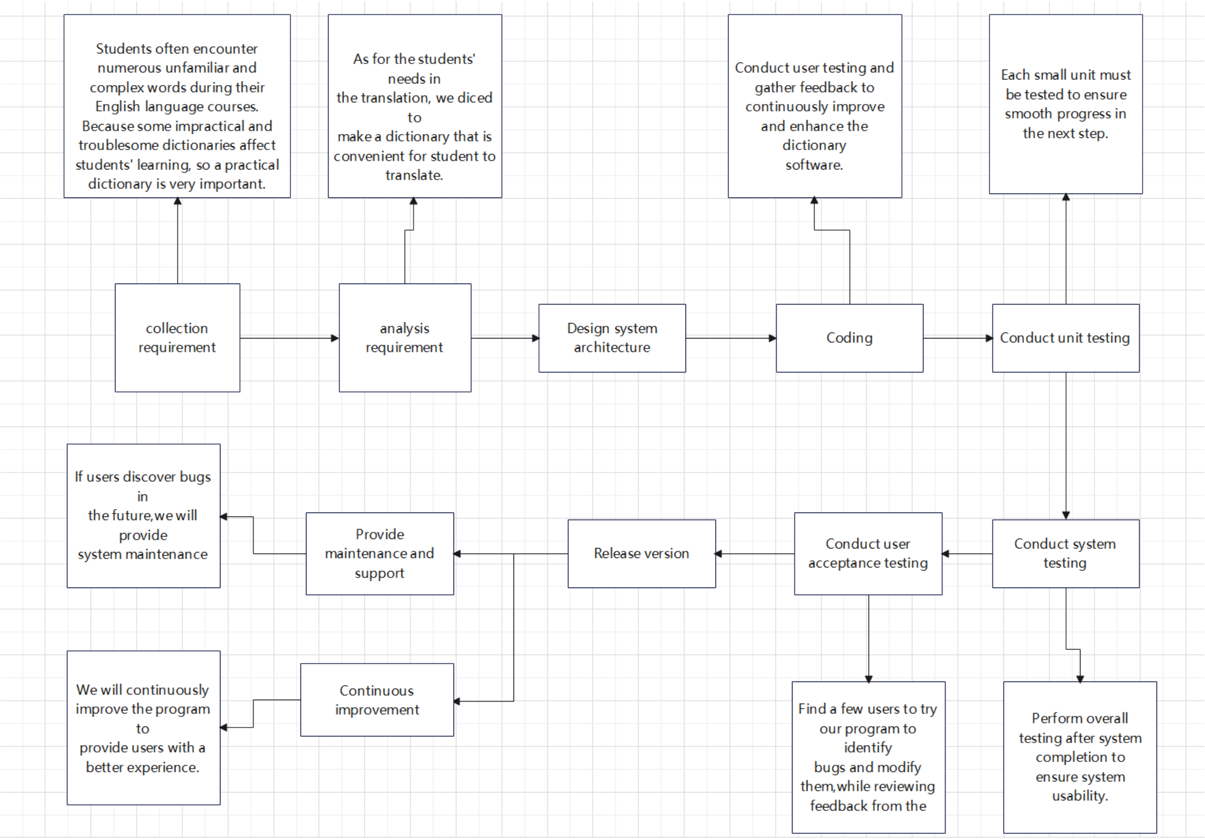

Students often encounter numerous unfamiliar and complex words during their English language courses. Because some impractical and troublesome dictionaries affect students' learning, so a practical dictionary is very important.
As for the students' needs in the translation, we diced to make a dictionary that is convenient for student to translate.
Conduct user testing and gather feedback to continuously improve and enhance the dictionary software.
Each small unit must be tested to ensure smooth progress in the next step.
collection requirement
analysis requirement
Design system architecture
Coding
Conduct unit testing
If users discover bugs in the future,we will provide system maintenance
Provide maintenance and support
Release version
Conduct user acceptance testing
Conduct system testing
We will continuously improve the program to provide users with a better experience.
Continuous improvement
Find a few users to try our program to identify bugs and modify them,while reviewing feedback from the
Perform overall testing after system completion to ensure system usability.

Figure 7: A legend that describes the business process for the project

7.1 Collection requirements

We know from students that many students are troubled by translation problems. Many paper dictionaries are very troublesome, not conducive to portability and low efficiency. Some electronic dictionaries also have the function of charging fees, and the intrusion of advertisements also makes many students feel disgusted. According to the needs of the students, we launched our electronic dictionary, our Aberdeen dictionary is free of charges and advertising, easy to use, is a dictionary dedicated to the use of students.

7.2 Design system architecture

Designing a system architecture involves systematically planning and organizing the structural components and elements that make up a complex software or technology infrastructure. It is the blueprint of the entire system, defining how the different parts work together to efficiently achieve a specific goal. A well-designed system architecture includes key decisions about system components, their interactions, data flow, scalability, security, and performance considerations. This architecture provides a framework for developers and engineers to guide, provides a clear roadmap for building and maintaining systems, and it ensures that all system elements are seamlessly integrated. In essence, it is the structural basis for building complex software, hardware, or network systems.

7.3 Coding

Coding involves crafting and building logic. Our programming team put a lot of effort into this hard work. Successful coding requires not only technical proficiency, but also problem-solving skills, as developers often encounter challenges and errors that require logical troubleshooting and debugging. Collaboration and clear documentation are also important, especially in team-based projects, to ensure that the code is understandable and maintainable for others. Therefore, the difficulty of programming is very high, which also highlights the spirit of our hope to serve the students.

7.4 Texting

Texting our software is a process that involves communicating with or sending text messages to a software program, application, or system. This interaction is often essential for various purposes, including user input, data retrieval, or system commands. It is commonly used in customer support, where users can send text messages to request assistance or information.

Texting our software underscores the importance of user-friendly and responsive interfaces, enabling users to interact with technology in a way that is intuitive and accessible.

7.5 Provide maintence and support

Providing maintenance and support to our users is a critical aspect of ensuring the ongoing functionality, reliability, and satisfaction of any software application, product, or service. It involves a dedicated commitment to addressing and resolving issues, bugs, or challenges that users may encounter during their interaction with the software. Maintenance encompasses regular updates, bug fixes, and system optimizations to keep the software running smoothly and securely. Support, on the other hand, entails offering assistance and guidance to users who may have questions, encounter technical difficulties, or require additional information about the software's features or usage. This collaborative approach not only enhances user experience but also fosters trust and loyalty among customers. By providing timely and efficient maintenance and support, we demonstrate our commitment to delivering a high-quality product and a positive user experience. This, in turn, contributes to the long-term success and sustainability of the software, fostering user satisfaction and loyalty, and potentially leading to valuable feedback that can drive further improvements and enhancements.

8. Conclusion

In conclusion, the development of an innovative and user-friendly dictionary software project is of utmost importance in revolutionizing the way students and language learners access and utilize linguistic resources. By optimizing the performance, reliability, scalability, security, usability, maintainability, operability, portability, and testability of traditional dictionary software, we aim to provide a reliable and efficient tool that empowers individuals in their language learning journey.

Through the integration of comprehensive vocabulary databases, advanced query algorithms, and offline functionality, our software will provide users with a seamless and efficient word lookup experience. By eliminating intrusive advertisements and irrelevant promotions, we strive to create an uninterrupted and focused learning environment.

The anticipated results of this project include maximized learning potential, increased language proficiency, and enhanced productivity for students and language enthusiasts. By leveraging cutting-edge technology and a commitment to user experience, our software will redefine the way individuals engage with language learning, facilitating their journey towards achieving linguistic proficiency.

9. References

Solomon Teferra Abate, Michael Melese, Martha Yifiru Tachbelie, Million Meshesha, Solomon Atinafu, Wondwossen Mulugeta, Yaregal Assabie, Hafte Abera, Binyam Ephrem Seyoum, Tewodros Abebe, et al. Parallel corpora for bi-directional statistical machine translation for seven ethiopian language pairs. In Proceedings of the First Workshop on Linguistic Resources for Natural Language Processing, pages 83–90, 2018.

Željko Agić and Ivan Vulić. JW300: A wide-coverage parallel corpus for low-resource lan guages. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3204–3210, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1310. URL https://www.aclweb.org/anthology/ P19-1310.

Roee Aharoni, Melvin Johnson, and Orhan Firat. Massively multilingual neural machine translation. arXiv preprint arXiv:1903.00089, 2019.

Orevaoghene Ahia and Kelechi Ogueji. Towards supervised and unsupervised neural machine translation baselines for nigerian pidgin. arXiv preprint arXiv:2003.12660, 2020.

Lo Alla, Dione Cheikh Bamba, Nguer Elhadji Mamadou, Ba Sileye O Ba, and Lo Moussa. Using lstm to translate french to senegalese local languages: Wolof as a case study. arXiv preprint arXiv:2004.13840, 2020.

Anonymous. Syntactic relevance xlnet word embedding generation in low-resource machine translation. OpenReview, 2020.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, et al. Massively multi lingual neural machine translation in the wild: Findings and challenges. arXiv preprint arXiv:1907.05019, 2019.

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. arXiv preprint arXiv:1802.06509, 2018.

Mikel Artetxe and Holger Schwenk. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. In https://arxiv.org/abs/1812.10464, 2018a.

Mikel Artetxe and Holger Schwenk. Margin-based parallel corpus mining with multilingual sentence embeddings. arXiv preprint arXiv:1811.01136, 2018b.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. arXiv, abs/1607.06450, 2016.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473, 2014. Ankur Bapna, Naveen Arivazhagan, and Orhan Firat. Simple, scalable adaptation for neural machine translation. arXiv preprint arXiv:1909.08478, 2019.