

|  |
| --- |
|  |

School of Computing

Final Year Project

UP769535 – Matthew Anthony James Hawkins  
Supervisor: Petronella Beukman

PJE40

Project Report

A flashcard web application to help efficiently learn Japanese

**Table of contents**

1. Introduction

1.1 Problem Statement

1.2 Project Aim

1.3 Understanding the Japanese Language

2. Literature Review

2.1 Abstract

2.2 Introduction

2.3 Methodology

2.4 Second Language Learning and Acquisition

2.5 Learning Vocabulary

2.6 Analysing existing software

2.7

3. Methodology

3.1 Development Methodology

3.2 Software and Technology Decisions

3.3 Requirements Specification

4. Design

4.1 Design Decisions

4.2 Use Case Overview of the Application

4.3 Low Fidelity Prototypes

4.4 Database Design

5. Implementation

6. Testing and Evaluation

6.1 Low Fidelity Navigational and User Interface Testing

6.2 Client Evaluation of Current System

6.2.1 Results and Requirements Comparison

7. Project Management

8. Conclusion

8.1 Future Development

9. References

10. Appendix

Introduction

* 1. Problem Statement

In the language learning domain there are a large some of services that are directly aimed at language learning, while others services are aimed at learning in general. However out of the many services that are currently available, only a small portion of these services are built upon research on how we learn, and in the terms of languages, how we acquire foreign languages.

1.2 Project Aim

The overall aim of the project is to allow language learners of Japanese to study more efficiently via simple flashcards. To be able to do this some research into language learning, spaced repetition and other learning techniques will be necessary.

In order to reach this aim a literature review will be carried out and any research that is found that suggests certain learning techniques are great for language learning will be noted and if possible, these will be turned into requirements to implement in the system.

The application will aim to achieve these aims by simulating physical flashcards using a question-answer pair.

1.3 Understanding the Japanese language

Modern Japanese is a partially logographic language that makes use of both its own two syllabaries and an adopted Chinese writing system. The adopted Chinese writing system in Japanese is known as Kanji and the two syllabaries are known as Hiragana and Katakana. All three of these systems combined make up the modern Japanese writing system. For this report, whenever Japanese text is used the corresponding Romanization (Romaji) of the characters will be shown too so that one can read the Japanese.

Japanese Kanji are often what learners of Japanese find the most difficult when it comes to learning the language. This is because the writing system is very complex with around 2000 common use Jōyō Kanji characters being used in printed texts and, although most of them are not in common use, over 50,000 other characters in the language according to The Great Chinese-Japanese Dictionary by Morohashi (2000).

Each Kanji character usually has 1 or more similar meanings and by combining characters together you can produce new words with similar or different meaning. For example 通, *meaning traffic; pass through; commute*, and 報, *meaning report; news; reward*, can be put together to make the word通報, *meaning report; notification; tip; bulletin*. Japanese Kana, Hiragana and Katakana, are two syllabaries that are made of characters that have 1 sound per character. For example あ is pronounced “ah” and written in Romaji as “A”. Unlike Kanji, the characters in Kana do not hold any meaning, much like most letters in English.

Hiragana is mainly used for words of Japanese origin or for Furigana. Furigana is Hiragana, and sometimes but more rarely Katakana, that is written above Kanji to tell the reader how a Chinese character is pronounced. 通報 is pronounced *tsūhō* (Hiragana:つうほう) and the Furigana version is:

Katakana is mainly used for loan words brought into Japanese from other languages, such as バイク, *lit. baiku, meaning bike*. Katakana is the same as Hiragana in terms of pronunciation, but each character is written differently. This means that each Hiragana has a counterpart Katakana. For example ア is pronounced “ah” and written in Romaji as “A” and has the Hiragana counterpart あ.

1. Literature Review

**Flashcard-based design and spaced repetition for language acquisition in Computer Aided Language Learning (CALL): a literature review**

2.1 Abstract

This literature review will explore the various different techniques, methods and tools that modern day language learners use and how they use them. It aims to give an insight into what is needed to acquire a second language (L2) and how we can apply this knowledge with Computer Aided Language Learning (CALL).

2.2 Methodology

Google Scholar and the University of Portsmouth’s library Discovery and catalogue search where used to find articles and books to get a feel and overview of the topic areas, journals and research. All resources found had to be in English and relevant to the topic area. As well as this, articles related to computer assisted learning where limited to articles from the past 15 years (2003 and onwards) to ensure relevancy.

Google search was used to find existing software that would be like the final product. The search query used to find the existing software was “free spaced repetition flashcard applications for language learning”. We then took the first 10 results and noted the amount of times a software was mentioned to choose 3 software which should be like the final artefact. We decided not to include FluentU in the final 3 software as the app was promoted by their own website and while the mobile versions for Anki are also mentioned on some sites, we decided to focus on the desktop application as it was mentioned heavily. See Table 2 in appendix.

The articles found in language and linguistics journals where limited to second language acquisition. The target end users are familiar with and use an immersion approach to learning languages which is based upon research done by linguist Stephen Krashen and as such his works will also be investigated.

The articles found in psychology journals where selected upon relevance to language acquisition and/or learning with technology.

Research was also limited to what was available online and in the University of Portsmouth library. Many articles that appeared relevant to the topic at hand where inaccessible through the University of Portsmouth on some research database websites.

The main subject areas searched include:

* Language learning
* Language acquisition
* Spaced repetition and the Spacing effect
* Mnemonics in language learning
* Japanese Written Language
* Flashcards and existing flashcard software

Combinations of these keywords where used along with advanced search options such as quotation marks (“”) and the negative symbol (-) to filter out irrelevant results in Google Scholar.

2.3 Introduction

This literature review uses a combination of academic articles, books, internet webpages and existing systems to answer some important questions that will have a large impact on the design of the application. These questions include: How do we acquire language? How does the design of a flashcard-based Computer Aided Language Learning (CALL) application affect a user’s ability to acquire and retain language? What effect does spaced repetition have on the acquisition and retention of language in CALL applications? What other learning techniques can be used with such CALL applications to increase the efficiency of language acquisition and retention?

Language learning has dramatically increased in accessibility to the public in recent years due to the advancement in technology and software development. Before the huge step in functionality with the Apple iPhone in 2007, many language learning projects where constrained by poor audio quality, limited storage capacity, poor internet connectivity, difficult text entry and low-resolution screens on older handheld devices, Godwin-Jones (2011, p. 2).

However nowadays we have an abundance of language learning software, a lot of which are just simple apps with no real research on language acquisition to back up their teaching and learning methods. This literature review aims to look at the research behind language acquisition, as well as existing systems, to draw up requirements for an efficient language learning web app.

2.4 Second Language Learning and Acquisition

There are a variety of different approaches currently being used to learn second languages and while often these are intertwined together in a variety of ways, there are two main approaches that learners tend to choose.

The first is where the learner learns the “building blocks” of a language with a bottom-up approach, starting from the basics of the language’s grammar and vocabulary, learning how to string sentences together and how the language’s alphabet is pronounced. A good example of this would be standard language classes that uses a textbook as the main material and where a teacher takes the learner step by step through different structures. According to Krashen (1982, p. 10) this is language *learning* and refers to conscious knowledge of a second language, i.e. knowing the rules and being able to talk about them and that this is independent from language *acquisition.*

The second is to use a top-down approach where the learner spends as much time exposed to the foreign language as possible. The idea being that the longer and more time you spend with the language, the more you will become able to understand and eventually output. This type of approach is common in immersion or bilingual programs used at schools in countries where there are one or more official languages, for example in Canada where both English and French are official languages (Central Intelligence Agency, n.d.). Immersion-based strategies work better than standard classes and going abroad to a country which speaks the target language according to research by Freed, B. F., Segalowitz, N., & Dewey, D. P. (2004, p. 276).

Krashen (1982, p. 10) says that these two approaches are in fact two distinct and independent ways of developing second language competence. He states language *acquisition* to be “a process similar, if not identical, to the way children develop ability in their first language” and that conscious language *learning* helps us only as a “Monitor”, which can change the output, speaking/writing, of the language *acquisition* system before or after output, Krashen (1982, p. 15). This implies that conscious learning plays a limited role in second language performance therefore language acquisition is the more important to focus on in order to gain language ability.

The Input Hypothesis, Krashen (1982, p. 21), states that “We acquire by understanding language that contains a structure that’s just beyond our current level of competence *(i + 1)*. This is done with the help of context or extra-linguistic information.” This is something that immersion learners will experience due to the massive exposure they get to the second language. The more exposure the learner gets to the second language, the more chances there are of being exposed to *(i + 1)* comprehensible input which can explain why immersion-based programs work better than study abroad and standard classes. The concept of comprehensible input is now widely accepted by most applied linguists to be necessary for language acquisition, Cummins (2000, p. 8). Learning from comprehensible input would imply that language learning software should focus less on trying to teach the language and instead simply provide language learners with comprehensible language aimed towards their current level.

2.5 Learning vocabulary

There are a variety of different approaches that people use when trying to learn vocabulary in Japanese. Kanji make up the building blocks of a big part of Japanese vocabulary, but a lot of the approaches are used for both Kanji and Japanese words.

**Rote repetition**

Rote memorisation tends to be a go to approach for Japanese learners particularly when it comes to learning Kanji. Japanese children are taught to repetitively write out Kanji by hand until they know how to write them, and this approach makes its way to Japanese learners from textbooks and native Japanese teachers as rote learning is one of the most common strategies for teaching Kanji, Green & Shimizu (2002, p. 235). However, not only is rote repetition less efficient than spaced recall, Atkinson (1972, p. 126) but learners of Japanese that have no previously knowledge of Kanji will find it harder using this approach than a Japanese child learning Kanji in Japan that already has seen plenty of Kanji in daily life.

**Word lists**

Vocabulary lists of words out of context have been one of the more popular approaches to teaching and learning vocabulary in second language learning. While word lists provide an efficient study method where students can quickly learn a large amount of words in a short period of time, Dolch (1949, p. 142) points out that, as an example, just because a child might understand all the meanings of all words, it doesn’t mean that they can fully understand what they read. Word lists lack contextual information to help produce vocabulary acquisition. However, word lists can be used in conjunction with other techniques such as using imagery or an example sentence to provide context and word usage.

**Mnemonics**

Using memory techniques can be effective when learning Japanese vocabulary, particularly when it comes to Kanji. Parts of, or whole, Kanji can be combined to form new Kanji and whole words. Remembering the Kanji, a book by Heisig (2001), teaches learners a Kanji’s meaning and its stroke order by using mnemonic stories and keywords for each character. In the example below we have the character 泊, meaning *overnight,* which is built up of the components ⺡, meaning *water*, and 白, meaning *white*.



Figure 1: Example of Heisig’s (2001, p. 74) method.

By combining keywords from the components that build up a character, and the characters keyword, the learner can create mnemonic stories that allow them to remember a character for when they need to write it. This method is extremely fast in allowing learners to learn the meanings and how to write Kanji characters, but it doesn’t teach the learner how to read each of the characters. Learning lists of vocabulary using mnemonic devices show considerably better results over other strategies when building vocabulary according to Meara (1980, p. 225) and Cohen (1987, p. 59) and while Meara (1980, p. 225) does point out that research on mnemonics often "completely ignore the complex patterns of meaning relationships that characterise a proper, fully formed lexicon, as opposed to a mere word list”, if a learner already associates a word or Kanji with a particular meaning then learning its pronunciation afterwards might be easier in the long-term.

**Vocabulary acquisition by reading**

There is extensive research that demonstrates that vocabulary acquisition through reading is possible (Krashen, 1989, p. 454; Ponniah, 2011, p. 138; Hafiz, 1989, p. 10). Beheydt (1987, p. 63) states that we need context when learning vocabulary: “From a psychological as well as a linguistic point of view, undeniably the first guideline would be that vocabulary must be learned in context. The meanings of words are more easily somaticized if they are embedded in a meaningful context”. With that all being said, there is a major issue with vocabulary acquisition via reading. This being that vocabulary acquisition via reading assumes that the learner already has a basic level of reading skill in the L2. Learning a language with a different orthography, such as English-speaking natives learning Japanese, can be particularly difficult (Bhide, 1990, p. 9). In fact, according to Laufer (1989, p. 319), the reader must be able to understand around 95% of words in a text to be able to acquire the words in the text they do not know. As a result, a means for making texts more comprehensible to lower this percentage is required if reading is to be used for vocabulary acquisition with beginners.

**Learning vocabulary with sentences**

This type of vocabulary learning is recommended by Antimoon (n.d.), a website about learning English written by two Polish men who learnt English as a second language. They recommend using sentence flashcards in conjunction with a spaced repetition software, such as Anki (n.d.) or SuperMemo (n.d.).

As with learning vocabulary by reading, it’s much more likely that a better understanding of a word’s correct usage will be gained by seeing the word used in a variety of contexts than by seeing it in a single dialog or in a list as noted by Godwin-jones (2010, p. 4).

There’s also a couple of other reasons why sentence flashcards are favoured over other card types to learn vocabulary. These include being easier to make and review than standard L2 to L1 vocabulary cards. This is because the learner can take sentences from native sources by copying and pasting them into their flashcard program and they are easy to review because the goal is to simply understand and read the sentence correctly.

**Spaced repetition and the spacing effect**

Spacing out review sessions allows a person to consolidate their understanding of a subject, to learn partially missed or completely missed information on the previous run and to also give the learner feedback to allow them to test their correctness of the knowledge that they gained from the previous run, David P. & Mohamed (1965, p. 149). As such the learner will gain a deeper understanding of the topic with each review. The spacing effect appears to have huge potential for improving classroom learning, Dempster (1988, p. 632), as well as in second language acquisition, Godwin-Jones (2010, p. 7).

An experiment by Bloom & Shuell (1981, p. 247) showed that distributed practice allowed learners to remember “substantially more words than those students who had studied under conditions of massed practice”, finding that the performance of the distributed practice group was 35% better.

Melton (1970, p. 603) and Seabrook, Brown, Solity (2005, p. 119) discuss a “lag effect” which causes learners to learn better when the spaces between study periods gradually increase each time and Pavlik & Anderson (2005, p. 567) also found that the spacing effect becomes cumulatively beneficial due to each spaced practice providing an additional advantage suggesting that as time goes on, and the period between reviews increases, the better an item is understood and stored in long term memory.

The results from Cepeda et al. (2009, p. 244) also suggest that the gaps between learning sessions should be increased to months, instead of days or weeks to “efficiently promote truly long-lasting memory” which coincides with the lag effect discussed by Melton (1970, p. 603) and Seabrook et al., (2005, p. 119).

**Flashcards**

Flashcards often have a front and a back where the user must recall the back of the card to get the flashcard correct. Flashcards are a great way of increasing second language vocabulary as they often make the learner retrieve and recall L2 vocabulary from memory, which facilitates learning as shown in research by Barcroft (2007, p. 48) and Karpicke & Roediger (2008, p. 967).

Phillips (2011, p. 61) suggests that using an intelligent and computer-controlled spaced repetition flashcard system, such as Anki (n.d.) or SuperMemo (n.d.), might be an effective way to improve scores in language tests and Kornell (2009, p. 1314) looked into optimising flashcards and found that “flashcards should be studied in relatively large stacks across multiple days.” as well as that spacing is more effective than cramming.

2.6 Analysing existing software

**Anki**

Anki (n.d.) is an example of a spaced repetition flashcard software that’s used by a range of students for many different subjects (AnkiWeb, n.d.). Anki (n.d.) claims that “it's a lot more efficient than traditional study methods, you can either greatly decrease your time spent studying, or greatly increase the amount you learn.”. According to Anki’s Manual (n.d.), “Anki’s spaced repetition algorithm is based on the SuperMemo 2 algorithm”. This is an older version of the algorithm that the flashcard app SuperMemo (n.d.) uses. Anki has a mobile, desktop and even a web version which all allow varying degrees of functionality. For this evaluation, we will be looking at the software designed to run on a Windows operating system.

Anki is a simple looking piece of software which allows users to create simple Front and Back flashcards, however, the software provides a flexible card and note creation system that allows users to make almost any type of flashcard they want by adding, removing and editing “Fields” on the cards.

It also includes a very well organised “syncing” system which forces a sync every time the program is open or closed to keep your flashcard reviews and changes up to date across all devices. Of course, this does mean that data loss can occur by accidently overwriting progress made on a device, however Anki also has local backups too in case of these events or in case of the loss of your entire collection.

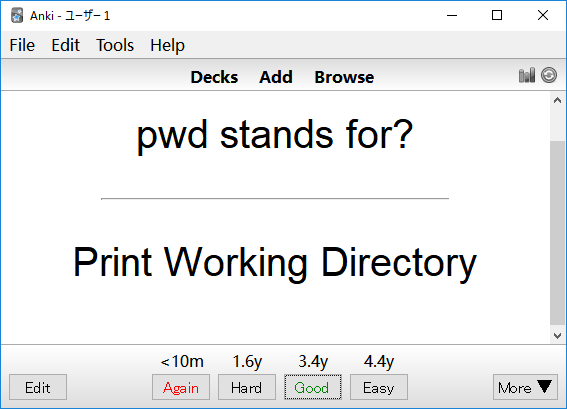
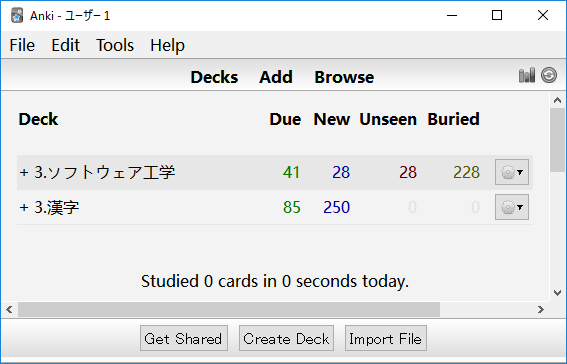


Figure 2. Anki’s home screen. Figure 3. An example of a flashcard in Anki.

The layout and style of cards can be changed very easily in Anki as the software uses HTML and CSS to display cards and this code can easily be changed by the user. Users can also create and share decks, note types and even addons to add extra functionality.　Creating cards is as simple as adding text, audio, photos etc. to text boxes for either side of the card.

Anki without addons supports Japanese characters but doesn’t allow the displaying of furigana. To get furigana functionality, the learner must install a Japanese Support (n.d.) addon, which isn’t perfect in its furigana generation and makes simple mistakes like generating いちにん (ichninn) for the word 一人, which is more commonly pronounced as ひとり (hitori).

Anki has default learn and review settings that the developers believe is best for the user, however they do allow you to change these, along with a massive amount of other settings, to improve your experience. Some of the default settings are somewhat questionable though, and some users (Optimize your Anki, 2017) of the program have created addons and suggested better settings to increase your efficiency.

Anki doesn’t provide any content for the user besides other user created content and as such the user must either use this content or make their own flashcards. This can be seen as both good and bad as it gives the user freedom in creating flashcards, but the user may not know the best techniques for flashcard layout for what they are attempting to learn or may learn mistakes that are in another person’s content.

**Memrise**

Memrise (n.d.) is slightly different to Anki as it, by default, uses a variety of different card types when learning and reviewing information. Memrise has courses that are created by Memrise itself but also community made courses too which bring up the same issues just mentioned with Anki’s shared user content.

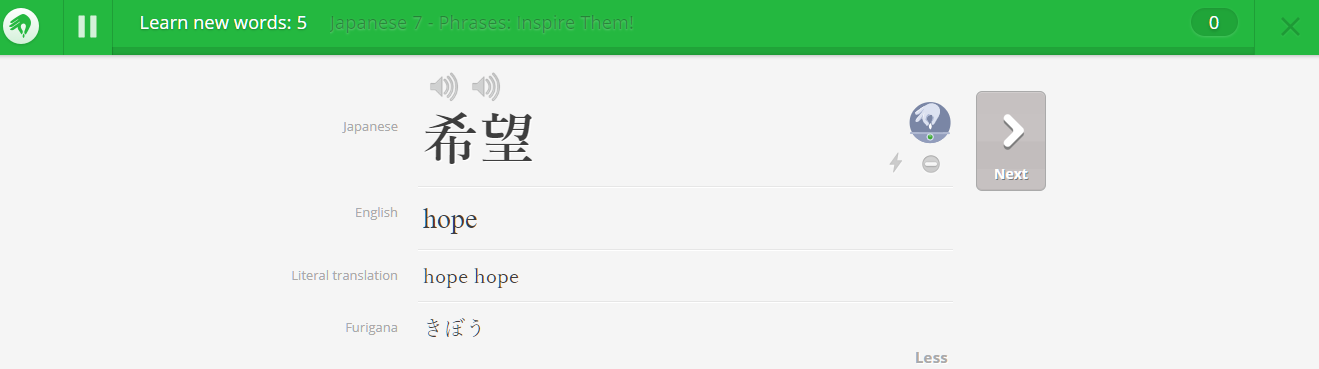


Figure 3. Example of a “Learn new” flashcard in Memrise.

The courses that are made by Memrise often come with native audio for words and sentences. Of course, users can create their own courses too so there is also a wide variety of different flashcards available.



Figure 4. Example of reviewing a flashcard after just learning it in Memrise.

Once a couple of new words are learnt, you are instantly tested on them by getting the English definition of the word and you are then tested to see if you know the correct Japanese, or vice-versa. Once the answer is selected, the native audio plays too. Having the audio on the back, after the user has thought about and answered the card appears to be better than having the audio said before the user answers as the latter gives away too much information to the user. Similarly having multiple answers may also be too easy for the student as this promotes recognition, which we know not to be as good as recall for long term memorisation **(insert citation about recall being better).**

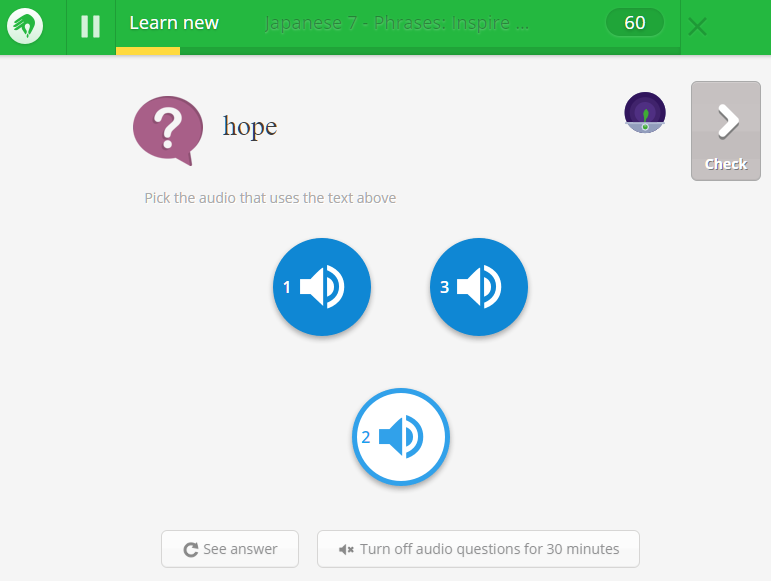


Figure 5. Example of an Audio review flashcard in Memrise.

Audio based flashcards is something to consider as this could be used to train the learners listening ability. In Memrise, they take the English translation and give 3 different audio clips of different words, one of the words being the Japanese translation. The user must select the right audio clip and click “Check” to see if they are correct.

The main issue that Memrise has is that it relies on English to teach learners. English and Japanese are languages that are very different and words in English can be translated to more than one word in Japanese that will have slightly different nuances and uses (e.g. “Hope” could be 希望, 見込み, 期待, 望み, 願う… etc.) and this could cause issues when trying to think of the correct answer as without context there can be multiple correct answers that may not be considered correct by Memrise. Instead of using translations, it might be best to have the learner acquire words by only using comprehensible sentences and without relying on English or another language. This also means that the application could be used by anyone of any language without having to translate the contents of flashcards, which would also save on costs.

There are also other types of flashcards that include having to type out the word, or sentence, based on the English translation with the letters on screen as well as listing to audio and then selecting the correct translation in English.

Memrise gives you points for learning and reviewing. By doing more activities on the site, you gain more points and go up the leader boards for different courses, putting you against other users. These points are also saved to your profile along with a rough estimation of the words that the system thinks you have learned. You also get a level, a “streak” which tracks daily goals and other learning stats.



Figure 6. Memrise’s brief stats overview for your profile.

Memrise also allows users to set daily goals, to focus on one skill e.g. Listening Skills as well as a variety of other useful looking tools such as access to native speakers, chat bots and grammar bots which are available in their paid version. Many of the features available in Memrise appear to be useful, however, further research into the effectiveness of each feature would have to be considered before implementing them into the artefact.

Memrise also claim that they make learning 3x more efficient by spacing study reminders (Memrise Science, n.d.). However, they allow learners to repeatedly study the same content despite these remainders so it’s possible for learners to go against the benefits of spaced repetition which may not be efficient in the long term. They also force learners to repeatedly review items just after they have learnt them. This is often in the format of about 2-3 minutes of constant testing of Japanese to English, and vice-versa, multi-answer, typing and listening questions which could feel tedious for the user.

**Quizlet**

Another service that appears to be popular is Quizlet (n.d.). Quizlet allows users to create study “sets”, or use other users sets, and study them in the form of flashcards. They also have a lot of other features such as being able to upload and share diagrams to study, using multiple choice answer questions, writing questions, spelling questions, tests and even games, with all these exercises revolving around sets of flashcards.

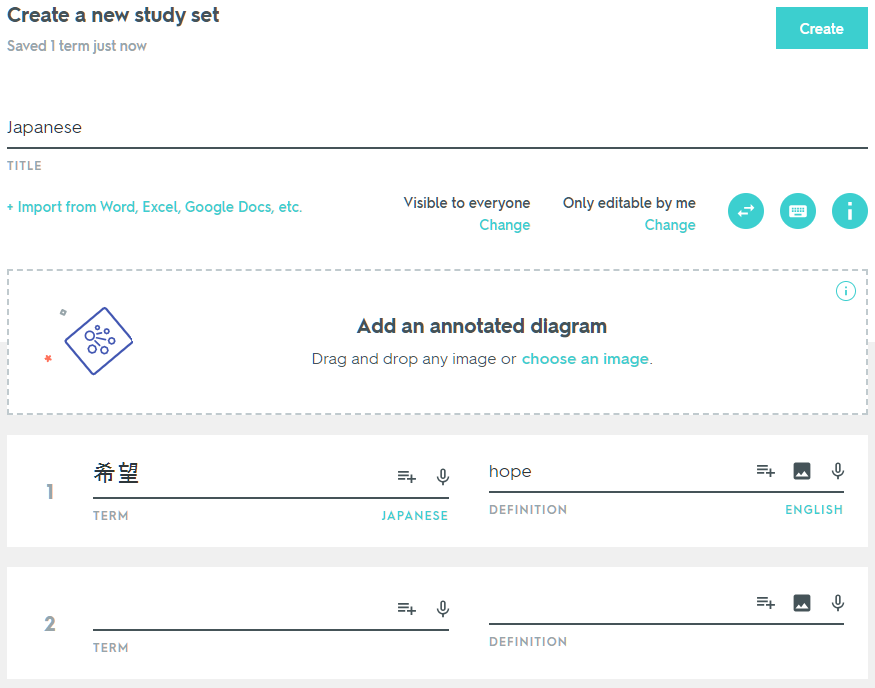
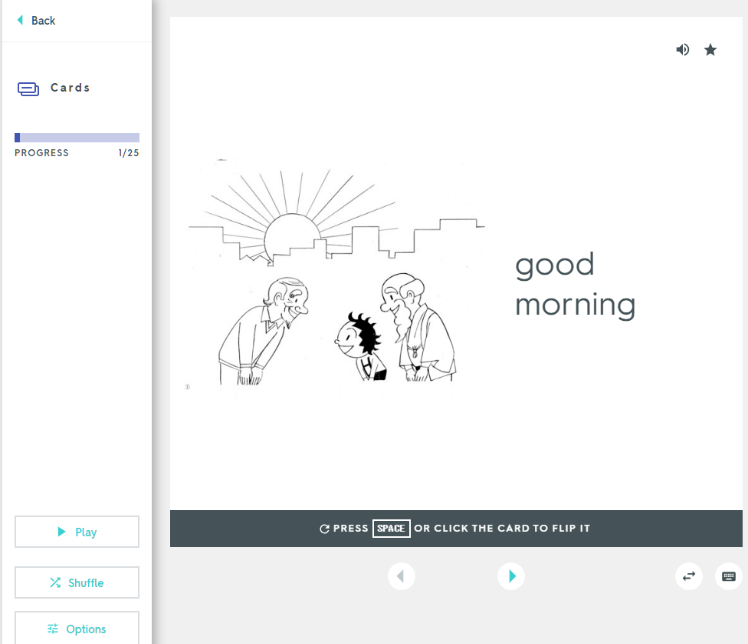
****

Figure 7. Quizlet’s “create a study set” page.

Sets consist of flashcards that have two sides, one being the term the other being the definition. If they are language cards you can choose which language the text is in although it’s unclear as to what functionality this adds to the cards. Each side can have images or audio on them as well. One of the things that really stands out as useful is the ability to import from Word, Excel, Google Docs and other programs, allowing students to potentially create sets straight from class notes and other teaching resources.

Figure 8. Front of a flashcard in Quizlet. 　　 Figure 9. Back of a flashcard in Quizlet.

Quizlet does use some form of spaced repetition for reviews but only in its paid for version, Quizlet Plus. According to their article, Studying with Long-Term Learning (n.d.) their Long-Term Learning is “based on the concept of spaced repetition”, with flashcards that the user doesn’t know well being showed up more often and well-known material not being shown to the user as often. However, how this is all done isn’t discussed in detail.

**Using Dictionaries**

Rahimi & Miri (2014, p. 1472) showed that students with access to a digital copy of a dictionary on their phones outperformed learners with a physical copy of the same dictionary. This study suggests that having easy access to a dictionary plays a vital role in a language learners success. The quick dictionary look-up Google Chrome extension rikaikun provides pop-up definitions for words and grammar and is very popular among Japanese learners with over 199,000 users, Speed (n.d.). A similar implementation of a hover over dictionary would be very useful for learners to have built-in to their language learning apps.

2.7 Conclusions and practical implications

Language acquisition relies on large amounts of input, often from an immersion environment, which exposes learners to plenty of “comprehensible input” which is essential for language acquisition to occur. Thus, the system to be developed should focus less on teaching learners about their L2 but instead provide a way of exposing the learner to i+1 comprehensible input to speed up language acquisition.

While a study that compares data of all the different learning strategies discussed in this literature review couldn’t be found, it appears that using comprehensible L2 sentence flashcards with spaced repetition is the most promising setup for learning vocabulary. When it comes to learning Kanji, it appears that flashcards based on mnemonics may also be advantageous when combined with another strategy to help learners cover all aspects of Kanji e.g. pronunciation. This strategy may work well with studying comprehensible L2 sentences flashcards given that the flashcards have native audio and furigana for teaching pronunciation but no evidence of such a strategy being used and being effective was found. Different layouts of these two types of flashcards need to be tested, however, and compared with results of other strategies.

The existing systems reviewed all had a variety of different features, some of the common ones being support for images and audio. They also all allowed users to create their own content as well as providing some way for users to share their created content with a community. It also appears that flashcard applications that use spaced repetition should allow users to adjust review settings and the underlying SRS algorithm to a certain extent to allow users to produce optimal results. The default settings for applications should also be considered by the developers to allow for efficient learning for even those users who do not wish to tweak settings. It also appears that there are many other possible features that may be useful for learners, e.g. built in dictionary look up, which can help learners reduce the time spent on studying.

1. Methodology

3.1 Development Methodology

Due to the nature of the end goal of the software, it will need to be tested for learning efficiency. This means that a development methodology that supports multiple iterations is required and thus the Spiral methodology was chosen. The Spiral methodology allows for quick production of a prototype and testing against itself and other systems to provide results which will create insights into further improvements.

3.2 Software and Technology Decisions

The software developed will be a web application that can be access on most mobile and desktop machines that have web browsers. There are a few reasons why this was decided over a standard mobile or desktop application. One of these reasons is to allow for scalability. In the future a desktop application may be developed along with a progressive web application to go on major mobile app stores. Progressive web applications are starting to take off and as a result the system will take into consideration the requirements needed to convert the system over to a progressive web application which could be put on mobile app stores with very little effort and would prevent the need for developing an entirely new mobile app for both major mobile operating systems, OS and Android. While at this stage the web application will not be a progressive web application, careful considerations will be taken during design to make it easier for scaling to the mobile app stores in the future. The web application should still be accessible by most commonly used mobile and desktop devices, changing in layout depending on screen size. The main reason the application won’t solely be developed as a standalone mobile application is because creating your own flashcards on a mobile device with a small screen is particularly difficult and frustrating.

Standard web technologies such as HTML, CSS, JavaScript, React and Node.js will be used to develop the system. SQL will also be used for database connection and querying. The reason for choosing these technologies is simply because the developer has experience using them in other projects and for gaining experience in JavaScript. While PHP, SQL, HTML and CSS may provide a basic application, JavaScript will be the main focus in order to allow greater functionality and a cleaner looking application.

3.3 Requirements Specification

When it came to the elicitation of requirements the method chosen was to use the observations made from research papers gathered for the literature review as well as the reviews of similar software. These observations were limited by whether they fit the scale of this project, whether they would benefit the learner in terms of learning and remembering new information and difficulty of implementation. Current software on the market had many pieces of functionality that would fit the nature of this application while other features also proved irrelevant due to several factors, namely due to research from the literature review showing that such features were obsolete. Some useful features from current software and the research are to be adopted and changed to fit the aim of the application too, most notably the idea of a “spacing effect” to help reduce the amount of work a user must do while also getting better results as well greater focus on giving the user context clues by using imagery and full sentences. Not everything mentioned in each of these sections can be included due to time, money and technical constraints and reasons for including and not including certain requirements will be discussed in an evaluation section further into this document.

The aim of these requirements is to define a foundation of features and components that are required to allow the application to succeed in its objective where other applications are lacking. The key feature here being the implementation of a spacing effect being combined with other small features to create an overall smooth and painless learning experience.

While language learning enthusiasts aren’t difficult to find, the time and energy spent on materials to educate potential end users on the concepts discussed in the literature review would take away from needed planning and development time. As such the project developer (who is a language learning enthusiast) is to explicitly validate the requirements and confirm which are to be added to the specification. These requirements are split up into functional and non-functional requirements while also being organised by priority.

All the functional requirements are those requirements that can be achieved via objectives and can be measured to ensure that they have been met or not met.

The requirements can be found below, grouped by order of priority. The non-functional requirements are also displayed in this way to indicate how essential a requirement is to the success of the project.

Non-functional requirements are less specific and are aimed at providing an objectively “good” system.

The possible rankings for each requirement are Must have, Should have, Could have and Won’t have. The rankings from Must have and Should have are important where as Could have and Won’t have are to be seen to after the main functionalities are implemented. These extra requirements will likely not be implemented until after the application has reached its first public version and will likely be implemented in future updates after the finished product is completed.

Must have (functional)

1. Must simulate flashcards by providing a front and a back to cards.
2. Either side of a card must be able to support Japanese characters.
3. The back of flashcards must only be shown once the user indicates to and must be hidden by default.
4. Must allow users to create their own text-based flashcards by adding text to the front and back of cards.
5. Users must be able to edit and delete any flashcard.
6. Must include an implementation of spaced repetition to schedule flashcard reviews.
7. Must automatically backup the users card collection, including scheduling information.
8. Changes to cards and scheduling information must be stored on a database.

Must have (non-functional)

1. Must have a usable and intuitive user interface.
2. Backups of data must be easily accessible to the user.
3. The app must run smoothly through study sessions meaning that the user must not wait for the software to load new cards.

Should have (functional)

1. Deciding on a card’s next review date should be done automatically by the system without interfering with the user’s study session.
2. Furigana support for Japanese text on both the front and back of cards with the ability to turn furigana on or off.
3. Allow users to upload image files for use in cards.
4. A “highlight and search in dictionary” option for text on cards.
5. Allow users to organise flashcards such as moving them between decks and changing the scheduling of cards including changing a card’s review date.
6. Track and show performance such as cards learnt, cards reviewed, cards added, days studied, time of day studied, projected reviews, time spent studying.

Should have (non-functional)

1. Should load images quickly.
2. Highlight and search in dictionary functionality should up-to-date and official Japanese-Japanese, Japanese-English and Kanji dictionaries.
3. A dictionary search should be relatively quick.

Could have (functional)

1. Allow users to upload audio files for use in cards.
2. Support for adding audio to the front and back of a card including a button to play audio as well as it auto playing when the front or back of a card is shown.
3. Option to change user interface to Japanese.
4. Option to turn on furigana for Japanese user interface.
5. Settings to change the style of flashcards such as text font, size, colour, background colour, etc.
6. Send notifications to remind users to study based on their normal study time for that day.

Could have (non-functional)

1. Should load audio quickly.
2. Should change user interface language without restarting the software.
3. Notifications could be done via browser, desktop or email.
4. Colour scheme of the application should match that of the client’s site.

Won’t have (functional)

1. Option to change user interface to other languages besides English and Japanese.

Won’t have (non-functional)

1. Support for plugins and addon code.
2. Other card formats.

4 Design

4.1 Design Decisions

The design decisions outlined below have been chosen to hopefully improve the usability and user experience of the artefact and have been chosen based on assumptions, previous research from the literature review as well as further research on UI design.

**Minimal navigation**

Keeping navigation simple by having clear and easy to see buttons/instructions is a must and making sure that users can easily get to where they want to, thus preventing confusion. To implement this a combination of forward and reverse navigation as mentioned in Google’s Material Design’s Understanding Navigation (n.d.), will be ideal for example making use of buttons to allow users to access deeper content while allowing them to return to the previous part of the app easily with their browser’s back button or with buttons for the main parts of the application which can be displayed on every screen.

**Key shortcuts**

Having key shortcuts for commonly used actions allows users to use the artefact more efficiently. As existing applications use key shortcuts, they should be included if users are to be able to study as quickly as they can using already existing systems as mentioned in Analysing Existing Systems.

**Flashcard design**

Anki (n.d.) simulates flashcards by showing the user the front first and then having the user press a button to display the back. The application has to simulate this traditional flashcard layout and must not steer away from this, else risk confusing users.

**Responsive UI**

The artefact is to be built as a web application to be used on a variety of different devices and as such must be responsive in its user interface. Users accessing the website, whether on mobile or desktop browsers, should have the application fit their screen naturally without reducing usability. As discussed on Google’s Material Design’s, Responsive Layout Grid (n.d.), UI regions should be consistent across all devices when changing for different screen sizes.

4.2 Use Case Overview of the Application

Figure 10 shows the layout of the application and how cases interact with each other and the external components of the web application such as users and the database that stores user data and flashcard data.

To gain access to the system a “New User” must create an account by using “Register” which will create a user account in the database. A user can then “Sign in” which will always trigger two other cases, “Verify Password” and “Get Flashcard Data”. If an error occurs during “Sign in” then the “Display Login Error” case may be triggered.

The “Create Flashcard Deck” case can be used by a “User” and will then be saved to the “External Database”. From the same screen as “Create Flashcard Deck” the “User” may trigger “Add Cards” to add new flashcards to a newly created deck. “Study Cards” and “Browse Cards” are similar in that the “User” may also trigger “Add Cards, as well as “Edit Cards” and “Delete Cards”, from these two screens also. When “Study Cards” is triggered and a flashcard is learnt or reviewed by a “User” the “Schedule Review” case must trigger to decide when the learnt or reviewed card will next be displayed to the user, which will be decided by a modified version of the Leitner System and the SM2 algorithm.

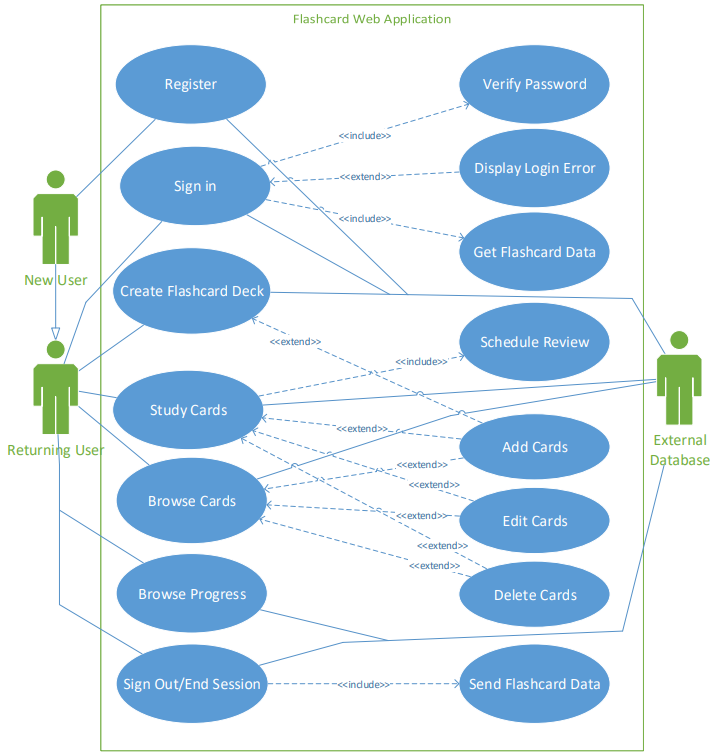


Figure 10. An overview of the application as a use case diagram.

“Browse Progress” can be accessed by the “User” which will show them usage data which will be saved to the “External Database” in relation to their user account.

“Sign Out/End Session” occurs when a user clicks a sign out button or the session ends due to the user not returning to the site within a certain time period. When this happens, the data that has been gathered in that session, such as scheduling, usage data, changes to flashcards and decks, will be sent to the “External Database” via “Send Flashcard Data”.

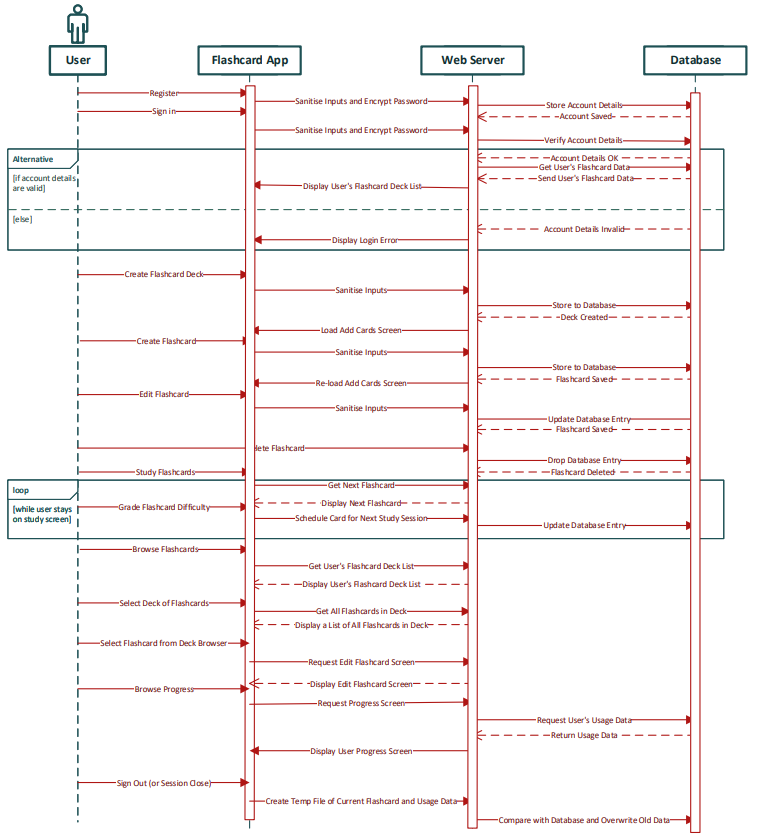


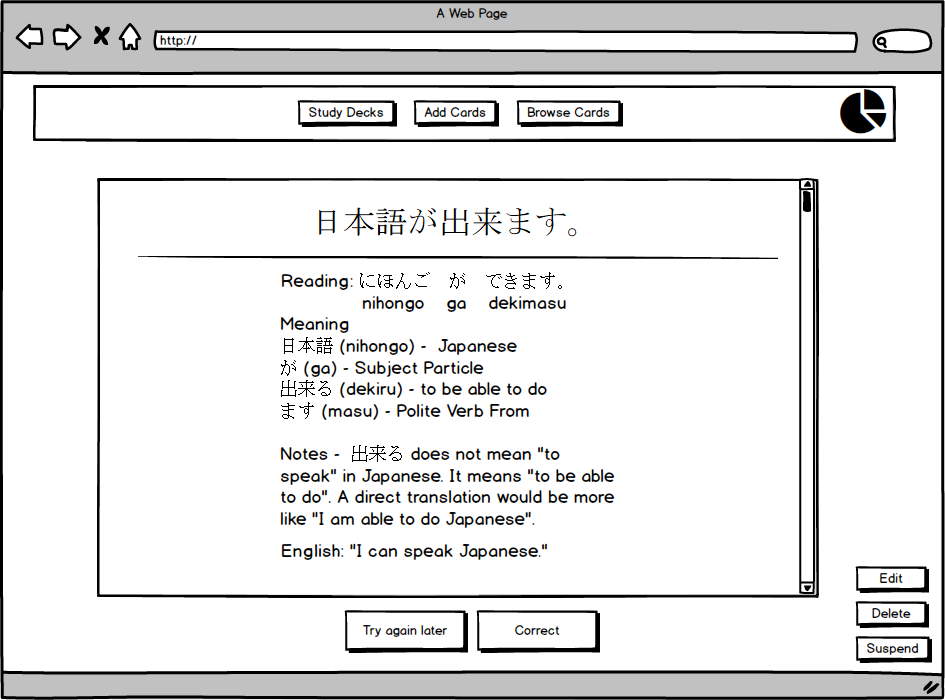
Figure 11. An overview of the application as a sequence diagram.

4.3 Low Fidelity Prototype

These 3 buttons are displayed when a user studies. Edit lets the user change the card, delete removes it from the database and suspend allows the user to remove the card temporarily.

Nav bar to be used across all screens of the app. This allows users to quickly access the most commonly used screens.

Balsamiq was used to design a wireframe which was then turned into a low fidelity prototype by adding links to buttons, allowing for a better understanding of how the application will be navigated. (See Appendix for working example).



Main flashcard layout. This is after the user has clicked the “Show Back” button.

The front is anything above the line and the back is below the line.

Answer buttons for user to grade themselves. This then determines how the card will be scheduled. E.g. Correct means the card won’t be shown for a while.

Figure 12. Low Fidelity Prototype for Desktop Browser (See Appendix)

It was necessary to consider the layout for mobile browsers too and thus a similar low fidelity prototype was created from a Balsamiq wireframe and can be seen in Figure 12 (See Appendix for working example).

The title of the app will be displayed here and like most other websites, if the logo/title is clicked then the user will be taken back to the main screen (deck list screen).

This button is used to navigate to the Progress and Usage screen.

Dropdown list for the user’s current decks. The user can select one of these decks and that decks cards will be displayed in the list further down the screen.

The user can also search their whole collection by not selecting a deck or they can search for cards inside a deck.

Selecting a card will allow the user to go to the Edit screen for that card, allowing them to edit, delete or suspend the card.

Figure 13. Low Fidelity Prototype for Mobile Browsers (See Appendix).

4.4 Database Design

For the first prototype of the system it has been decided that use of Google’s Firebase will be used because it’s quick to setup and allows the use of json which can be easily manipulated in JavaScript. However, Firebase will not be used for the final version of the application for several reasons which include storage limits, lack of knowledge when it comes to using multiple tables and because the client/developer of the artefact already has web hosting which has unused databases and storage available.

The first prototype of the system won’t yet have user accounts, so Firebase is fine to use for testing the saving of and reading of flashcard data. The layout of the json to be used will be a simple card object as shown below. At this stage this is more data storage via json than an actual database.

"cards": [

{

"id": 1,

"jp": "Japanese goes here",

"reading": "Reading goes here",

"meaning": "Meaning goes here.",

"en": "English goes here. "

},

]

The final version of the system will be using a MySQL 5.5 database provided by IONOS Hosting. The application should store three main pieces of data in this database; user accounts, flashcard information and progress/usage data. At this stage progress/usage data isn’t that important so this hasn’t been included in the following MySQL Model for the final version of the artefact.

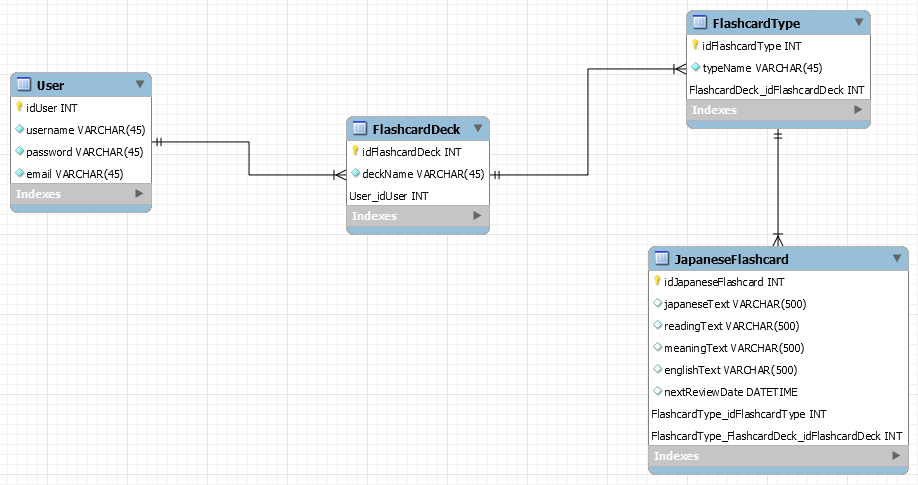


Figure 14. MySQL Model of database tables to be used in final version of the application.

The final version of the database will consist of at least the User, FlashcardDeck, FlashcardType and JapaneseFlashcard tables, each table having a 1 to many identifying relationship with each other in that respective order, i.e. User 🡪 1 to Many 🡪 FlashcardDeck, etc. This is because 1 user may have multiple flashcard decks which may have multiple flashcard types in it which may consist of multiple flashcards. The use of flashcard types may not yet be implemented into the system but the table has been added for scalability reasons as if other types of flashcards should be added it would mean having to redesign the database resulting in possible loss of user data.

On the second implementation the SQL database has been simplified temporarily due to implementation issues and complications. During development there were numerous bugs that where difficult to solve and while testing the “flashcard type” table was removed. As mentioned, for scalability purposes it’s good to have and as such will be added in a future iteration.

5.0 Implementation

Due to the nature of the project, it was thought best to take a spiral approach to implementation in order to deliver functioning prototypes soon for testing. While this did occur in a sense when it came to the low fidelity prototypes, the next high-fidelity prototype took longer to develop than expected. The original Gantt chart created in order to plan out the project aimed at getting the entirety of development completed and finished by 26/02/19, however this did not occur due to various reasons discussed below.

The developer decided in the beginning that sticking to the Gantt chart created at the beginning of the project would prove to be beneficial as it would save time as well as allow for any problems, or setbacks that may occur. Due to some setbacks which occurred, however, development soon got behind schedule.

The first high-fidelity prototype developed was a working application that worked with react.js, Firebase and some HTML and CSS in order to create a mini-flashcard application that would work for one user. No account creation was implemented and neither was the desired spaced repetition algorithm to schedule reviews of flashcards as they where studied. The issue with slow development arose because the developer was not confident in JavaScript and despite this decided it was best to use react.js and firebase to develop this application as it would develop his skills. This made implementation slow and difficult, which also pushed back other parts of the project as well as the final implementation, resulting in some functionality missing from the final application.

This effect was then echoed further when, as defined in the design section, the database of choice was switched from firebase to SQL. The reason for this change was that the developer had more knowledge with SQL than with firebase, and knew how to setup a SQL database for a multi-user application. This meant re-creating a large portion of the prototype in order to create the next iteration. This process took a long time despite the SQL database being created within a few minutes. The main issue was how to get the information to and from the application. At this stage the application didn’t really have a back end and just sent data to and from firebase from the front end, which was only done for the purpose of developing a quick prototype.

To solve this the developer had to create two separate servers to talk to each other, the back end and the front end of the application, all while having the back end communicate with the SQL server hosted on a local phpMyAdmin server. This was something the developer was not expecting to be that difficult due to his previous experience with PHP and while the methods of sending and getting data are essentially very similar to PHP, it was hard to get everything connected due to the unfamiliarity of react and node.js. It was much harder than expected which extended development time.

Creating a RESTful api for the back-end web server was something that took sometime to figure out as well. The api was needed to get and post data from and to the database. There are many different ways of going about it and it was difficult to find resources to learn how to do this with an SQL database. It took around two weeks to create a back-end server with a RESTful api using the Express module could communicate with both a SQL database and the front-end JavaScript of the application.

The IDE that was mainly used throughout this project was Visual Studio Code and while it was a very useful tool during the development of this project as it allowed the creation of terminals within VSC itself, as it’s not a language specific IDE its bug reporting wasn’t very useful. Often bugs where traced from compile errors in the browser or via Google Chrome’s console feature. When moving from the firebase database to the SQL database, due to a restructure of the entire application, many different issues arose. One particular bug occurred when attempting to refactor the logic behind getting flashcards from the database to display on screen during a study session. This particular bug was incredibly hard to fix as no actual compilation error would occur because of it. After using Chrome’s console to track down whether exactly the front end was getting flashcards from the server or not, it was discovered that while the logic was correct, due to forgetting to change .jp

jp={this.state.currentCard.jp}]

jp={this.state.currentCard.japaneseText}

to .japaneseText to match the new database column name. As this code is rendering a flashcard using data in the form of JSON from the SQL database, and because there were no errors in the rest of my code, finding this minute mistake took almost a week to fix and drastically set back development time very closely to the development date which meant that adding in any new functionality wasn’t possible due to the lack of time. Better debugging awareness would have solved this issue way sooner

GitHub was used throughout the implementation to store and track changes, however some issues where encountered with the GitHub Desktop software which prevented it from being used often and as such it was not used as much as it should have been. Version control and tracking changes is an essential part of making sure a project is maintainable, and reduces risk in case anything where to happen to the source code, as it allows code to be simply rolled back to a previous state.

6.0 Testing and Evaluation

This section aims to compare and evaluate how successful the project was at meeting its aims by using analysis from the client and end users. Section 6.1 looks at User Interface design before implementation which allowed better insight into layout while section 6.2 looks at the evaluation of the current system against the original functional and non-functional requirements.

6.1 Low Fidelity Navigational and User Interface Testing

Balsamiq was used for the low fidelity prototypes of the application which proved to be beneficial as balsamiq allows the creation of links between different views, making it interactive and thus allowing developers to test a User Interface before implementing it in a real system. This allowed for early navigation testing to ensure that navigation would be simple for users.

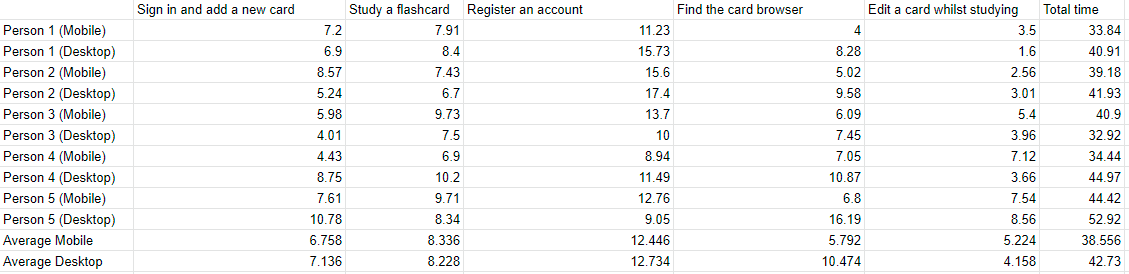
In order to ensure that navigation was something that was right from the beginning, early user testing was conducted with 5 anonymous users as seen below. Each user was given a task to complete on both mobile and desktop and their time was recorded in seconds 

Figure 15. (See Appendix Table 3 for a bigger table).

Some observations made during the test where:

* 1 person was confused on the pc version when buttons where to be selected in the navigation bar, they assumed that this meant what page they were on
* 1 person struggled to find the settings page
* 2 people said that the example flashcard had too much text on it and that maybe a scroll bar would be good so the information can be spaced out better

This proved useful as it highlighted some issues to be aware during actual implementation and showed that research into user interface design would be desirable when the next implementation occurs.

The last point was something that had to be changed straight away and as such the design documentation was changed to add a scroll bar for the flashcard component. This simply because without a scroll bar, some of the content will not be viewable which is not acceptable.

The data showed that registering took the most amount of time out of all the tasks. This is likely to be due to the fact that users have to type out information instead of just clicking through different views and therefore will take longer than other tasks.

Finding the card browser took longer than expected as it is a relatively large button at the top of the page, however this may mean that a combination of the colour scheme and size maybe causing an issue.

6.2 Client Evaluation of Current System

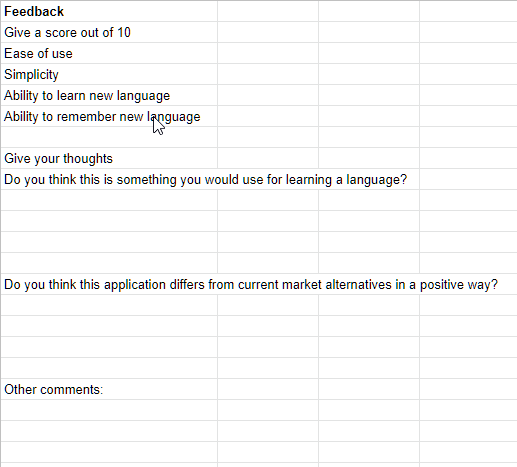
A questionnaire was drawn up in order to obtain feedback on the system, allowing for the ability to determine whether the current system meets its aims. This questionnaire was given out to individuals after the current implementation of the system was developed. The questionnaire uses a combination of questions with a 0-10 scoring (10 being high and 0 being low), and longer answer questions.

Figure 16. Questionnaire give to users for evaluation

A total of 8 users filled out the questionnaire as seen below, however not all of them answered all the questions.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Ease of use | Simplicity | Ability to learn new language | Ability to remember new language | Do you think this is something you would use for learning a language? | Do you think this application differs from current market alternatives in a positive way? | Other comments |
| 7 | 9 | 5 | 2 | Possibly | The maker mentioned the idea of spacing but said that it wasn’t implemented yet, so I feel like this is just a simple flashcard app that is not much different to other apps in the genre | I think with a few extra features this could be a really useful application |
| 6 | 8 | 4 | 3 | There are other applications that seem better so no | Not really, other app seem better with more features |  |
| 8 | 8 | 6 | 1 | No | It’s easy to use but other than that not really | n/a |
| 9 | 7 | 7 | 4 | Maybe | It has potential to be but right now no | n/a |
| 8 | 9 | 5 | 4 | Yes | It’s really simple and easy to use which is great. Lots of apps have far to many features and confuse the user so keeping it to the bare essentials is good. | If all the janky bits are fine tuned and tweaked I would use this because of its sheer simplicity. |
| 7 | 6 | 4 | 2 | Not sure | No | n/a |
| 6 | 7 | 5 | 3 | If I had to switch from what I am using now then maybe | It’s simplistic which I love as it means I can get straight into studying, not many apps are like this. | n/a |
| 8 | 7 | 5 | 5 | Yes, I would use as I like flashcards for learning languages | It keeps the learning process simple which is good | It seems like it still needs work, maybe add some features for helping people look words or something |

Figure 17. Questionnaire results

While this feedback isn’t entirely positive, it is very useful in given insight into how users feel about the application. Overall users found it easy to use but also showed concern due to the lack of features in comparison to other applications, however, other users like this simplicity as it meant it was easier for them to focus on their studying. This shows that Must have non-functional requirements 1. and 3. were met but it also shows that other requirements were not due to the lack of features that had not yet been implemented.

6.2.1 Results and Requirements Comparison

Here the requirements are displayed once again in figure 18, however this time showing whether the requirement was met or not. This show with a simple green MET, a red NOT MET or a red N/A after each requirement. N/A is used in situations where the requirement is based off another requirement that hasn’t been met yet therefore makes the requirement redundant. Only Must have and Should have requirements have been included in this comparison as the Could have and Won’t have requirements are “extra” requirements intended to be implemented further down the road after the application has been released onto the market.

Must have (functional)

1. Must simulate flashcards by providing a front and a back to cards. **MET**
2. Either side of a card must be able to support Japanese characters. **MET**
3. The back of flashcards must only be shown once the user indicates to and must be hidden by default. **MET**
4. Must allow users to create their own text-based flashcards by adding text to the front and back of cards. **MET**
5. Users must be able to edit and delete any flashcard. **NOT** **MET**
6. Must include an implementation of spaced repetition to schedule flashcard reviews. **NOT** **MET**
7. Must automatically backup the users card collection, including scheduling information. **MET**
8. Changes to cards and scheduling information must be stored on a database. **MET**

Must have (non-functional)

1. Must have a usable and intuitive user interface. **MET**
2. Backups of data must be easily accessible to the user. **NOT** **MET**
3. The app must run smoothly through study sessions meaning that the user must not wait for the software to load new cards. **MET**

Should have (functional)

1. Deciding on a card’s next review date should be done automatically by the system without interfering with the user’s study session. **MET**
2. Furigana support for Japanese text on both the front and back of cards with the ability to turn furigana on or off. **MET**
3. Allow users to upload image files for use in cards. **NOT** **MET**
4. A “highlight and search in dictionary” option for text on cards. **NOT** **MET**
5. Allow users to organise flashcards such as moving them between decks and changing the scheduling of cards including changing a card’s review date. **NOT** **MET**
6. Track and show performance such as cards learnt, cards reviewed, cards added, days studied, time of day studied, projected reviews, time spent studying. **NOT** **MET**

Should have (non-functional)

1. Should load images quickly. N/A
2. Highlight and search in dictionary functionality should up-to-date and official Japanese-Japanese, Japanese-English and Kanji dictionaries. N/A
3. A dictionary search should be relatively quick. N/A

The majority of Must have requirements have been met as well as a Should have requirement as well. However, nearly all the Should have requirements where unable to be completed due to time constraints. These requirements are not essential to the overall functionality of the project and as such where decided to be less important than those of the Must have requirements, but these requirements will need to be added into the next iteration of the application in order to bring a smoother experience for learners.

For each requirement Must have and Should have requirement you can find two tables below describing each requirement as well as an analysis of the outcome of each requirement after development. There are two separate tables for functional and non-functional and as mentioned briefly earlier, Could have and Won’t have requirements are not included in this section as those requirements are intended for future versions of the system and not for the initial release. Green indicates that the requirement was met to an acceptable standard and Orange indicates that the requirement was not met.

Functional

|  |  |  |
| --- | --- | --- |
| Requirement | Description | Analysis |
| Must have 1. | Simulate flashcards by providing a front (e.g. question, something that is shown to the user as a hint) and a back (something hidden until the user requests it to be shown). | Achieved. |
| Must have 2. | Flashcards must support Japanese characters. | Achieved. |
| Must have 3. | Back of flashcard must only be shown after user indicates via an action e.g. button click or hover over. | Achieved. |
| Must have 4. | Users can create their own text-based flashcards by creating a flashcard that has a front and a back. | Achieved. |
| Must have 5. | Users can edit/delete flashcards | This functional requirement was not quite met in time. The back-end code was finished for deletion but never implemented on the front end while editing hadn’t been implemented at all. This was mainly due to time constraints caused by issues in development as mention in the Implementation section of this report. |
| Must have 6. | The application schedules flashcards by spacing them out based on a spaced-repetition algorithm. | While the basis of this was implemented such as the dates for last studied being stored in the database, and only allowing cards that are ready to be learnt to be shown on or after the correct date, the actual scheduling has yet to be implemented. This was due to a lack of time due to the major issues in development as discussed in the Implementation section. |
| Must have 7. | A user’s card data must be backed up, this includes scheduling information. | Achieved. |
| Must have 8. | Changes to cards and scheduling information must be stored on a database. | Achieved. |
| Should have 1. | Deciding on a card’s next review date should be done automatically by the system without interfering with the user’s study session. | Achieved |
| Should have 2. | Support for Furigana (how Japanese is pronounced) on the flashcards | Achieved, however not automatic. Could be improved later to make it easier when making a card. |
| Should have 3. | Allow images to be used in flashcards | Due to difficulties and possible security risks this was dropped for the time being. Images are an important part of the system and will need to be included in a later date. |
| Should have 4. | Highlight and search word in dictionary functionality. | Most browsers allow “Search on web” features when you highlight a word and as such this requirement was less important when other major issues that were preventing the completion of the project and as such was left out. |
| Should have 5. | Allow users to organise flashcards e.g. moving them between different decks, adjusting a card’s scheduling information etc. | The back bones of this is in place with the structure of the database, however not all of the backend code was finalised in time to warrant giving user’s access to this section as it wouldn’t entirely work. |
| Should have 6. | Performance tracking and displaying of cards, time studied etc. | Due to time constraints and the fact that more important requirements had yet to be completed, performance tracking was set aside for the time being. |

Non-Functional

|  |  |  |
| --- | --- | --- |
| Requirement | Description | Analysis |
| Must have 1. | User interface is intuitive and easy to use | Achieved based on responses from the client questionnaire in Figure 16 and 17. |
| Must have 2. | Backups are easily accessible to the end user | This requirement was not yet achieved as currently backups are just of the database and its structure. A way of getting such backup to the user while only showing the user their data has yet to be worked out. |
| Must have 3. | The application runs smoothly with no load times and no major interruptions for the user. | Achieved. |
| Should have 1. | Image loading should be fast. | Not applicable as image support was not implemented. |
| Should have 2. | Highlight and search in dictionary functionality should up-to-date and official Japanese-Japanese, Japanese-English and Kanji dictionaries. | Not applicable as a dictionary search function was not implemented. |
| Should have 3. | Dictionary lookup should be fast. | Not applicable as a dictionary search function was not implemented. |

For the most part any unmet requirements where down to major implementation issues that caused the project to fall far behind schedule, which meant that Must have requirements where made more important and focused on first and foremost in order to get the basic functionality of the application up and running in time for release. Any unmet Must have and Should have requirements that have not been met at this stage must be fixed and implemented correctly ready for the next version, and this will require good planning and project management.

7.0 Project Management

As mentioned in the Implementation chapter, throughout the project there where a variety of difficulties that where much more than mere challenges for the developer. A lot of these caused major setbacks in development which then had a knock-on effect on the overall outcome of the actual application.

Even before development started, some lapse from the original schedule had already occurred. This seemed to be due to having a schedule that was a bit too optimistic, which was in one aspect a good thing as it did keep motivation levels high which lead to more work output, but when creating the original plan there was no consideration for other ongoing projects, external work, personal issues etc. When considering a plan for such a large project, even smaller things, such as illness, should be taken into consideration to allow a bit of leeway if anything does go wrong, to allow the project to get back on track.

8. Conclusion

Overall, while the application didn’t meet all the requirements laid out in 3.3 Requirements Specification, a lot was learnt about creating a full stack web application. Many technologies where used to create the final application and many lessons where learnt throughout the project.

8.1 Future Development

The application is currently at a good standing point and can act as a good base for reaching the rest of the aims for the project as well as for completing the other Requirements laid out in 3.3 Requirements Specification. It can easily be scaled up to any size and with a few tweaks here and there could be ready for deployment within the next few months.

9. References

Anki - powerful, intelligent flashcards. (n.d.). Retrieved 2 November 2018, from https://apps.ankiweb.net/

Anki Manual. (n.d.). Retrieved 14 November 2018, from https://apps.ankiweb.net/docs/manual.html#what-spaced-repetition-algorithm-does-anki-use

Atkinson, R. C. (1972). Optimizing the learning of a second-language vocabulary. *Journal of Experimental Psychology*, *96*(1), 124–129. https://doi.org/10.1037/h0033475

Ausubel, D. P., & Youssef, M. (1965). The effect of spaced repetition on meaningful retention. *Journal of General Psychology; Provincetown, Mass., Etc.*, *73*, 147–150.

Barcroft, J. (2007). Effects of Opportunities for Word Retrieval During Second Language Vocabulary Learning. *Language Learning*, *57*(1), 35–56. <https://doi.org/10.1111/j.1467-9922.2007.00398.x>

Beheydt, L. (1987). The semantization of vocabulary in foreign language learning. *System*, 15, 55-67.

Bhide, A. (2015). Early literacy experiences constrain L1 and L2 reading procedures. *Frontiers in Psychology*, *6*. https://doi.org/10.3389/fpsyg.2015.01446

Blocki, J., Komanduri, S., Cranor, L., & Datta, A. (2015). Spaced Repetition and Mnemonics Enable Recall of Multiple Strong Passwords. *Proceedings 2015 Network and Distributed System Security Symposium*. https://doi.org/10.14722/ndss.2015.23094

Bloom, K. C., & Shuell, T. J. (1981). Effects of Massed and Distributed Practice on the Learning and Retention of Second-Language Vocabulary. *The Journal of Educational Research*, *74*(4), 245–248.

Bower, J. V., & Rutson-Griffiths, A. (2016). The relationship between the use of spaced repetition software with a TOEIC word list and TOEIC score gains. *Computer Assisted Language Learning*, *29*(7), 1238–1248. https://doi.org/10.1080/09588221.2016.1222444

Card-based design combined with spaced repetition: A new interface for displaying learning elements and improving active recall. (n.d.).

Chukharev-Hudilainen, E., & Klepikova, T. A. (2014). The effectiveness of computer-based spaced repetition in foreign language vocabulary instruction: a double-blind study. *CALICO Journal*, *0*(0). https://doi.org/10.1558/cj.v33i2.26055

Cohen, A. D. (1987). The Use of Verbal and Imagery Mnemonics in Second-Language Vocabulary Learning. *Studies in Second Language Acquisition*, *9*(01), 43. <https://doi.org/10.1017/S0272263100006501>

Cummins, J (2000). Immersion Education for the Millennium: What We Have Learned from 30 Years of Research on Second Language Immersion. *University of Toronto*. Retrieved 31 October 2018, from <https://www.researchgate.net/publication/255638397_Immersion_Education_for_the_Millennium_What_We_Have_Learned_from_30_Years_of_Research_on_Second_Language_Immersion>

Dempster, F. N. (1988). A Case Study in the Failure to Apply the Results of Psychological Research. *American Psychologist*, 8.

Dolch, E. W. (1949). The Use of Vocabulary Lists in Predicting Readability and in Developing Reading Materials. *Elementary English*, *26*(3), 142–177.

Ebbinghaus, H. E. (1964). Memory: A contribution to experimental psychology (Henry A. Ruger & Clara E. Bussenius, Trans.). New York: Dover. (Original work was published in 1885).

Fernández, C. (2011). Input Matters in SLA edited by PISKE, THORSTEN, & MARTHA YOUNG–SCHOLTEN. *The Modern Language Journal*, *95*(4), 673–674. https://doi.org/10.1111/j.1540-4781.2011.01258.x

Freed, B. F., Segalowitz, N., & Dewey, D. P. (2004). CONTEXT OF LEARNING AND SECOND LANGUAGE FLUENCY IN FRENCH: Comparing Regular Classroom, Study Abroad, and Intensive Domestic Immersion Programs. *Studies in Second Language Acquisition*, *26*(2), 275–301. https://doi.org/10.1017/S0272263104262064

Godwin-Jones, R. (2010). Emerging technologies from memory palaces to spacing algorithms: approaches to second language vocabulary learning. *Language, Learning & Technology, 14, 2,* 4-11.

Godwin-Jones, R. (2011). Mobile Apps for Language Learning. *Language Learning & Technology, 15,* 2-11.

Gu, Y., & Johnson, R. K. (1996). Vocabulary Learning Strategies and Language Learning Outcomes. *Language Learning*, *46*(4), 643–679. https://doi.org/10.1111/j.1467-1770.1996.tb01355.x

Hafiz, F. M. (1989). Extensive reading and the development of language skills. *ELT Journal*, *43*(1), 4–13. https://doi.org/10.1093/elt/43.1.4

Heisig, J. W. (2001). *Remembering the kanji*. Tokyo: Japan Publications Trading Co.

Japanese Support - AnkiWeb. (n.d.). Retrieved 14 November 2018, from <https://ankiweb.net/shared/info/3918629684>

Kang, S. H. K. (2016). Spaced Repetition Promotes Efficient and Effective Learning: Policy Implications for Instruction. *Policy Insights from the Behavioral and Brain Sciences*, *3*(1), 12–19. <https://doi.org/10.1177/2372732215624708>

Karpicke & Roediger. (2008). The Critical Importance of Retrieval for Learning. *Science, 319, 5865,* 966-968. Retrieved 13 November 2018, from: http://science.sciencemag.org/content/319/5865/966?casa\_token=ACJqFC7dCbAAAAAA%3AWh93XPkSNuywgywQmgGX4xmZlSkOVgqfRhfLa8OsS5dEBKE8rydll-8bp0qiUm0IzJe4UtJwIi4u1hGc

Kornell, N. (2009). Optimising learning using flashcards: Spacing is more effective than cramming. *Applied Cognitive Psychology*, *23*(9), 1297–1317. https://doi.org/10.1002/acp.1537

Krashen, S. (1989). We Acquire Vocabulary and Spelling by Reading: Additional Evidence for the Input Hypothesis. *The Modern Language Journal*, *73*(4), 440–464. https://doi.org/10.2307/326879

Krashen, S. D. (1982). *Principles and practice in second language acquisition* (1st ed). Oxford ; New York: Pergamon.

Laufer B. (1989). What percentage of text-lexis is essential for comprehension? *Special Language: From Humans Thinking To Thinking Machines,* 316-323. Retrieved 14 November 2018, from <https://www.researchgate.net/publication/248424656_What_percentage_of_text-lexis_is_essential_for_comprehension>

Learn fast and forget about forgetting | SuperMemo.com. (n.d.). Retrieved 11 November 2018, from <https://www.supermemo.com/>

Long, M. H. (1983). *Native Speaker/Non-Native Speaker Conversation in the Second Language Classroom* (Working Paper). Retrieved from <http://scholarspace.manoa.hawaii.edu/handle/10125/38598>

Meara, P. (1980). Vocabulary Acquisition: A Neglected Aspect of Language Learning. *Language Teaching*, *13*(3–4), 221. https://doi.org/10.1017/S0261444800008879

Melton, A. W. (1970). The situation with respect to the spacing of repetitions and memory. *Journal of Verbal Learning and Verbal Behavior*, *9*(5), 596–606. <https://doi.org/10.1016/S0022-5371(70)80107-4>

Morohashi, T. (2000). Dai Kan-Wa Jiten (15th ed.) [*The Great Chinese-Japanese Dictionary*]. Tokyo: Taishukan.

Nakata, T. (2011). Computer-assisted second language vocabulary learning in a paired-associate paradigm: a critical investigation of flashcard software. *Computer Assisted Language Learning*, *24*(1), 17–38. https://doi.org/10.1080/09588221.2010.520675

Nicholas J. Cepeda, Noriko Coburn, Doug Rohrer, John T. Wixted, Michael C. Mozer, and Harold Pashler. (2009). Optimizing Distributed Practice Theoretical Analysis and Practical Implications. *Experimental Psychology*, *56*(4), 236–246. https://doi.org/10.1027/1618-3169.56.4.236

Pavlik, P. I., & Anderson, J. R. (2005). Practice and Forgetting Effects on Vocabulary Memory: An Activation-Based Model of the Spacing Effect. *Cognitive Science*, *29*(4), 559–586. https://doi.org/10.1207/s15516709cog0000\_14

Pavlik, P. I., & Anderson, J. R. (2008). Using a model to compute the optimal schedule of practice. *Journal of Experimental Psychology: Applied*, 101–117.

Phillips, J. (2011). An Investigation into The Effect of Targeted Vocabulary Learning Using a Spaced Repetition Flashcard System on TOEIC Scores.

*The Aoyama Gakuin University & Women’s Junior College Bulletin* *65*, 55-62. Retrieved from <https://www.agulin.aoyama.ac.jp/opac/repository/1000/12507/>

Ponniah, R. J. (2011). Incidental Acquisition of Vocabulary by Reading. *The Reading Matrix*, *11*(2), 5.

Rahimi, M., & Miri, S. S. (2014). The Impact of Mobile Dictionary Use on Language Learning. *Procedia - Social and Behavioral Sciences*, *98*, 1469–1474. https://doi.org/10.1016/j.sbspro.2014.03.567

Reddy, S., Labutov, I., Banerjee, S., & Joachims, T. (2016). Unbounded Human Learning: Optimal Scheduling for Spaced Repetition. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD ’16*, 1815–1824. https://doi.org/10.1145/2939672.2939850

Rukmini, S. (2017). Content-Based Language Learning and Communicative Approach to English Language Teaching for Technology and Management Courses: Integration and Implications. *IUP Journal of English Studies*, *12*(4), 84–88.

Seabrook, R., Brown, G. D. A., & Solity, J. E. (2005). Distributed and massed practice: from laboratory to classroom. *Applied Cognitive Psychology*, *19*(1), 107–122. https://doi.org/10.1002/acp.1066

Sentence items (cards) in SRS | Antimoon. (n.d.). Retrieved 11 November 2018, from http://www.antimoon.com/how/usingsm-makeitems-sentence.htm

Shared Decks - AnkiWeb. (n.d.). Retrieved 15 November 2018, from https://ankiweb.net/shared/decks/

Shimizu, H., & Green, K. E. (2002). Japanese Language Educators’ Strategies for and Attitudes toward Teaching Kanji. *The Modern Language Journal*, *86*(2), 227–241. <https://doi.org/10.1111/1540-4781.00146>

Speed, E. (n.d.). *rikaikun – Google Chrome web extension.* Retrieved 1st November 2018, from <https://chrome.google.com/webstore/detail/rikaikun/jipdnfibhldikgcjhfnomkfpcebammhp>

Staub, A., Grant, M., Astheimer, L., & Cohen, A. (2015). The influence of cloze probability and item constraint on cloze task response time. *Journal of Memory and Language*, *82*, 1–17. https://doi.org/10.1016/j.jml.2015.02.004

The Semantization of Vocabulary in Foreign Language Learning | Request PDF. (n.d.). Retrieved 1 November 2018, from https://www.researchgate.net/publication/223293827\_The\_Semantization\_of\_Vocabulary\_in\_Foreign\_Language\_Learning

The World Factbook — Central Intelligence Agency. (n.d.). Retrieved 31 October 2018, from https://www.cia.gov/library/publications/the-world-factbook/geos/ca.html

Vanpatten, B., & Cadierno, T. (1993). Input Processing and Second Language Acquisition: A Role for Instruction. *The Modern Language Journal*, *77*(1), 45–57. https://doi.org/10.1111/j.1540-4781.1993.tb01944.x

10. Appendix

**Table 1**

|  |  |
| --- | --- |
| Search result | Software mentioned |
| <https://qz.com/1211561/how-to-learn-a-language-use-spaced-repetition/> | Anki, Tinycards, Memrise, iKnow and Pleco |
| <https://www.fluentin3months.com/spaced-repetition/> | Anki |
| <https://www.fluentu.com/blog/best-foreign-language-learning-flashcards-apps/> | FluentU, Anki, Brainscape, Voxy, British Council, StudyBlue |
| <https://www.fluentu.com/blog/srs-spaced-repetition-language-learning/> | Anki, FluentU, SuperMemo |
| <https://en.softonic.com/solutions/what-are-the-best-spaced-repetition-apps> | Quizlet, Memrise, AnkiDroid, VocApp |
| <https://www.iwillteachyoualanguage.com/blog/app-review-flashcards-the-best-app-for-learning-vocabulary> | Flashcards Deluxe, Quizlet, FlashcardExchange (Cram) |
| <https://fluent-forever.com/the-method/spaced-repetition/> | Anki, Anki for OS, AnkiDroid |
| <https://nihongoperapera.com/flashcards-insufficient.html> | *n/a* |
| <https://www.ankiapp.com/> | AnkiApp |

**Table 2**

|  |  |
| --- | --- |
| Software | Total mentions from Table 1 |
| Anki | 5 |
| AnkiApp (not associated with Anki) | 1 |
| AnkiDroid (Anki for Android) | 2 |
| Anki for OS | 1 |
| Brainscape | 1 |
| British Council | 1 |
| Cram (FlashcardExchange) | 1 |
| Flashcards Deluxe | 1 |
| FluentU | 2 |
| Memrise | 2 |
| Pleco | 1 |
| Quzilet | 2 |
| StudyBlue | 1 |
| SuperMemo | 1 |
| TinyCards | 1 |
| VocApp | 1 |
| Voxy | 1 |

Low fidelity prototype (download and view in a PDF reader to use navigation): <https://github.com/Matchoo95/Final-Year-Project-UP769535-Matthew-Hawkins/blob/master/Design/Prototypes/Low%20Fidelity/Low%20Fidelity%20Prototype%20PDF.pdf>

Table 3

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Sign in and add a new card | Study a flashcard | Register an account | Find the card browser | Edit a card whilst studying | Total time (s) |
| Person 1 (Mobile) | 7.2 | 7.91 | 11.23 | 4 | 3.5 | 33.84 |
| Person 1 (Desktop) | 6.9 | 8.4 | 15.73 | 8.28 | 1.6 | 40.91 |
| Person 2 (Mobile) | 8.57 | 7.43 | 15.6 | 5.02 | 2.56 | 39.18 |
| Person 2 (Desktop) | 5.24 | 6.7 | 17.4 | 9.58 | 3.01 | 41.93 |
| Person 3 (Mobile) | 5.98 | 9.73 | 13.7 | 6.09 | 5.4 | 40.9 |
| Person 3 (Desktop) | 4.01 | 7.5 | 10 | 7.45 | 3.96 | 32.92 |
| Person 4 (Mobile) | 4.43 | 6.9 | 8.94 | 7.05 | 7.12 | 34.44 |
| Person 4 (Desktop) | 8.75 | 10.2 | 11.49 | 10.87 | 3.66 | 44.97 |
| Person 5 (Mobile) | 7.61 | 9.71 | 12.76 | 6.8 | 7.54 | 44.42 |
| Person 5 (Desktop) | 10.78 | 8.34 | 9.05 | 16.19 | 8.56 | 52.92 |
| Average Mobile | 6.758 | 8.336 | 12.446 | 5.792 | 5.224 | 38.556 |
| Average Desktop | 7.136 | 8.228 | 12.734 | 10.474 | 4.158 | 42.73 |