**Getting Usability into Development Environments Report**

**Purpose of the System**

The domain for this application is language learning. The aim is that users will be able to learn languages contextually via geocaching and memory techniques with flashcards. To make the process of language learning more efficient, the app will allow users to unlock vocabulary in their target language based on locations they visit. This idea allows the user to more efficiently learn the words that they will regularly come across in daily life.

The user will also gain experience points for each place they visit, encouraging them to learn more if they want to and therefore visit more places and gain more caches. As a user studies, they will also gain experience points so that users who study more gain access to more content. Rarer locations will contain harder words that will be level locked. Users who have gained enough experience points can then unlock them.

The overall purpose for the user is to be able to learn languages in an efficient yet fun way, as if the entire learning process was a game.

Within the app they should be able to:

* Create an account and sign in.
* Use a map to find geocached vocabulary flashcard decks.
* Learn and review decks of flashcards.
* Skip flashcards due to difficulty.
* View progress, level and experience points.

The target users could be anyone who wants to learn a language and as such could be of any age. For usability testing purposes people between the ages of 18-50 will be selected for prototype testing.

**How Caching is used in this System**

The app will display a list of locations, e.g. “Train station” or “University”. The user then clicks on one and a list of locations will appear with coordinates and the distance from the user. If the user clicks on one of these options, more information, e.g. a map, will appear. The user then heads to the location using the information supplied by the app. When they enter the radius of the geocache, a notification will appear on their phone and tell them that they can gain a new set of cards and some experience points. If they select “yes” to this, then the user’s profile will get updated and synced to the cloud. Users can also sync their data themselves via a button.

**Usability Concepts in Relation to the System**

Users are expected to be of varying ages, using a variety of different devices. The app should be easily usable so that the user does not make too many mistakes to complete a task. Jakob Nielsen came up with several usability evaluation heuristics to help improve the usability of systems fast and cheaply, in hope to solve issues like these for software designers. He has 10 heuristics on usability for User Interface Design which will be used in this system, along with guidelines from other experts, to identify and measure usability.

The main concepts of this system are; finding flashcards via geo-caching, and then studying these flashcards. Nielsen (1995, para. 2) states that “users should always be kept informed about what is going on via appropriate feedback and within reasonable time”, which is important for this system as it has two main concepts. This includes such things as confirmation messages, possible loading screens and possible help documentation. If documentation is included then Nielsen (1995, para. 11) also suggests that it “should be easy to find, focused on the user’s task and a simple list of instructions and not too large”. This system will be designed in a way that should mean that help and documentation is made redundant, however, as an added extra it can’t hurt the user if it is added in.

When considering content, the terms and language used must be “simple to the user, rather than complex system terminology” as Nielsen (1995, para. 3) states. E.g. Mentioning “caches” in this app will not be helpful, as most users may not realise what geocaching is. It is better to use phrases that any user, new to the subject or not, will be able to understand.

As Nielsen (1995, para. 5) says in “Consistency and standards”, “you should not confuse users by using different terminology, situations or actions when they actually mean the same thing”. This is also suggested by the Android Developers (n.d., para. 13) where they say that if something looks the same then it should always work the same. In terms of this system, this means keep layout, objects, images etc. similar across devices.

The app should have an easy learning curve and be simple to cater for all ages. Shneiderman (n.d., para. 10) states in his Heuristics that reducing short-term memory load is important, as “you want to avoid interfaces where users must remember information between different displays”. As this is a learning app, users must make user of their short and long term memory. However, this does not mean that the use of the rest of the system should be taxing on the user. By reducing short-term memory load throughout the system, users should find learning new words easier. They will have less things to remember, therefore having higher potential to learn more.

The Android Developers (n.d., para. 7) state that “short phrases with simple words” is important as “people are likely to skip sentences if they’re long.” Small sentences, especially in a foreign language, are easier to understand than longer sentences. As the systems aim is for users to learn foreign languages via sentence flashcards, this should be taken into consideration for cache content, as well as system objects.

**Operationalizing Usability Concepts**

In Usability Metrics (2001, para. 5) Nielsen lists the basic measures of usability:

* Success rate
* Time taken per task
* Error rate
* User satisfaction

He also mentions other metrics such as the amount of times users backtrack to find the correct page. He then discusses comparing two designs and how to quickly tell if a new design has improved in usability. He recommends giving users tasks, then record how long it takes the users to complete the tasks.

A simple metric that can be used to indicate improvement, is to calculate “how long it takes users to do stuff” for each design, as recommended by Nielsen in Usability Metrics (2001, para. 14). Using the times taken for each task you can add them up and see the time difference between different prototypes. By doing this you get a “usability score” which can tell you the change in usability between designs.

**Task Allocation**

The following table shows the tasks that will be performed throughout the system. Each task will be caused by an action caused by user input or due to an action in the system itself.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task | Mobile | Desktop | Server | User |
| Look up possible caches | Yes | Yes | No | Yes |
| Check progress | Yes | Yes | No | Yes |
| Sync data across devices | Yes | Yes | No | Yes |
| Store data | Yes | Yes | Yes | No |
| Check map | Yes | Yes | No | Yes |
| Review flashcards | Yes | Yes | No | Yes |
| Obtain flashcards from cache | Yes | No | No | Yes |
| Obtain experience points from cache | Yes | No | No | Yes |
| Get data from google maps (e.g. whether the current location is a shop, bank, park etc.) | No | No | Yes | No |
| Sign in/Create account | Yes | Yes | No | Yes |
| Create/edit account info | Yes | Yes | No | Yes |
| Change display language | Yes | Yes | No | Yes |
| Change learning language | Yes | Yes | No | Yes |
| Authorise account/sign in | No | No | Yes | No |
| View map | Yes | Yes | No | Yes |
| Skip flashcard | Yes | Yes | No | Yes |

**Potential Issues**

The app could have a variety of accessibility issues. There are different types of disabilities, many can make "small screens" harder to use, e.g. partial blindness. In this case the user would prefer a device with a large screen. If this app is not optimised for accessibility, e.g. buttons or text are small, then the user will find it harder to use. Users will use the mobile app outside which could have environmental issues. Glare could affect seeing the screen, so this may affect the brightness of the color scheme. Also, not everyone in the UK has access to 3G and roaming services. This could affect how users obtain caches.

**Initial Interface Specification**

**How the System Works**

The mobile designs are for Android devices therefore the user can navigate with the back, home and switch app buttons that come with Android.

The first thing the user must do on either version is to register an account then sign in. Once they sign in they will be taken to the home page (mobile) or to the find decks page (desktop).

The home page shows the main functions to the user. The three main options here are, study, find decks and view map. If the user clicks on “study”, then they will move to a page that lists decks that they have already found.

From the study decks page, the user can click a deck and begin studying. Here they will see a sentence in their target language, attempt to read and understand it, then will click “Show back” to display the back of the flashcard which shows the translation, meaning of the words in the sentence and any other notes. They then grade themselves and move on to the next card. They will continue this process until they have run out of cards to learn for today, or go back.

Clicking the “Find decks” option, users will be taken to a list of flashcards that are close to them. This will show some decks and they can find more by clicking map at the bottom of the list. If they click a deck, then they will be taken to a unique page for that deck.

If the user clicks “View Map”, they will be taken to the map so that they can view the locations of all flashcard decks.

If the user clicks the chart at the top of the application, they will be taken to a progress page. If they click the sync button, a sync window will appear and sync the users’ progress with the server. If the user clicks the languages button, then the user will be taken to their language settings.

The desktop version includes features for experienced users e.g. keyboard shortcuts. “-“, “+” and “scroll wheel” can be used to control the map, and CTRL+Z can be used to undo a learning a flashcard.

**Site map**

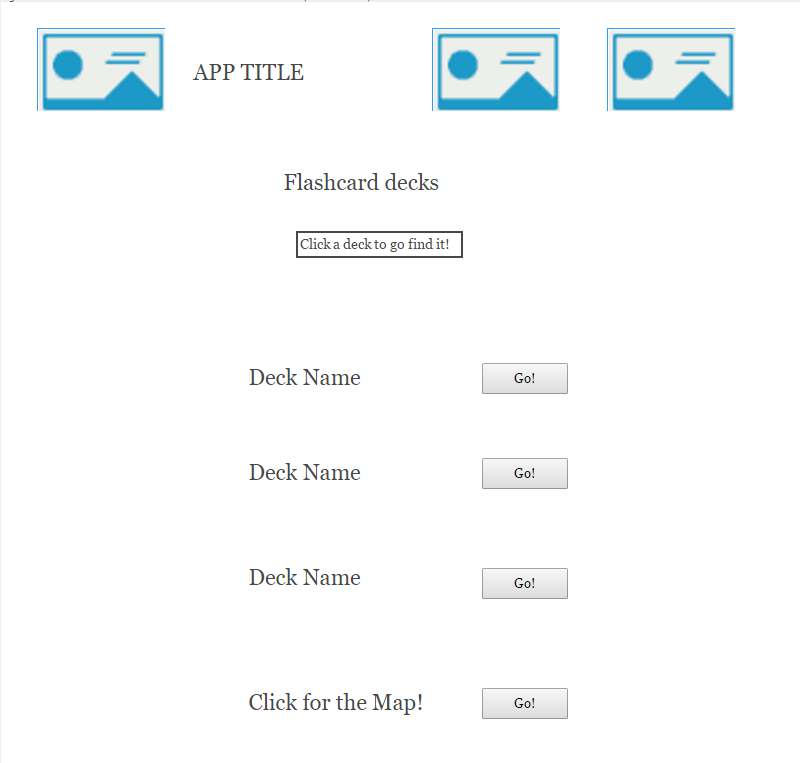
The below site map gives an overview of the applications navigation. Each page on the 3rd layer can be accessed by each other on desktop, whereas the mobile version makes use of the built-in Android “back” button to go back to previous pages.

**Wireframe of Main Components**

The Android Developers (n.d., para. 8) claim that “pictures are faster than words”, telling us to “consider using pictures to explain ideas” as “they get people’s attention and can be much more efficient than words.” The system will therefore implement logos and small images that visually represent cache locations on the map, ticks for confirmation, graphs for progress etc. These images should be easy to understand for the user.

The left most image should be a “graph”, the next should be “sync” then the next is “change languages”.

**Find decks page**

****

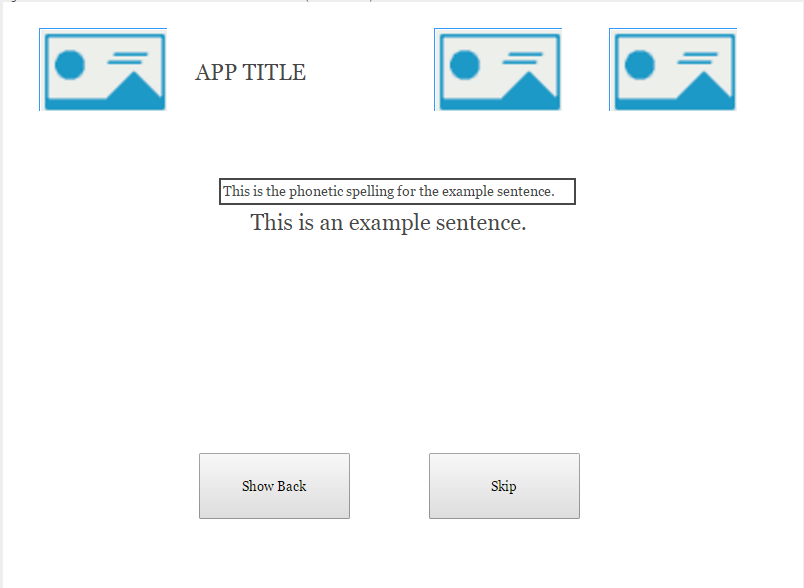
The Android Developers (n.d., para. 7) also observe that using “short phrases with simple words” is important as “people are likely to skip sentences if they’re long.” This is a simple rule but is important to remember when writing explanations, button names, and content.

Lots of different apps use buttons or linked text to navigate between screens. Here, both text and buttons will take the user to the next page. This means that, whatever the user is used to, their instinct will get them where they want to be.

The mobile app for Android uses Androids’ “Roboto Medium“ font. The desktop version uses Microsoft’s “Microsoft Sans Serif” font. All text will be a readable size (no lower than size 14), titles and button labels will be larger to make them stand out more.

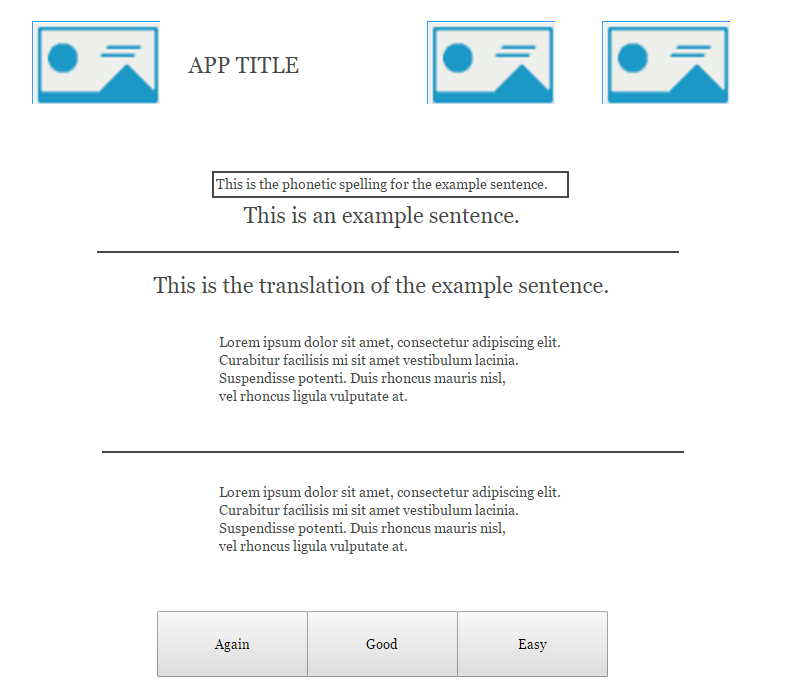
**Example of Studying a Flashcard (Front and Back)**

As mentioned before, use of simple sentences to make learning easier is essential. The top line is how the sentence is pronounced and the bottom line is the sentence. Each language has methods for explaining its pronunciation so this will be included to aid users. By including this, we increase our accessibility and allow for a wider audience.



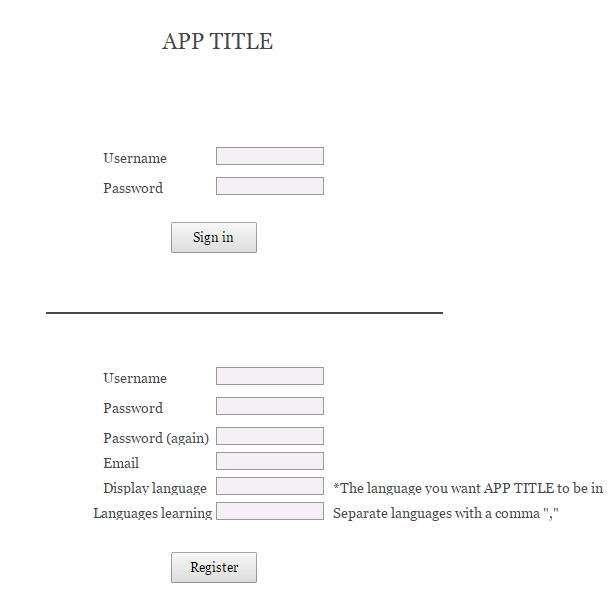
Simple, large and clearly labelled buttons so that the user knows what each one. When the “Show Back” button is pressed, the back of the card is displayed. “Skip” will move the user on to the next sentence.

These top three buttons and the title are kept the same through every page in the **mobile** application.



The back of the flashcard gives a translation of the original sentence, definitions of words and extra notes. The original front of the card is kept at the top so that the user can reference it.

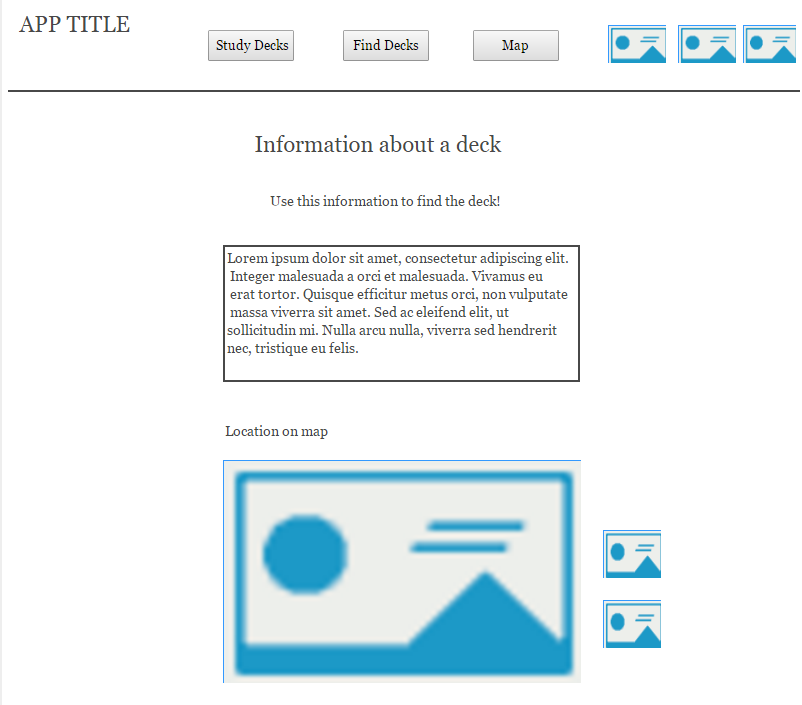
Buttons are clearly labelled so the user knows how to grade themselves. A short line of text is also included to explain the grading system.



“Helpful hints that speak the users’ language”, as recommended by Nielsen (1995, para. 3). As “display language” might confuse some users, hints explaining in more detail have been added.

Text entry fields. This should be obvious to the user that they should enter text here.

Nielsen (1995, para. 6) stats that “careful design which prevents a problem from occurring” is “even better than good error messages”. This should be implemented by giving instant feedback to the user as they type into a field. If what they are typing is incorrect, then the cross will turn to an “X” and a message will tell them how to fix the issue. This makes the registration process easier for the user which is important as this is the first stage of getting the user to use the system.



The map will show users where decks are. It will contain standard images to identify the user’s location and the deck’s location to reduce cognitive load on users, as using text would make it hard to read and use.

This is where information such as coordinates will go.

For desktop, the “navigation bar” appears on every window besides the sign in window. This makes sure that the user can get back to the main functions of the system and so they know where they are in the program.

Magnifying glasses, “zoom in” and “zoom out”.

**Low Fidelity Prototypes**

Using the low fidelity prototypes, data will be obtained via a series of user tests. Five users of different computing capabilities will be chosen to test both the mobile and desktop versions of the low fidelity prototypes. There will be five tasks the users will be asked to complete which should cover many the usability issues throughout the application. The low fidelity prototypes will be printed off and placed in front of the user. They will be told to follow the task and the tester will keep an eye on where the user wants to click, or move to. When they try this the tester will interact with them by handing them the new screens or messages that would appear on the real system.

**Mobile Low Fidelity**

Balsamiq has been used for the low fidelity prototypes of both versions of the system because of the ability to quickly make prototypes. Paper prototypes do have the advantage of being super quick to create, however, they might not give great feedback if they aren’t drawn to a relatively high standard. For this reason, using a piece of software to quickly create shapes, readable text etc. is a great idea. This will make prototype testing a lot easier and will allow the tester to gain more accurate data. Balsamiq is also great for creating similar screens as you can just copy and paste, then make slight. Once the screens are ready, you can print them and ask someone to test them. From this process you can very easily get data and then plan changes ready for the high-fidelity versions.

Using the wireframe, the prototypes for mobile where created. Any images and interactive features have been inserted so that the user can get a better understanding of the “feel” of the app during testing. This should provide more accurate results.

*Here are all the designs for the mobile low fidelity prototype.*

|  |  |  |
| --- | --- | --- |
| **Sign in / register**  **C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Sign in and Register account.png** | **Sign in / register**  **C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Sign in and Register account.png** | **Home page for Android Users Only**  **https://lh6.googleusercontent.com/6WkIKt8jT3LjMlVTciPIs-Ll4dTkSwaldBgYNLa8wAq-lcF3CI8ExFmAy0NAXExr0EFAbKupE-8Cc2jM2qdnI4hi58ffKadD_jfmdVTeMleTVBl95RE2Czjk2CzxkNt-8mcLaPM** |
| **Map view**  **C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Map.png** | **Find decks page** | **Study decks page**  **Studying flashcards screen** |
| **https://lh5.googleusercontent.com/q51x9qqh2rEwzyuDLTkMAxQ6zUyMfKbSbOpkLJBcGt0gt35QtY1SjK_mOukssRjp3hLaJbW19_k1Hxy2_XNJHEuU9i-9jId8Fhm3ynFW2-VYtvGlgy5ANdyV2qTyb-umEL-EqEwStudying a Card (Front)** | **Studying a Card (Back)**  **https://lh5.googleusercontent.com/q51x9qqh2rEwzyuDLTkMAxQ6zUyMfKbSbOpkLJBcGt0gt35QtY1SjK_mOukssRjp3hLaJbW19_k1Hxy2_XNJHEuU9i-9jId8Fhm3ynFW2-VYtvGlgy5ANdyV2qTyb-umEL-EqEw** | **Information about a Deck**  **C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Deck Information Example.png** |
| **Information about a Deck**  **C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Deck Information Example.png** | **Progress Page**  **Progress and Statistics** | **Progress Page**  **Progress and Statistics** |
| **Language Settings**  **C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Language Settings.png** | **Changed Details**  **Changed Details** | **Obtain new deck?**  **Obtain new flashcard deck_** |
| **Sync**  **Sync** | **Obtained a new deck!**  **C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\You have obtained new flashcard deck!.png** |

**Desktop Low Fidelity**

Here are the low fidelity designs which were created in Balsamiq. Again, Balsamiq was used due to its simplicity and ease of use. The desktop low fidelity prototype has been created from the wireframe designs but made to be a bit more understandable for the user. Any images that were not specified in the wireframe have been included and a basic “fake” implementation of what the interactive map would look like has also been implemented.

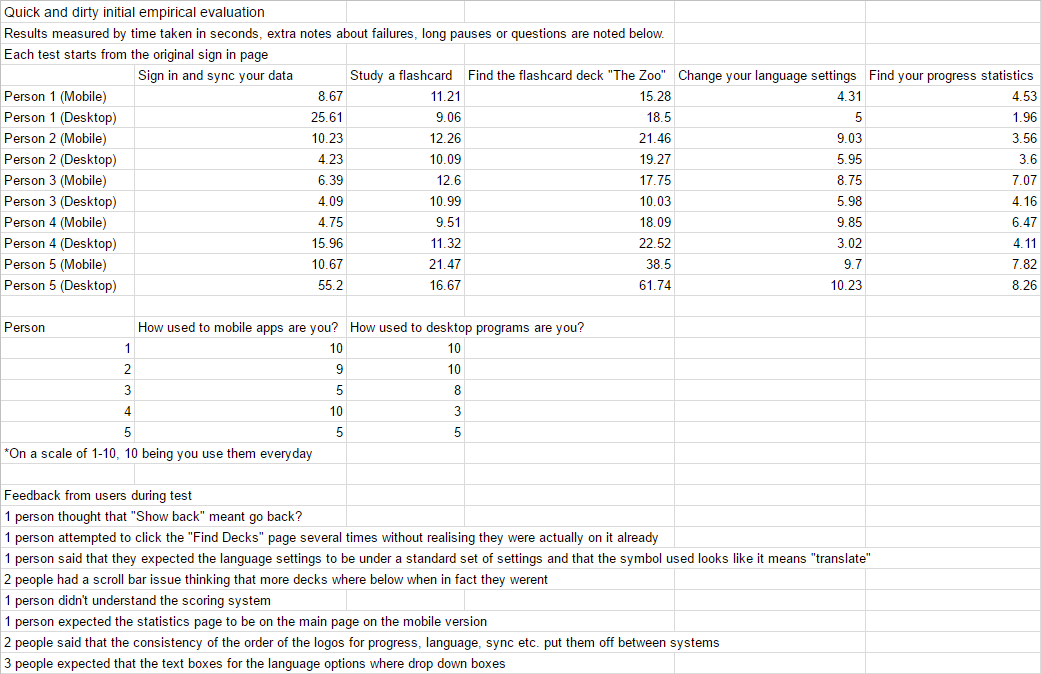
|  |  |
| --- | --- |
| **Sign in and Register** | **Information about a Card**  **C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Deck Information Example.png** |
| **Studying a Flashcard (Front)**  **Studying a flashcard front example** | **Studying a Flashcard (Back)**  **Studying a flashcard back example** |
| *All the designs can be found in the appendix.* | |

**Quick and Dirty Initial Empirical Evaluation**

Once the low fidelity prototypes where made, testing was done to see how usable the prototypes are. Using the techniques in “Operationalizing Usability Concepts”, five users were chosen to test both prototypes. The prototypes created on Balsamiq were printed out and then screens were taken away and placed in front of the users as they clicked objects on the paper prototypes. During the testing, the users were given set tasks and the time taken to complete these tasks was measured. The users were each assured that it was the prototypes’ usability that was being tested and not them. They were also encouraged to keep talking while they performed tasks. Making users do this, as recommended by, Nielsen in Thinking Aloud: The #1 Usability Tool (2012, para. 6) allows us to see the exact thought process of the user, where they are going, what they think is the correct path, why they did something etc. All this information gives us a clue as to how to layout navigation, buttons, text etc. as it tells us exactly where users go wrong and therefore where the usability of the prototype needs improving.

After testing was completed, analysis was done on the results of the tests to see what needs improvement. Process of evaluation of these results can be seen below.

*Empirical evaluation results*



**Analysis of Results**

**General Feedback from the users**

It appears that syncing proved to be difficult for some users, especially on the desktop version, as 3 users took over 10 seconds to complete this task, one of them taking almost a minute. The last user is someone who doesn’t use apps that often, however, and they didn’t entirely seem to know what “Sync” meant.

During the process of finding and studying a flashcard, most users coped fine in finding and learning it in decent time. During the studying of a flashcard, person 5 of the mobile version thought that “Show Back” meant to take them back to a previous page, when in fact it meant to show the back of the flashcard. This same person also didn’t understand the scoring system of the flashcards. This suggests that the buttons text needs changing and that the explanation for the scoring system should be written better.

The 3rd test was included to see if the users could figure out how to find more geocaches that were not on the list in the “Find Decks” page. To no surprise, this took a bit longer for them to complete, however, they all realized that by clicking on “View more via map”, that they could find the deck. Hinting to the user in some way that the map exists, and that they can use it to find other flashcard decks, could be implemented.

Both changing language settings and finding the progress page were very easy for all users and was achieved very quickly, showing that easy to understand logos can be very effective. A couple of people mentioned that they didn’t like how the locations of these buttons were inconsistent between prototypes and one person said that the logo for language settings looked too much like it meant “translate”. These comments will be taken into consideration when making changes in the next implementation of prototypes.

**Modification of Interface Specification**

Changes to the original interface specification have been noted here and have been implemented in the high-fidelity prototypes in hope that they will improve usability.

**Change History of Interface Specification**

* Text for “Show Back” button during studying of flashcards changed to “Show Answer”.
* Message box now displays if a user tries to navigate to a page that they are already on.
* Added a settings page that will contain language settings within it. Language settings page has been removed. Settings page is now accessible to the user via a burger menu in the navigation bar (mobile) and within “Edit” on a menu bar (desktop).
* The text for the scoring system has been made bigger so it stands out more and is easier to read. The explanation has also been changed so it is easier to understand.
* Navigation bars on both mobile and desktop have been improved and are now more consistent.
* Changed text boxes, for language settings, to drop down menus.
* Removed sign in and register form for mobile and instead made them separate. A new “title” page has been added which gives the choices to sign in or register, which then takes them to the correct page.

**High Fidelity Prototypes**

When it came to creating the high-fidelity prototypes, choosing tools that allow the prototype to move into full development later was important. This will save time and money on re-creating the system as development can just be continued from the high-fidelity prototype. All that would need adding is the systems functionality, as at this stage a lot of the functionality isn’t yet fully implemented, but is instead meant to look like it is for usability testing purposes.

**Mobile High Fidelity**

The mobile high fidelity has been implemented with proto io, which allows quick and high quality development of real-looking prototypes. You can easily add interaction, effects and add more complex features that other software like Balsamiq can’t do. Proto io also allows you to download the source code, allowing you to develop them further. As this application is aimed at Android users, during the creation of this prototype, Android features and standards were implemented. For example, all the buttons you see are the default Android recommended buttons as well as the navigation bar, text font and size, as well as the progress bar for syncing. The reason for doing so is to keep users using objects that they familiar with as recommended by usability.gov in User Interface Elements (n.d., para. 1) where they state that “choosing to adopt those elements when appropriate will help with task completion, efficiency, and satisfaction”. When run on a phone or via proto io’s testing facility it looks and feels like a native Android application and could very easily be implemented as a real application if the functionality is developed.

*Here are all the designs for the mobile high fidelity prototype*

|  |  |  |
| --- | --- | --- |
| **Title** | **Sign in** | **Register** |
| **Home page for Android Users Only** | **Find decks page** | **Study decks page** |
| **Studying a Card (Front)** | **Studying a Card (Back) C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\10.1-Studying Flashcards (Back).png** | **Information about a Deck (scrollable)** |
| **Settings (scrollable)C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\11.1-Settings.png** | **Progress Page (scrollable)** | **View Map** |
| **Study Finished Message** | **Changed Details Message**  **C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\15.1-Changed details.png** | **“Obtain new deck?” Message**  **C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\16.1-Obtain new deck_.png** |
| **Content Locked Message** | **Sync Message** | **“Obtained a new deck!” Message** |

**Desktop High Fidelity**

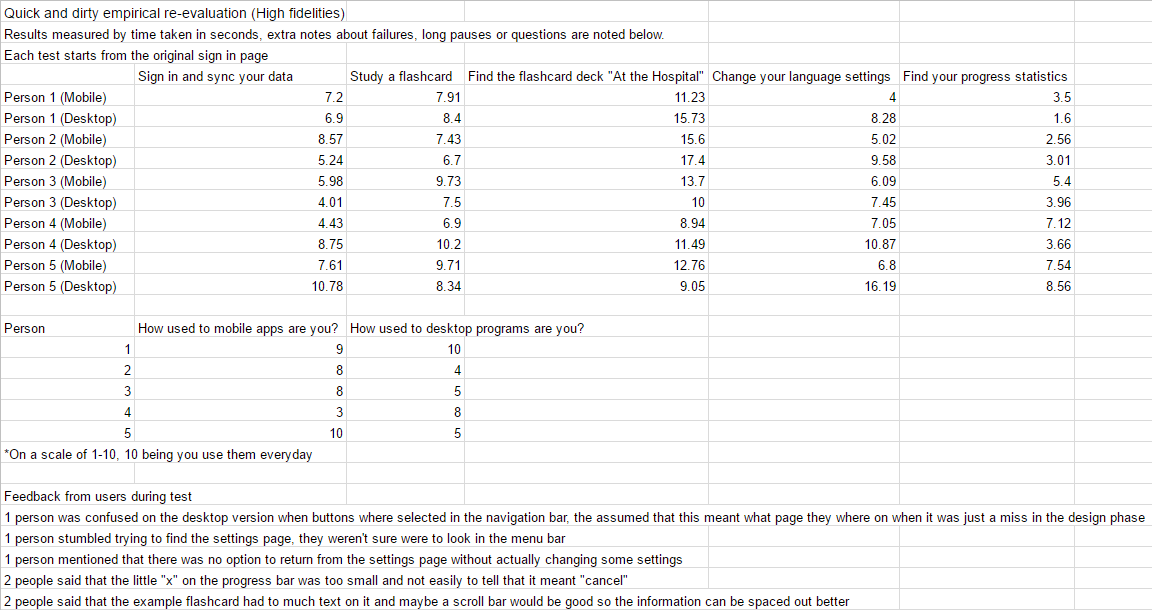
The desktop high fidelity has been implemented using Visual Studio and has been programmed in VB.NET. Visual Studio allows for quick creation of programs that run-on Windows. Without changing too many settings, these programs tend to look like very real, generic Windows based programs, which is great for creating a high-fidelity prototype users testing the system will feel like they are using a normal system. Visual Studio and VB.NET are quite capable of producing fully fledged programs. It is simple to create a prototype in Visual Studio and the fact that you can then develop these prototypes into real systems makes Visual Studio a great choice for creating a high-fidelity prototype. At this stage, only basic functionality is required for the prototype and most complex functionality can often be made a lot simpler with “on click” events to hide and show objects/forms, thus making it incredibly efficient to produce high quality looking prototypes.

|  |  |
| --- | --- |
| **Sign in and Register**  C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Manabu.vshost_2017-03-22_19-10-20.png | **Information about a Card**  Unfortunately, on the machine used to test the high fidelity, the version of IE that the function Navigate() in VB.NET uses is out of date. However, the testing was done on different machines where this worked fine.  **C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Manabu.vshost_2017-03-22_19-28-44.png** |
| **Studying a Flashcard (Front)**  **C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Manabu.vshost_2017-03-22_19-15-00.png** | **Studying a Flashcard (Back)**  C:\Users\マット\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Manabu.vshost_2017-03-22_19-15-10.png |
| *All the designs can be found in the appendix.* |  |

**Quick and Dirty Empirical Re-Evaluation**

During testing of the high fidelities, the testing process was conducted in the exact same way as before, except this time the prototypes were not printed out on paper and 5 new users who had never seen the system before were chosen. As before they were given 5 tasks which were the same as the tasks used to test the first prototypes. These results were then analyzed and a comparison of the two sets of prototypes can be seen in the next section.

*Empirical re-evaluation results*



**Analysis of Results**

**General Feedback from test users**

Overall the prototypes appear to get each task done at a quicker pace than the low fidelities which is great. This means that most changes made to the interface specification have made improvements on the usability of the system. A more in-depth analysis comparing the two sets of prototypes results can be seen in the next section.

During use of the desktop prototype, one person was slightly confused by the fact that buttons on the navigation bar would be “highlighted”, which they assumed to be an indication as to which page they were currently on. This is not the case and is simply a miss in the design process. Visual Studio automatically adds this and it will need to be changed in the next implementation.

As the desktop version doesn’t have a burger menu for settings, one person took a little longer than expected to find the settings page. After a slight hesitation, they realized it was in the menu bar on the top left. This was only a small hesitation and only occurred for one person so it is debatable that this should cause a change.

Someone also mentioned that there is no way of moving out of the “settings” page unless they change some of their settings (desktop). Users may not always want to change their settings, so this needs to be changed in the next implementation.

**Calculating Each Prototype’s “Success”**

By implementing Nielsen’s method of quickly calculating the improvement of a design in Usability Metrics (2001, para. 12), the average time it took to complete tasks where taken and then added up to give a total time. This total can then be compared with each fidelity to see the difference in time it takes for users to complete all the tasks. The below table shows the average speed of which it took a user to complete a task as well as the total time it took for all tasks to be completed per prototype.

|  |  |  |
| --- | --- | --- |
| **Low Fidelity Prototypes** | **Mobile** | **Desktop** |
| Task 1 | 8.14s | 21.02s |
| Task 2 | 13.41s | 11.63s |
| Task 3 | 22.22s | 26.41s |
| Task 4 | 8.33s | 6.04s |
| Task 5 | 5.9s | 4.42s |
| Total time it takes to complete all tasks | 57.99s | 69.51s |
| **High Fidelity Prototypes** | **Mobile** | **Desktop** |
| Task 1 | 6.76s | 7.14s |
| Task 2 | 8.34 | 8.23s |
| Task 3 | 12.45s | 12.73s |
| Task 4 | 5.8s | 10.48 |
| Task 5 | 5.22s | 4.16s |
| Total time it takes to complete all tasks | 38.56s | 42.73s |

We can therefore tell from this data that the mobile fidelity has a 19.43s time decrease for successfully completing all tasks and that desktop has a decrease of 26.78s. This gives us a good idea of how usable the high fidelity prototypes are compared to the low fidelities and shows that changes made to the interface specification have had a positive impact on usability throughout the system, as tasks are now much quicker to perform for users. One thing to note though is that because the low fidelities were tested with print outs, changing screens could have had an effect on time taken to perform tasks.

**Modification of Interface Specification**

Changes to the original interface specification have been noted here and will be implemented in the next implementation of the system.

**Future Changes to Interface Specification**

* Change navigational buttons so that the current page is “highlighted” so the user can tell what page they are on. (Desktop)
* Add a “cancel” button to the settings page so that users can leave the page without having to change their settings.

**Critical Evaluation**

This project has been a good experience and has allowed me to develop a variety of skills in many different areas such as designing and implementing prototypes, analyzing/operationalize usability and testing applications.

If I had to do this process again I think I would possibly consider trying to make low fidelity mockups with paper drawings to see just how they compare with the quality of that produced by software. This would be interesting to see as they are rather efficient to produce but the main reason I decided against using them was because I lack any form of drawing skills. I also think that I could have expanded in to other areas during usability testing, attempting to analyze the memorability, learnability, satisfaction etc. from users instead of going for just numbers that give an idea of the usability of the system. Analyzing these topics in more depth can provide more insight into the usability of the system and therefore improve it further so if I continue development of this system, which I want to, then I will attempt to perform more detailed tests.

Defining usability concepts was quite a hard thing to do as there are so many guidelines and ideas out there. I therefore based a lot of my usability concepts and analysis on Jakob Nielsen’s work. I felt that I could have expanded and research a bit further abroad to incorporate more concepts, however, this would mean the length of the report getting out of hand. However, this has made me aware that not just one set of guidelines or heuristics are good enough to define usability of a system. Of course, they do a very good job of it but you need to consider a lot of different aspects which requires quite a bit of research.

The writing of this report and the development of these prototypes has given a great insight into how incredibly important usability is within development environments. Without usability testing, we have absolutely no idea if our programs are usable or not. Usability testing provides a great way to see how to make your program easier to use for your target users and without it, your system could be incredibly unusable. I also learnt that although testing can be expensive, usability testing is not as complex as it seems and can be a short, simple process that can have a massive impact on your program (and therefore the success of the program).

I think that if I was to develop the application into a real system that I would probably have to run it through this testing process a few more times with first a new prototype and then again for the actual system. Although the high fidelity had some pretty good feedback and was all round better than the low fidelity, it still had issues and is far from a final application on both the desktop and the mobile. Those remaining issues would have to be resolved and I feel that a lot more tasks could have been conducted to unveil and more usability issues that could be residing in areas of the system that have remained untested. When performing the testing with paper printouts I felt that the time taken to hand a new screen to the user could have affected the results slightly and thus don’t fully trust my results for the low fidelity. This is another reason why I would like to perform more testing.

Overall, this project has had a huge impact on my understanding of usability and its importance in development as well as allowing me to develop a variety of skills in prototyping, testing and research.

References

10 Heuristics for User Interface Design: Article by Jakob Nielsen. (n.d.). Retrieved from <https://www.nngroup.com/articles/ten-usability-heuristics/>

Affairs, A. S. (2013, October 09). User Interface Elements. Retrieved from <https://www.usability.gov/how-to-and-tools/methods/user-interface-elements.html>

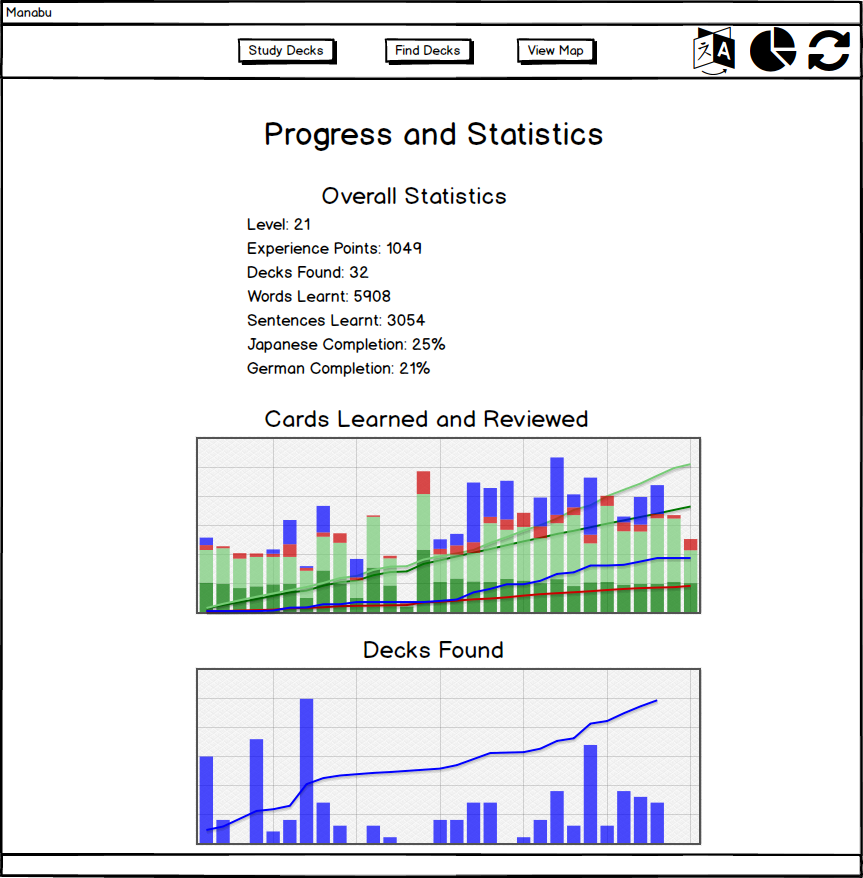
Android Design Principles. (n.d.). Retrieved from <https://developer.android.com/design/get-started/principles.html>

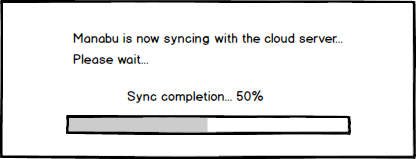
Ben Shneiderman. (n.d.). Retrieved from <https://www.cs.umd.edu/users/ben/goldenrules.html>

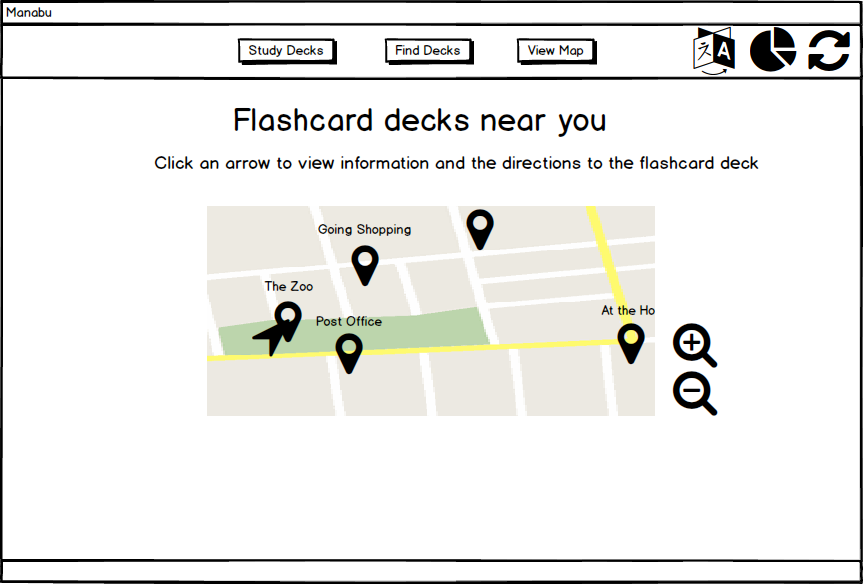
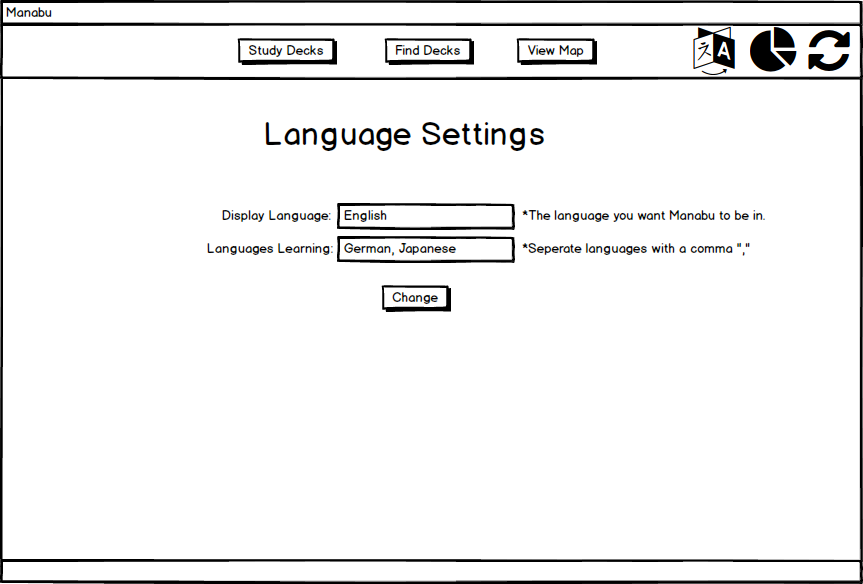
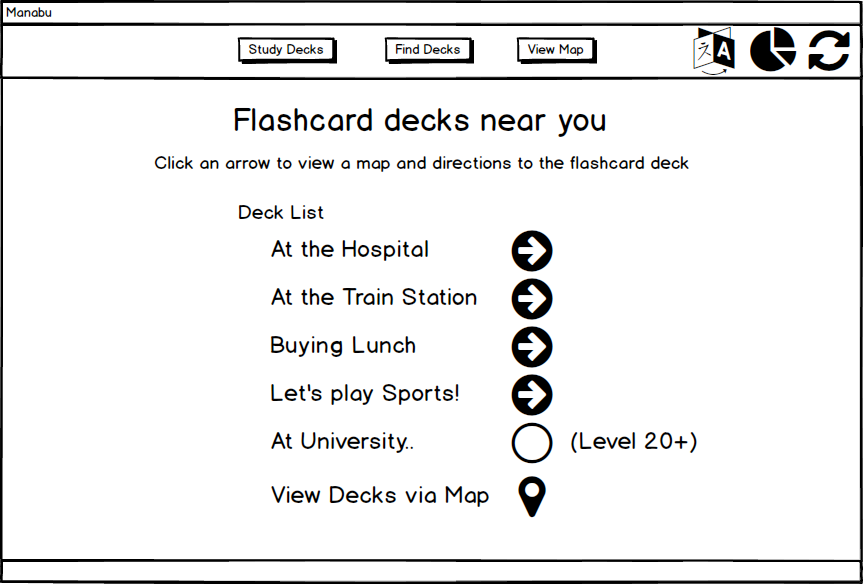
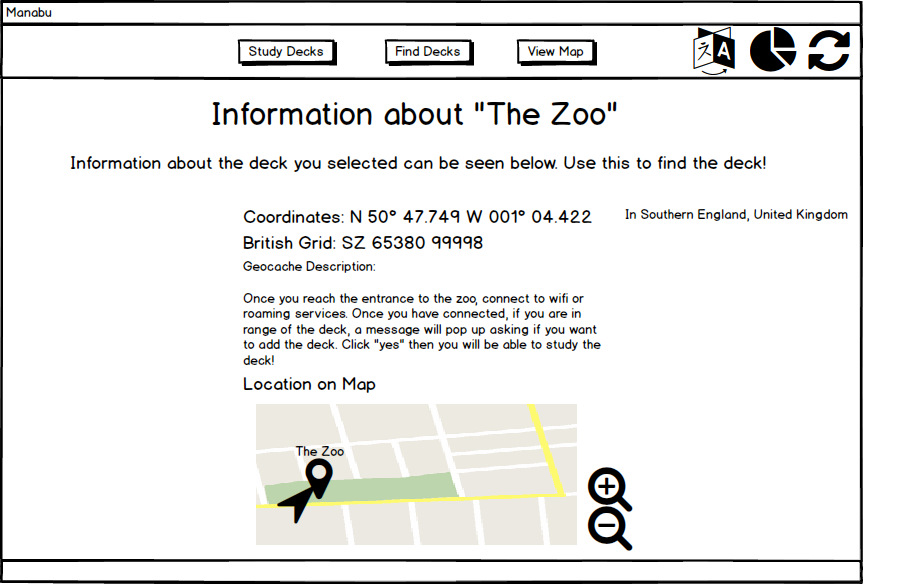
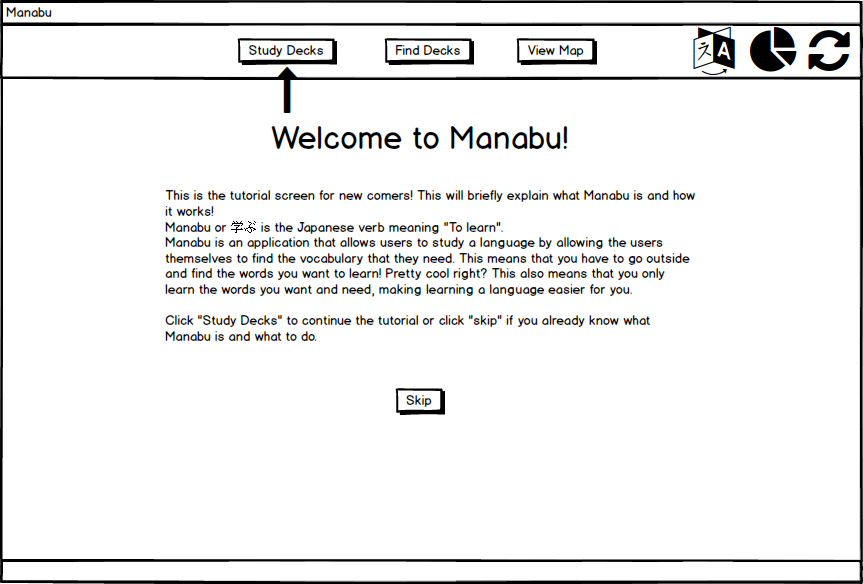
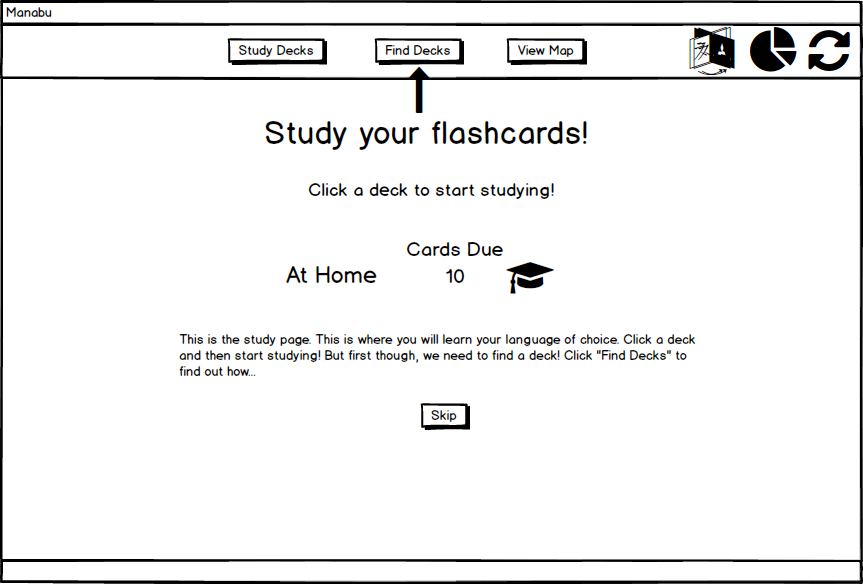
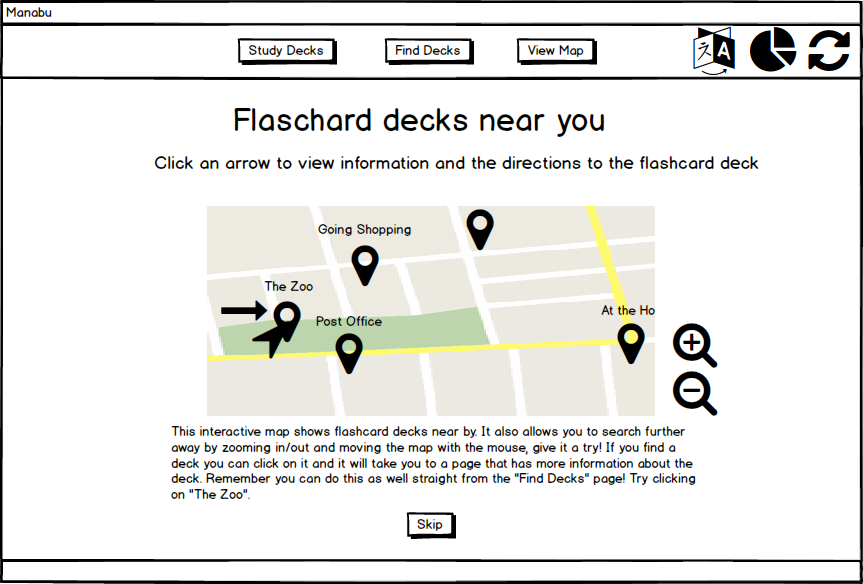
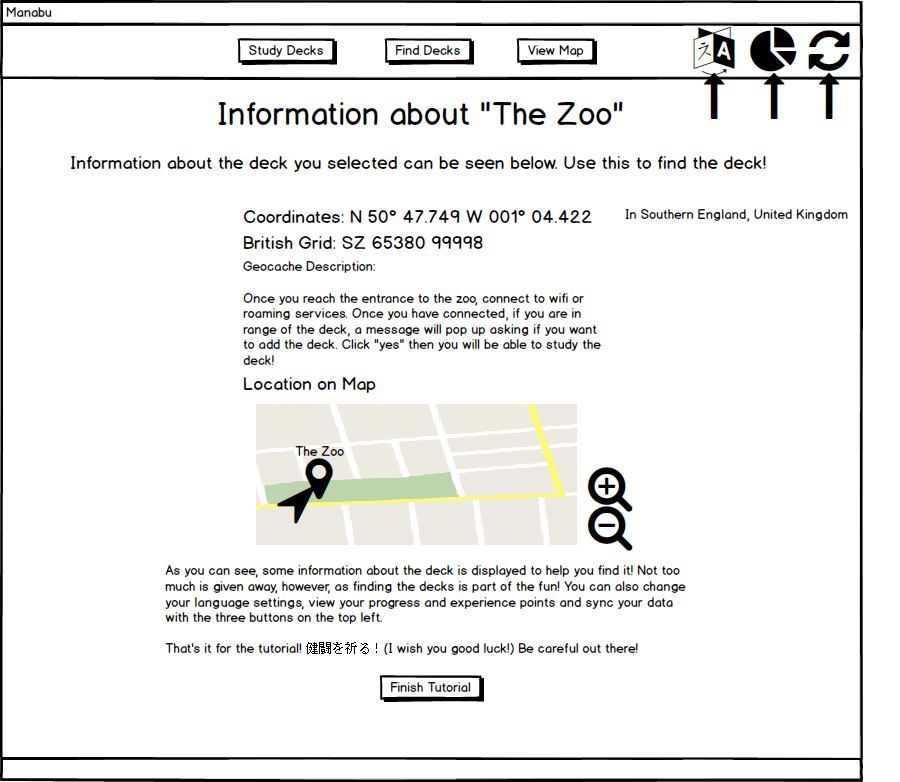
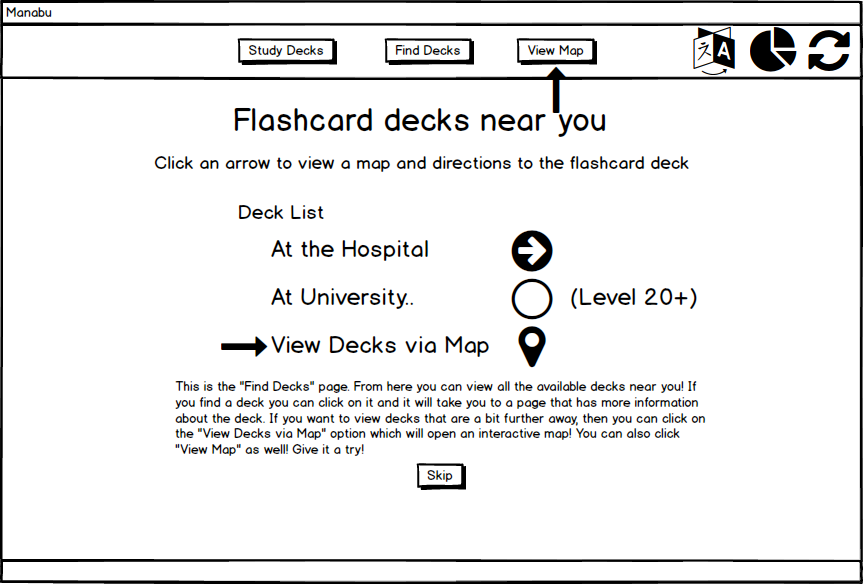
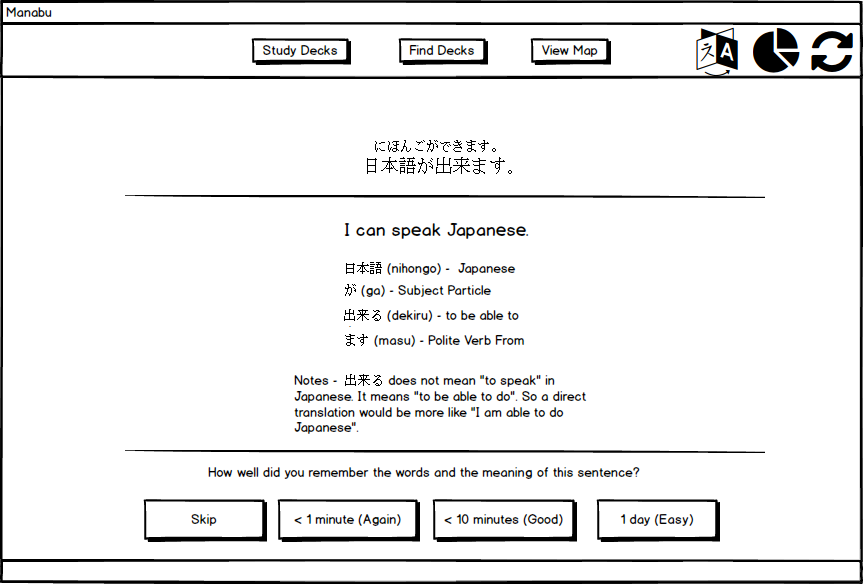
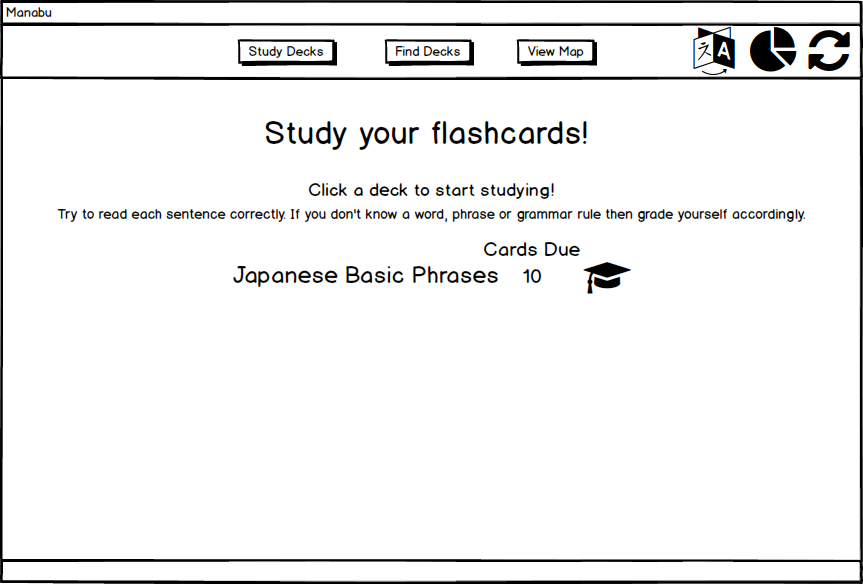
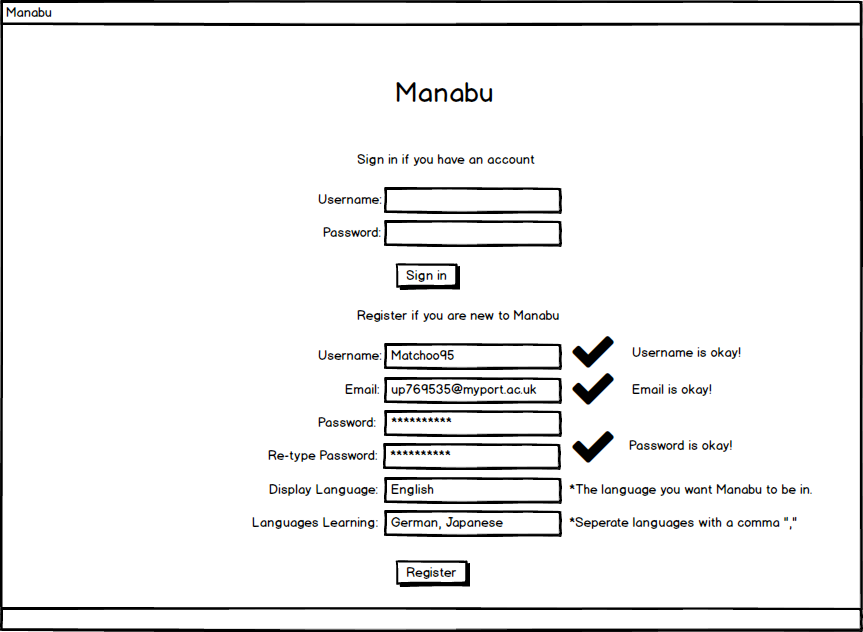
Usability Metrics. (n.d.). Retrieved from <https://www.nngroup.com/articles/usability-metrics/>

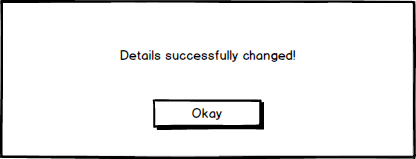
**Appendix**

**All the Desktop Low Fidelity Designs**

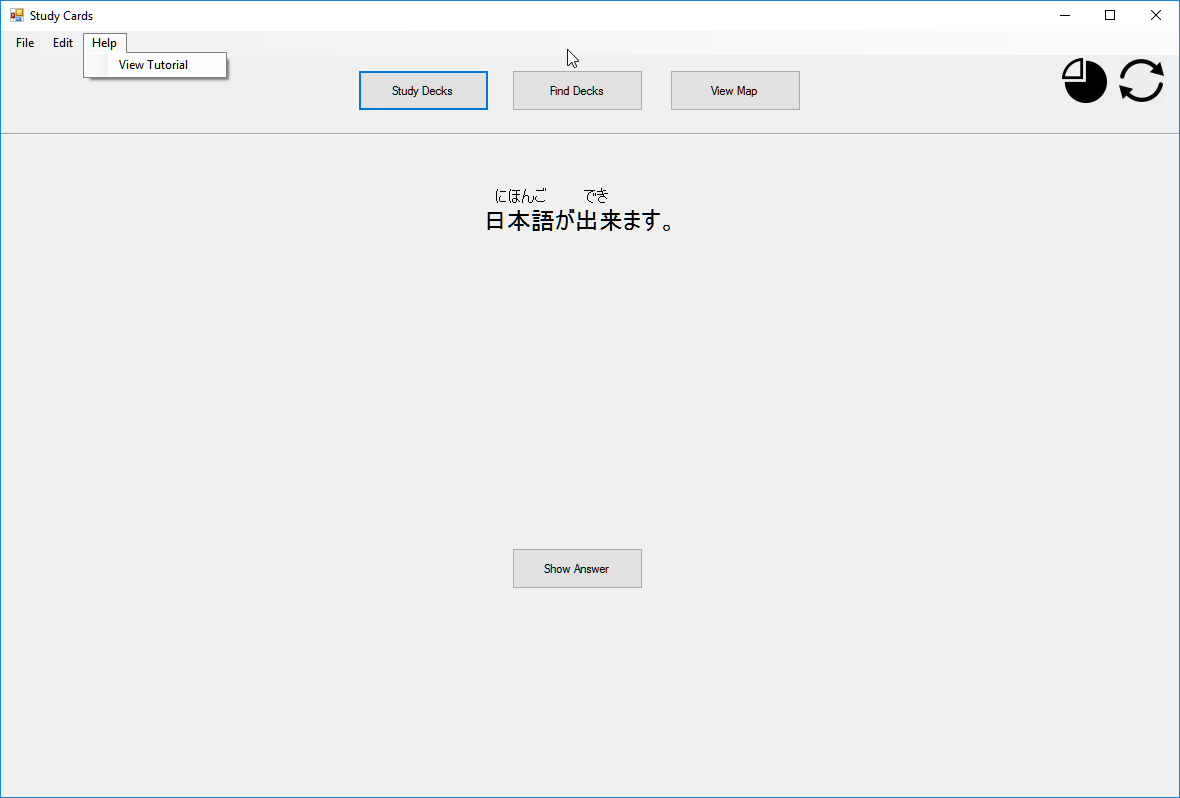
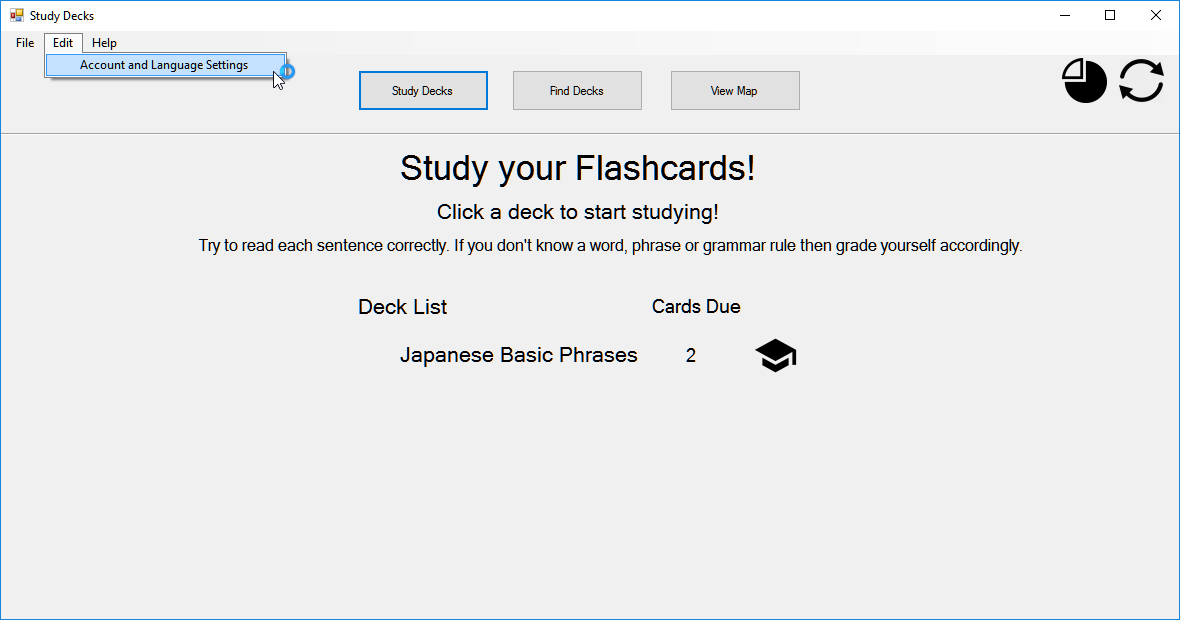
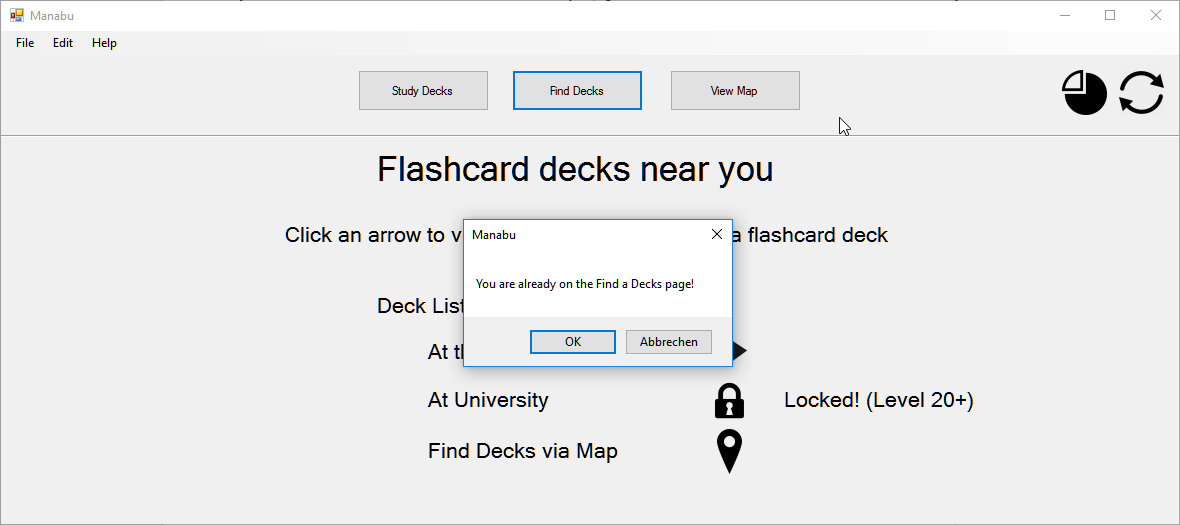
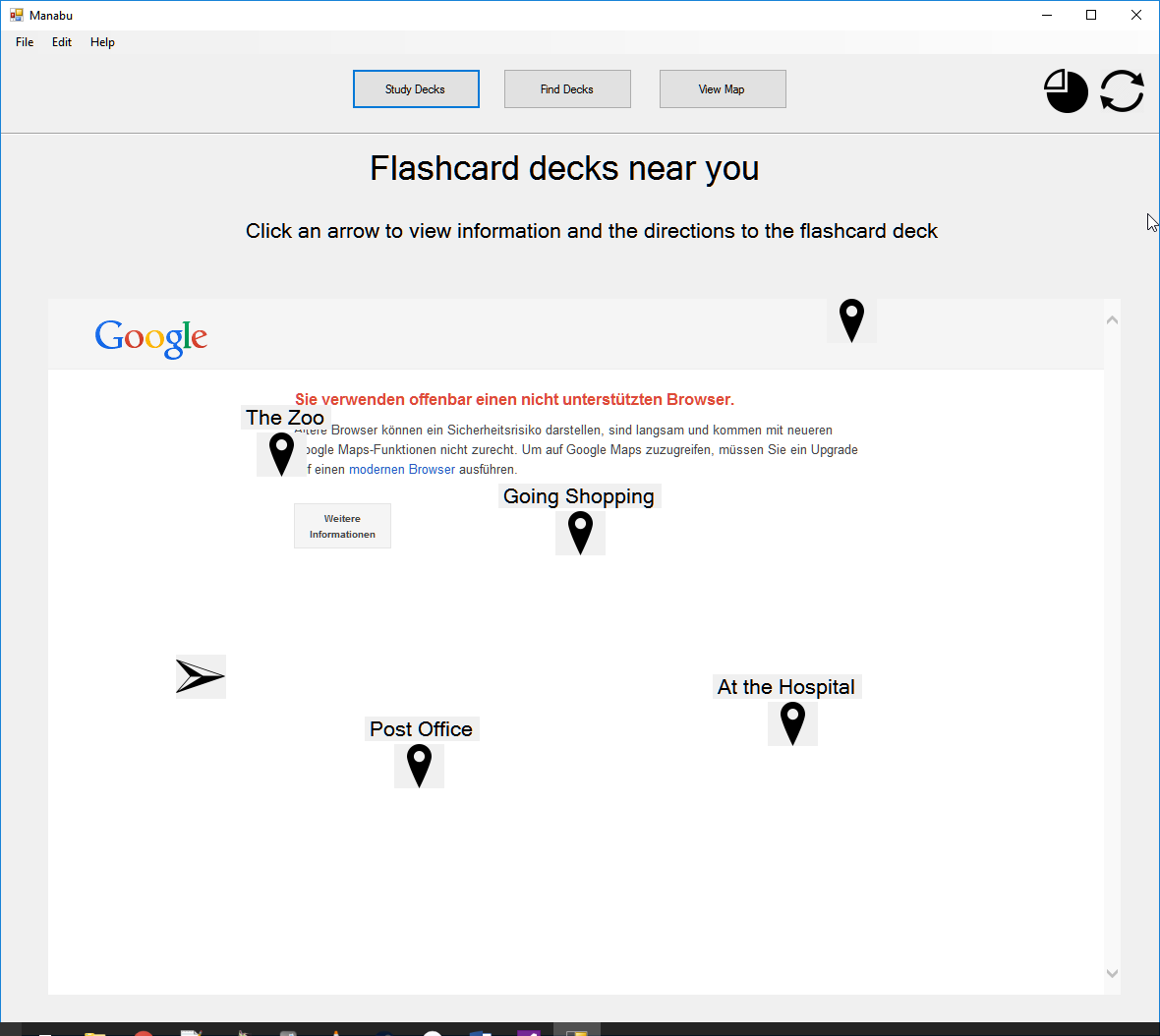
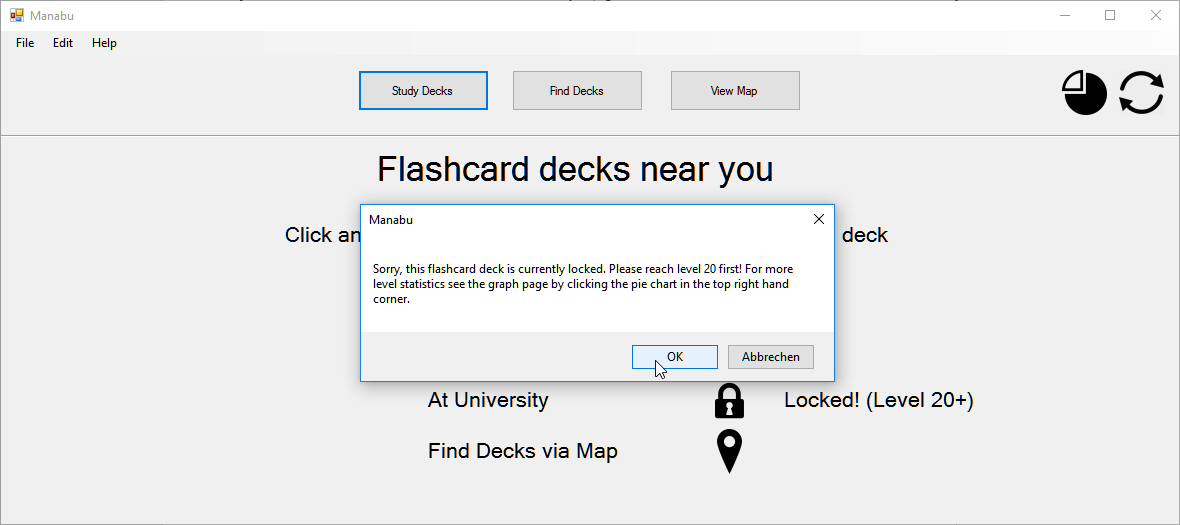
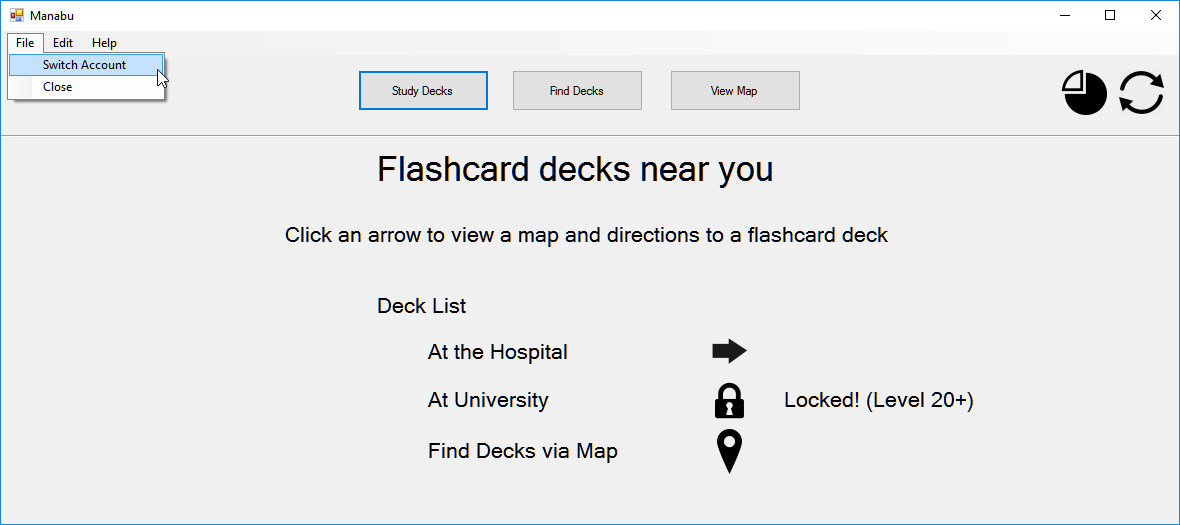
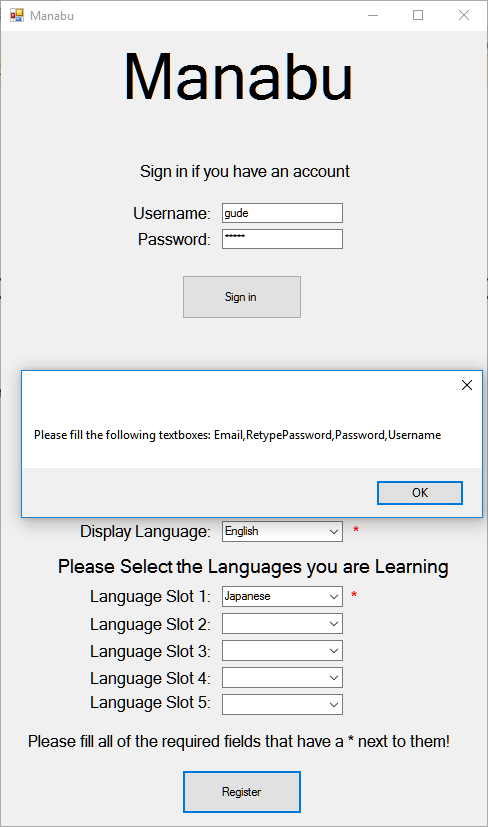
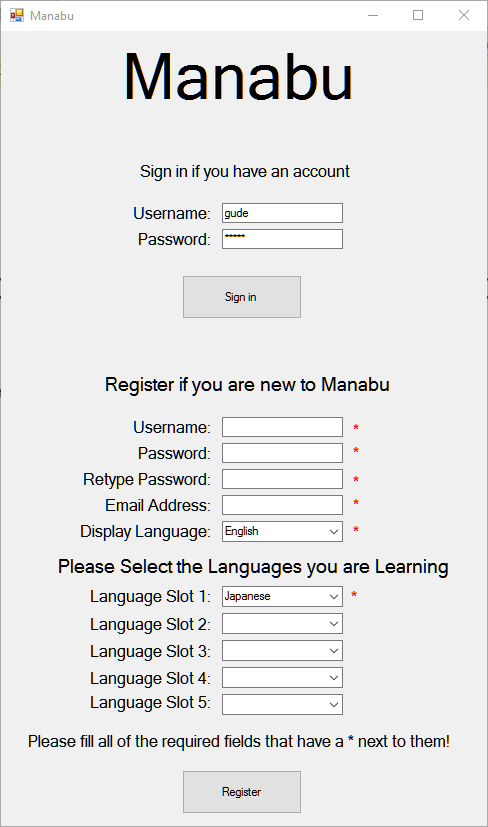
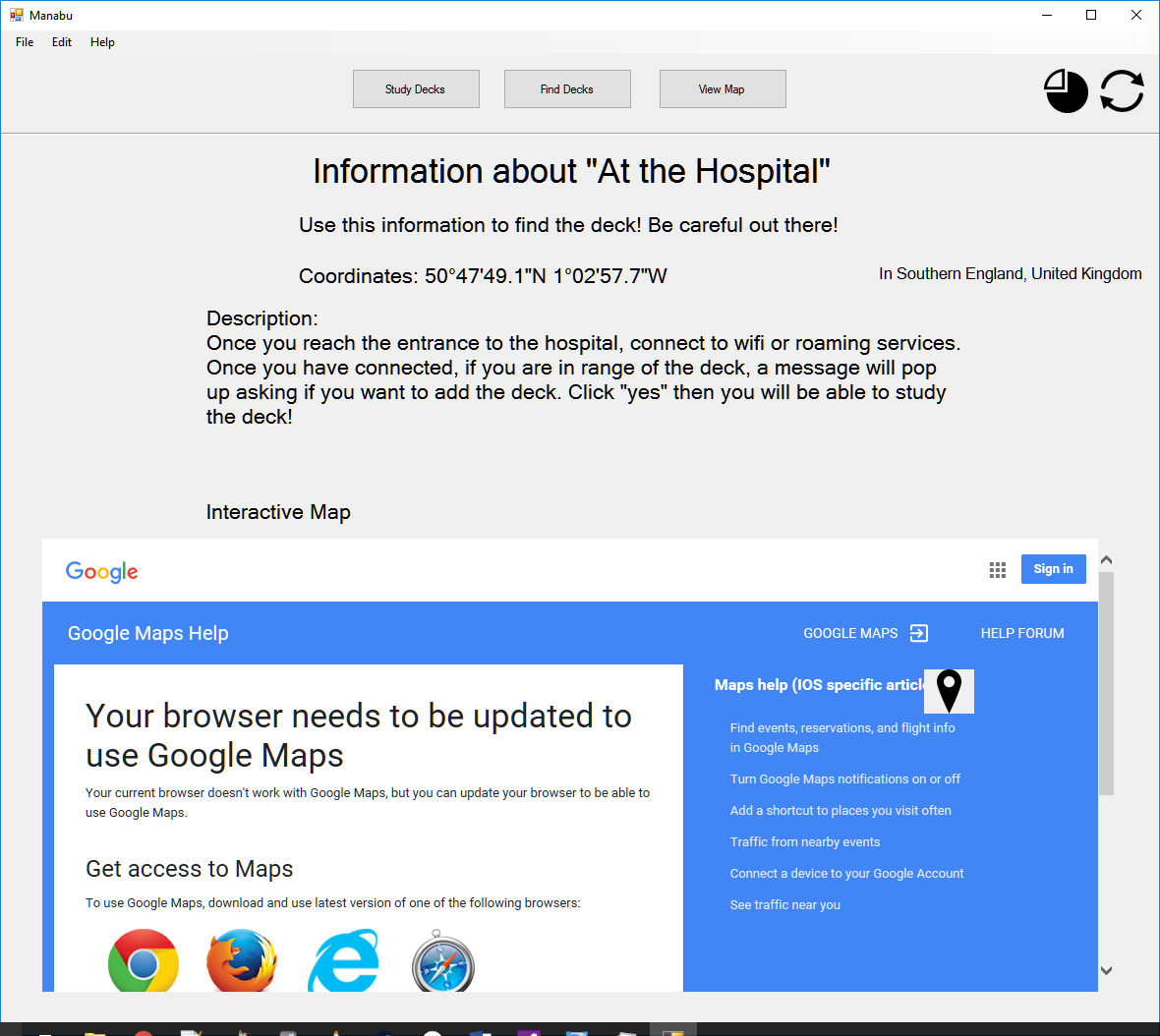


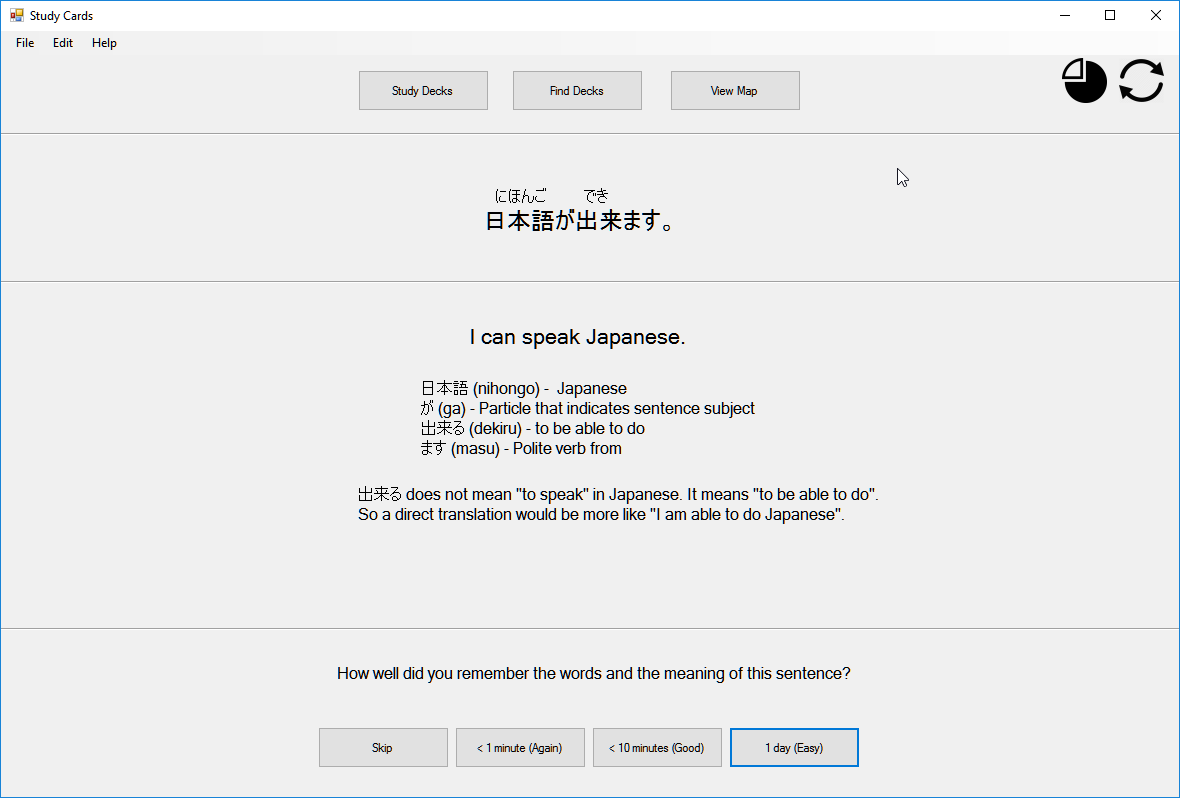






**All the Desktop High Fidelity Designs**



****

