**Getting Usability into Development Environments**

**Section 1**

**The 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 more experience and higher levels. Certain rarer locations that will contain harder words will be level locked, so that only users who have gained enough experience points can unlock and learn 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 flash card sets.
* Learn and review sets of flash cards.
* Skip separate flash cards in case of issues with difficulty.
* View their 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 categories of locations, such as “Train station” or “University”. The user can then click on one and a list of locations will appear along with coordinates and distance from the user. If the user clicks on one of these options, then more information such as a map and information about the words they can learn 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 automatically appear on their phone and will tell them that they have gained a new set of cards and some experience points. This will then get updated to the user’s profile and synced to the cloud. These updates will then be sync across devices when the user logins in (on a device).

**Usability Concepts**

Users are expected to be of varying ages, using a variety of different devices. Thus usability is incredibly important. The app should be easily usable to a point that the user does not make too many mistakes to get to a certain point or to do a certain task. Jakob Nielsen has invented 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 suggestions from other experts, to identify and measure usability.

This system has two main concepts. Find new flashcard decks via geo-caching, and studying found flashcard decks. Having two main concepts means that the system should be designed in a way that keeps the user informed as to what they are doing all the time, so that they do not get confused. Nielsen (1995, para 2) states that users should always be kept informed about what is going on via appropriate feedback and within reasonable time. For this system, 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 what the user’s task, 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 in the app, the terms and language that is used must be simple to the user rather than complex system terminology as Nielsen (1995, para 3) states. For example, mentioning “caches” in a geocaching app is not always that helpful, as most users may not realise what geocaching is or even what a cache is. For simplicity, it is better to use phrases that any user, new to the subject area or not, will be able to understand. In this systems case, there is a good chance that if a user is learning a language, they may wish to change their display language. This will make the app much harder for them to use, but lots of people do this to help learn languages. Due to this, the system must use simple language with a combination of buttons and images. This also has an influence as to how they will use the app as well. As such, this will affect the order in which information will appear. Content will need to be displayed in a logical order to prevent confusion.

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 User Experience Team (n.d., para 13) where they say that if something looks the same then it should always work the same. Consistency across the system is important to prevent confusion for users.

The app should have an easy learning curve and should be very simple to cater for all ages. Shneiderman (n.d., para 10,11) states in his Heuristics that reducing short-term memory load is a must as you want to avoid interfaces where users must remember information between different displays. Now, this will be in that GUIs to certain extent as the design of the flashcard quiz requires users to have to remember information. However, this does not mean that use of the rest of the system should be taxing on the user. By reducing short-term memory load in this way throughout the rest of the system, users should find learning new words or grammar rules when studying a lot easier. This is because they will have less things to remember, meaning they have more potential to learn more.

The Android User Experience Team (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 read and understand than longer sentences. As the system is aimed at getting users to learn foreign languages with individual sentence flashcards (gained from caches) this guideline should also be taken into consideration for the apps cache content as well as system objects.

**Operationalize usability concepts**

In Usability Metrics (2001, para 5) Nielsen lists the most basic measures of usability which he lists as follows:

* Success rate (can users perform the task?)
* Time taken to complete a task
* Error rate
* User satisfaction

He also mentions other possible metrics such as the amount of times users need to backtrack to find the correct window/page. He then goes on to discuss comparing two designs and how to quickly tell if a new design has improved in usability. He recommends taking tasks that the user must do then to record how long it takes the user to complete the task.

To do this you would gather the data for each design first then, for each one, add up the time taken so you get a number which he calls “how long it takes users to do stuff” for each design. From here you can simply calculate a percentage improvement based on the time difference. He also means though that this can be misleading as if certain tasks are not performed as often but improve in efficiency then these tasks could skew the results. When some tasks are performed more than others, it is better to work out percentage of improvement separately first and then get a geometric mean of the tasks percentages. By doing this you then get a fair “usability score” which can tell you just how higher or lower usability is for a new design.

**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 huge variety of accessibility issues, especially for the mobile version. There are many different types of disabilities out there and many are affected by "small screens", for example, partial blindness. In this case the user would prefer a large screen to see the app on and may own a tablet for this purpose. However, if this app is not optimised for accessibility and the buttons are still small or the text is just as small as a normal mobile version, then the user will have a hard time using it. The user will be using the mobile app outside which could have some environmental issue which should be included. For example, glare could affect the ability to see the screen so this may affect the colour scheme.

**MENTION LIMITATIONS BY 3G and WIFI ETC, REFERENCE HOW MANY PEOPLE HAVE 3G IN THE UK**

5) High fidelity prototypes for both Mobile and Desktop applications (must use Visual Basic for the Desktop prototype)

6) Quick and dirty empirical re-evaluation

7) Interface Specification modification

**Section 2**

**Initial Interface Specification**

**Description of how the System Works**

The mobile designs are meant for Android devices therefore the user has access to the back, escape and switch app buttons that comes with the Android OS so that they can navigate the application. Here is a quick description of both the mobile and desktop versions.

The first thing the user must do on either version is to register an account (if they don’t yet have one), or sign in. Once they have signed in they will be either taken to the home page (mobile) or to the find decks page (desktop). Instead of a “home page” for the desktop version, we have a simple toolbar at the top where the user can access the three main features of the system.

For the home page on the mobile version, we have a simplistic layout that gives the user everything they need in one small place. They have three main options here which are to study or find decks and to view a map. If the user clicks on the study option, then they will be taken to a page that has a list of decks that they have already found. An example of this page can be found in the appendix below (Figure \_).

From the study decks page, the user simply clicks on the deck and they can begin studying. From 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 card. Native audio plays during the pressing of the “Show back” button and the user will be able to look at the translation and meaning of the words in the sentence (as well as any other notes). They will then grade themselves as to how well they understood and read the sentence. Once they click a grade, a new sentence will appear. They will then continue this process until they click “back” on their Android device or until they have run out of cards to learn for today. The application will schedule cards per the Spaced Repetition technique for optimal learning efficiency.

If the user clicks the “Find decks” option, then they will be taken to a list of decks that are close to them. This will only include a certain number and if they wish to see more then they must click on the map at the bottom of the list. When they click on an option in this list they will be taken to an individual information page for that deck. An example of this can be seen in the appendix below (Figure \_).

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

If the user clicks the pie chart at the top of the application, then they will be taken to the progress page. If they click the circle created by two arrows, then a sync window will appear and sync the users’ progress with the server. If the user clicks the button at the top which shows different languages, then the user will be taken to their language settings. Examples of all these pages can be found in the appendix (Figures \_ to \_).

The desktop version also includes some features for more experienced users such as shortcuts. For the above example “-“, “+” and “scroll wheel” can be used to control the map. When it comes to studying flashcards on the desktop version, users can use CTRL+Z to go back to a card and use the number keys to quickly answer cards.

The main differences between the desktop and mobile version is that the desktop version implements a larger navigation bar, whereas the mobile version uses a main menu to reach the same effect, allowing the user to easily navigate between pages on either version.

**Site map**

The below site map gives a brief overview as to how each page is connected via each other. Each page on the 3rd layer can be accessed by each other on the desktop version whereas the mobile version just makes use of the built-in Android “back” button to go back to the previous page.



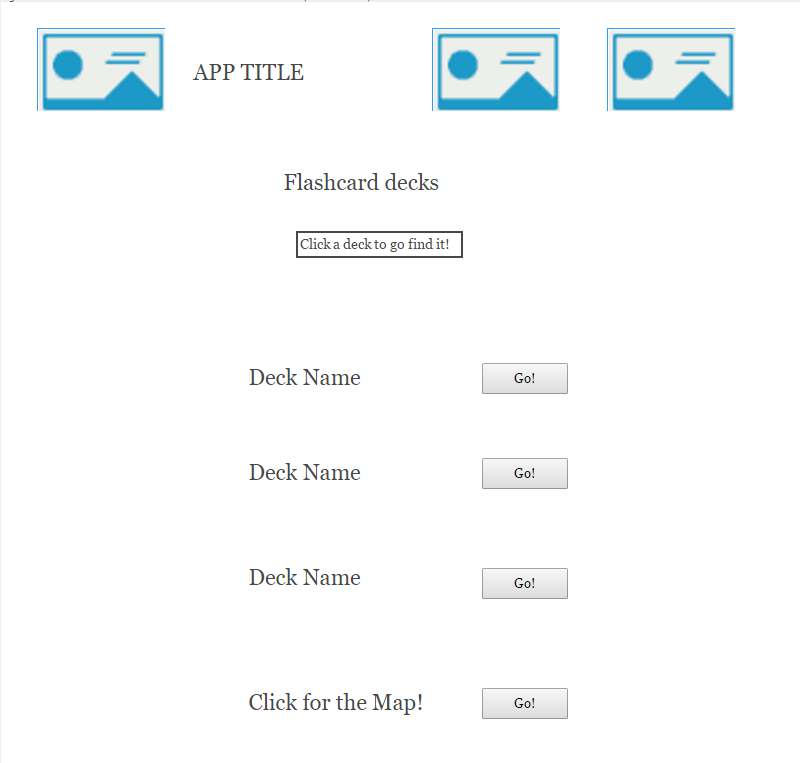
*Figure 1 - Site map of both systems (check description above for details)*

**Wireframe of Main Components**

The Android User Experience Team (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.” Based on this guideline the system will implement logos and small images that visually represent cache locations on a map, ticks for confirmation, graphs for progress and two arrows in a circle to mean “sync”. 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**

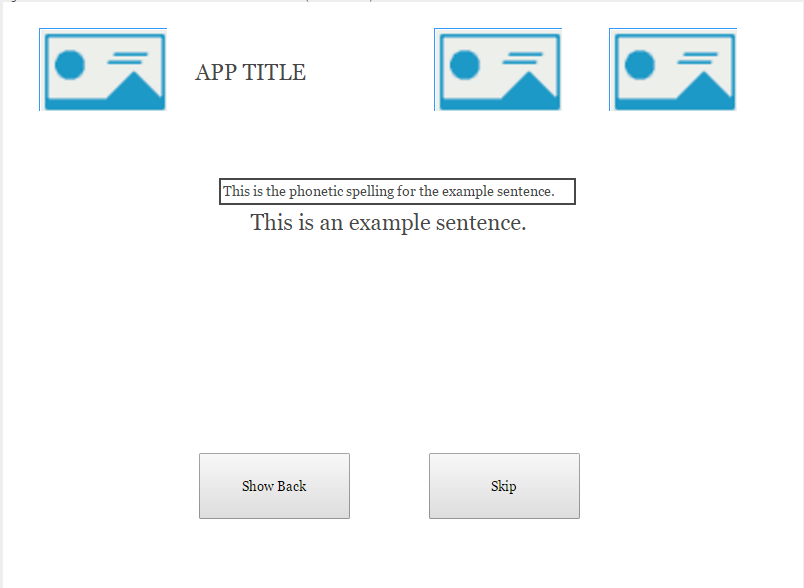
****

The Android User Experience Team (n.d., para 7) also observe that using “short phrases with simple words” is very important as “people are likely to skip sentences if they’re long.” This is an incredibly simple rule but is important to remember when writing any form of explanation on how to use certain features, button names, and even content itself.

Lots of different types of apps will use buttons or linked text to take a person to the next page. Both the text and the 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 will use Android‘s “Roboto Medium“ font were as the desktop version will make use of Microsoft’s “Microsoft Sans Serif” font. All text will be at a readable size (no lower than size 14), titles and button labels will be larger to make them stand out more.

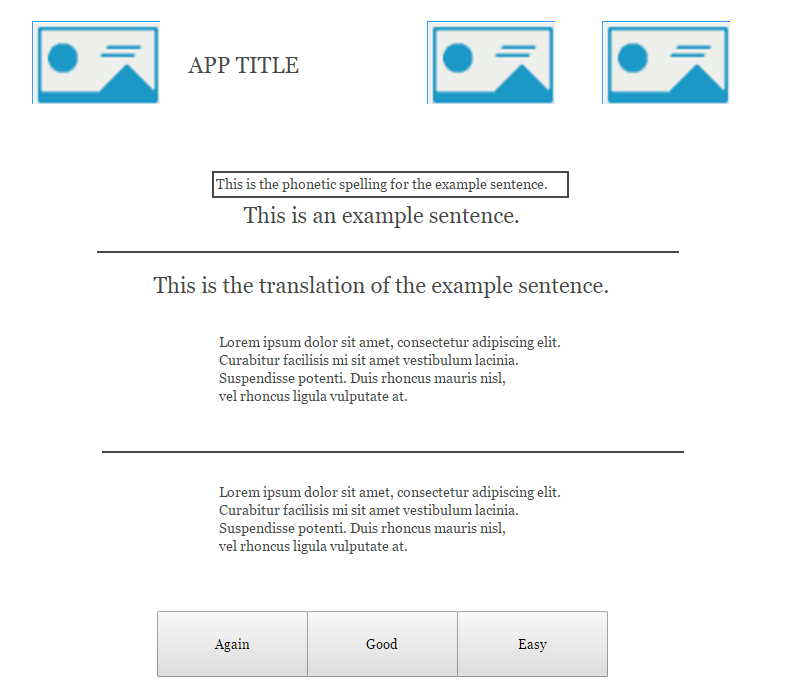
**Example of Studying a Flashcard (Front and Back)**



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 main sentence. Each language has methods for explaining its pronunciation so this will be included to aid users. This is more of a content choice than anything but it does influence usability. Including this makes it easier for beginners to learn how words are read and pronounced, reducing cognitive strain. The application should be able to teach a variety of people that are at different levels. By including this, we increase our accessibility and allow for a wider audience.

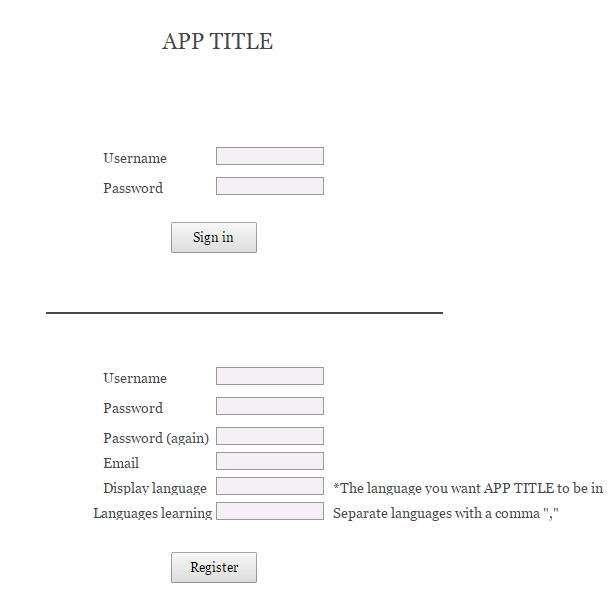
These top three buttons and the title should be kept the same through every page in the **mobile** application.

Simple, large and clearly labelled buttons so that the user knows what each one does and what will happen when they click them. When the “Show Back” button is pressed, native audio for the sentence is also played. “Skip” will move the user on to the next sentence.



The back of the flashcard. Gives a translation of the original sentence, definitions of each word and grammar rule and any extra notes. The original front of the card is kept at the top so that the user can reference it.

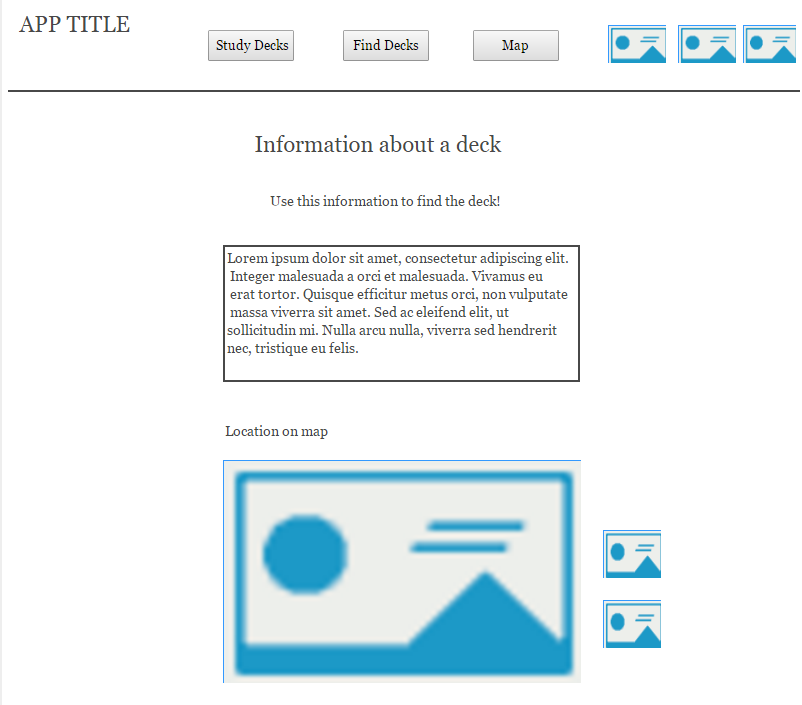
Buttons here should be clearly labelled so the user knows how to grade themselves. A short line of text should also be included as a short explanation.



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

Helpful text hints that speak the users’ language, as recommended by Nielsen (1995, para 3). As “display language” might confuse some users, it would be best to add a hint explaining in a bit more detail what the system wants from the user. The last box has a hint incase users need to add multiple languages.

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 here by giving instant feedback to the user as they type in to each 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.



This is where the information such as coordinates will go.

This is the map which will show the user where the deck is. It will contain standard images to identify the user’s location and the decks’ location to reduce cognitive load on users, as using text would make it hard to use. These should be easily understood by the user and will allow them to quickly understand what is going on.

For the desktop version, this “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 easily and so that they know where they are relative to the rest of the program.

These will be small magnifying glass “zoom in” and “zoom out” images. The magnifying glasses tell the user that they can zoom in and out of the graph.

**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 the actual prototype testing a lot easier and will allow the tester to gain more accurate data. Taking the wireframe as a base, the prototypes for mobile has been made (see below). Balsamiq is also great for creating similar screens as you can just copy and paste them over and make the slight changes you need to make. Once all the screens are created you can print them and ask someone to test them very easily, get data, then plan changes ready for the high-fidelity versions.

Since the creation of the wireframe, more research into usability concepts has been done and some slight changes have been made, such as extra feedback to the user for forms. This is a small change but it is worth noting now as the feature is not on the wireframe. As well as this, any images and interactive features have been inserted so that the user can get a better understand and “feel” of the app during testing. This should provide slightly 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 due to its simplicity and ease of use. The same process was conducted as with the mobile low fidelity. 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. This can then be used as test with some users to see how usable the current prototype is.

|  |  |
| --- | --- |
| **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.* | |

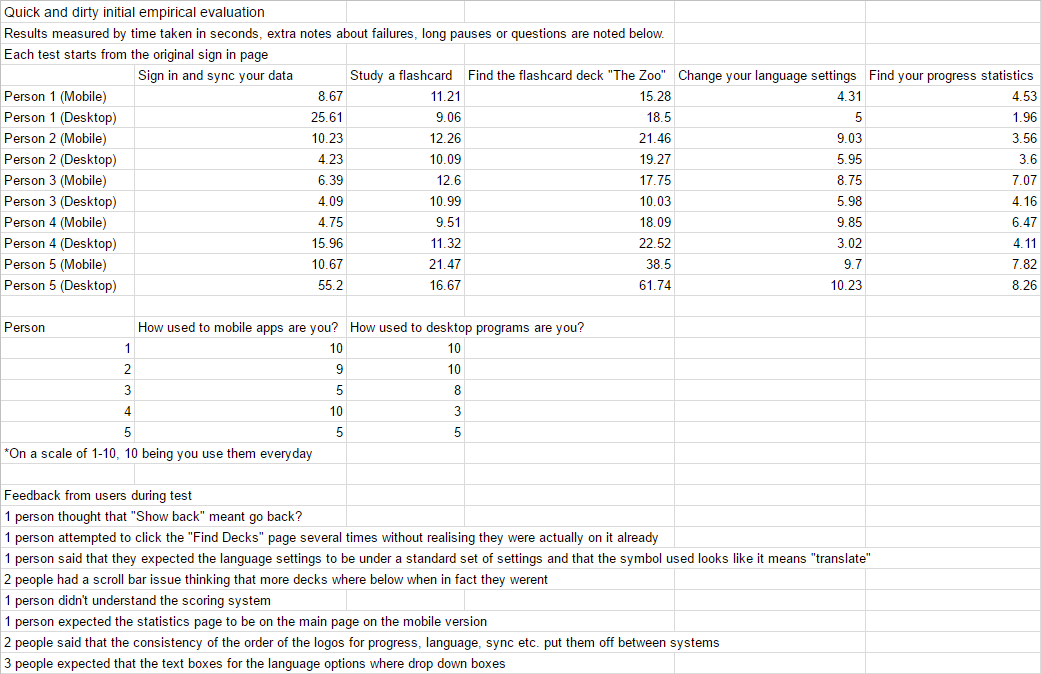
**Quick and Dirty Initial Empirical Evaluation**

Once the low fidelity prototypes where made, tested was taken out to see how usable they both where. Using the techniques in “Operationalize Usability Concepts”, 5 users where chosen to test both prototypes. The prototypes where tested with these sample users and the results were recorded. During the testing the users were given set tasks and the time taken to complete these tasks was measured. 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. The where each assured that it was the prototypes usability that was being tested and not the themselves, while also encouraged to keep talking while they perform each task. Making tests do this, as recommended by, \_\_\_\_\_\_\_\_\_\_\_\_\_ 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 they user has gone wrong and therefore where the usability of the prototype has gone wrong.

After this process was complete, analysis was done on how to improve the results of the tests that have been conducted, by making changes to the prototypes.

After the test results where analysed, the making of the high-fidelity design, with improvements, was started. Once a high fidelity has been made, tests for the same tasks will be conducted and results recorded. Changes will then be made and updates to the prototype to increase usability will be considered, implemented and the prototypes will then be tested again.

*Empirical evaluation results*



**Analysis of Results**

It appears straight away 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 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. Because of this, it might be a good idea to include a feature that automatically syncs.

During the process of finding and studying a flashcard, most users coped fine in finding and learning it in decent time, in both prototypes, especially considering this is the first time they have seen the app. 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”, they could view and find the information for “The Zoo”. This could be improved by hinting to the user in some way that the map exists and that they can use it to find other flashcard decks.

Both changing language settings and finding the progress page were very easy for all users and was achieved in 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 this buttons where 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.

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

By implementing Nielsen’s method of quickly calculating the “success” of a design, I have quickly created a table with the average time of each task for mobile and desktop. This will be compared with a similar table for high fidelity later.

|  |  |  |
| --- | --- | --- |
|  | **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 |

This shows at this stage that the mobile prototype is currently slightly more usable than the desktop version. This will be tabled will be created again later after the tests for the high-fidelity prototypes, so that we can see the difference between the two and can tell whether the high fidelities have improved or not.

**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 change 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 on the navigation bar in both the desktop and the mobile version.
* 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 just a sign in and register form for mobile and instead gives the user the choice to sign in or register which 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 functionality, as at this stage a lot of the functionality isn’t yet fully implemented but is instead meant to look like it does for usability testing purposes.

**Mobile High Fidelity**

The mobile high fidelity has been implemented with proto io which allows quick development of prototypes and designs. You can easily add interaction, effects and add more complex features that other software like Balsamiq doesn’t have. Proto io also allows you to download the source code to allow you to work on your prototypes if you want to develop them further. Below you can find screenshots of the actual mobile low fidelity. 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, and the progress bar for syncing. The reason for doing so is to keep users using features and functions that they are already used to as recommended by \_\_\_\_\_\_\_\_\_\_\_\_ where they state that \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. 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.

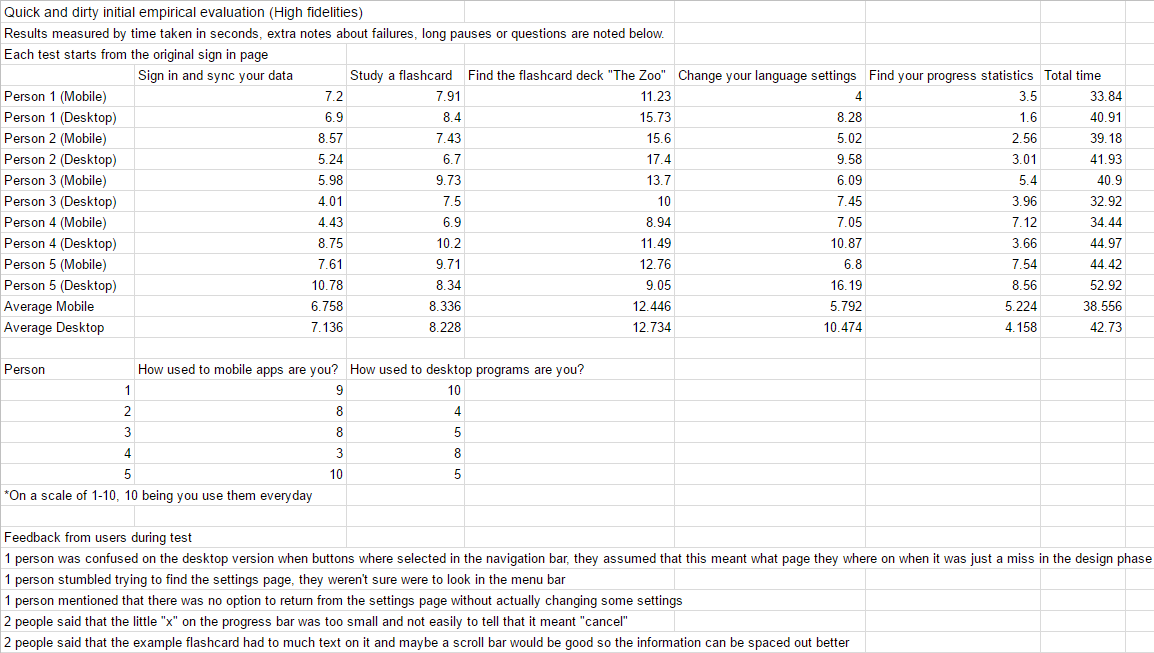
*Here are all the designs for the mobile high fidelity prototype*

|  |  |  |
| --- | --- | --- |
| **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 being 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 generic Windows based programs, which is great for creating a high-fidelity prototype as the user feels like the are using a normal system. Visual Studio and VB.NET are quite capable of producing fully fledged programs which makes this a great environment to create a prototype in. It is simple to create a prototype and the fact that you can then develop said prototype into a real system makes Visual Studio a great choice for 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 and hiding/showing objects and 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 were 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.* |  |



*The rest of the designs can be found in the appendix.*

**Section 3**

**Evaluation**

Section 3 20%

1) Critical evaluation of, and reflection on your process, and the role of prototyping in empirical evaluation.

We expect you to upload

(as a single archive in ZIP format, named with your student ID, max size <100Mb)

1) Report

2) Software developed (demonstration in class)

3) Evidence of **paper prototypes** and evaluation

**CHECK ALL YOUR REFERENCES**

**CHECK FOR DUPLICATE TEXT**

**IF TOO MUCH TEXT, CONDENSE USABILITY CONCEPTS**

**Write in 3rd person / we actively**

Works Cited

"10 Heuristics for User Interface Design: Article by Jakob Nielsen." *10 Heuristics for User Interface Design: Article by Jakob Nielsen*. Web. 01 Mar. 2017.

"Android Design Principles." *Android Developers*. Web. 27 Feb. 2017.

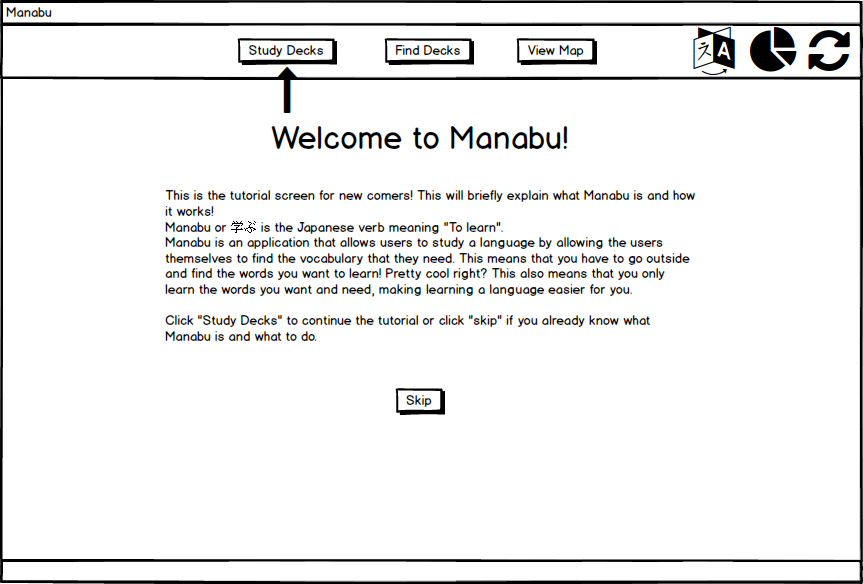
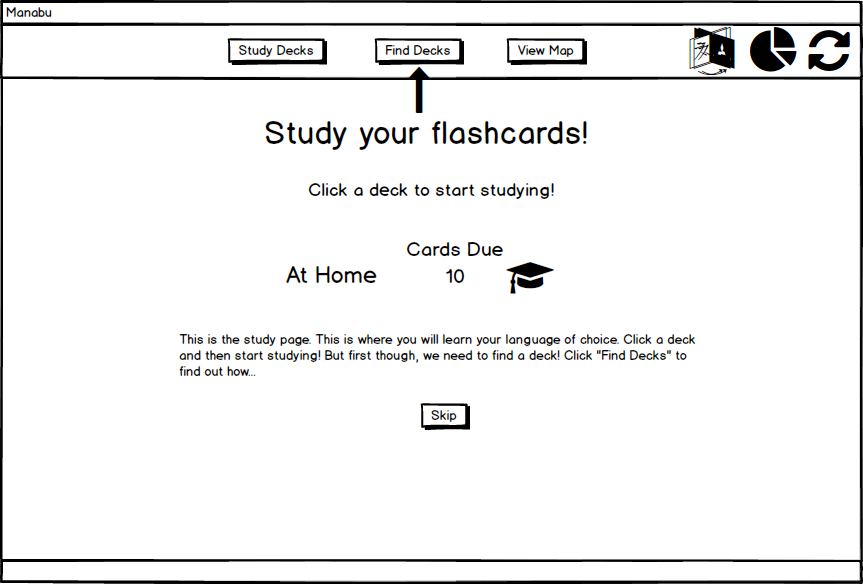
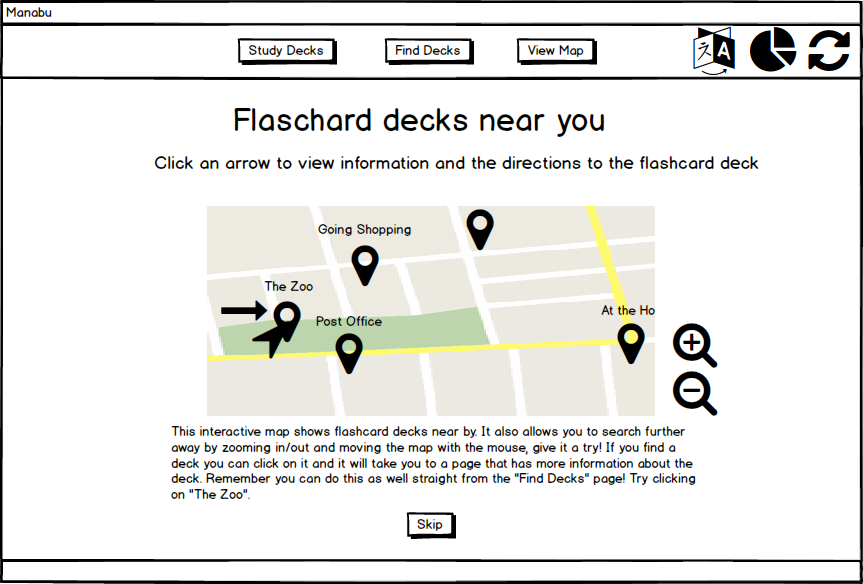
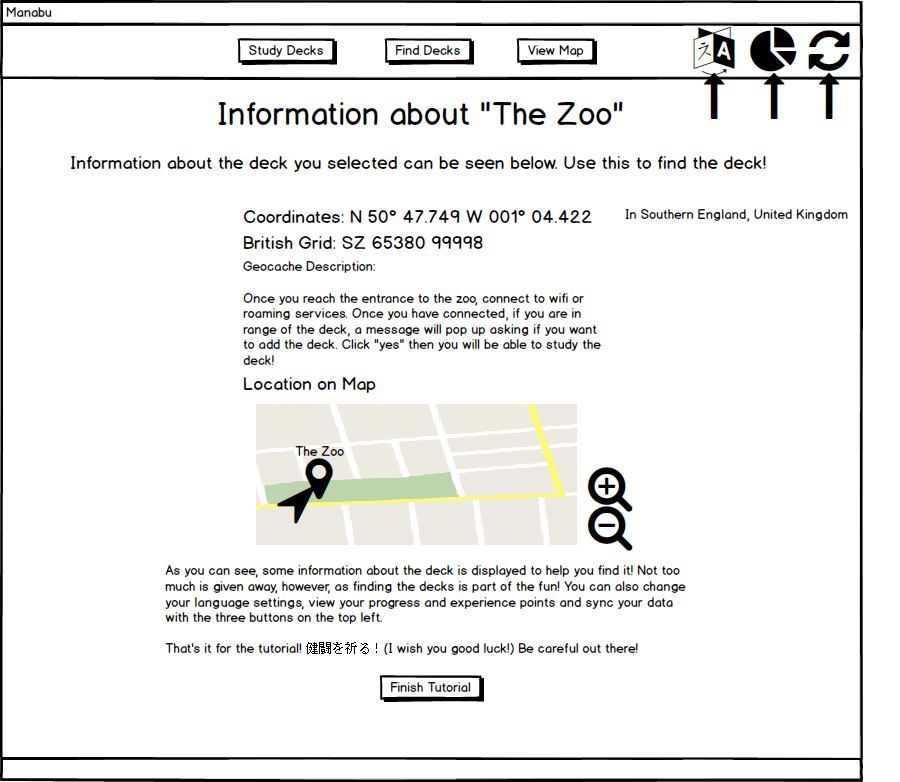
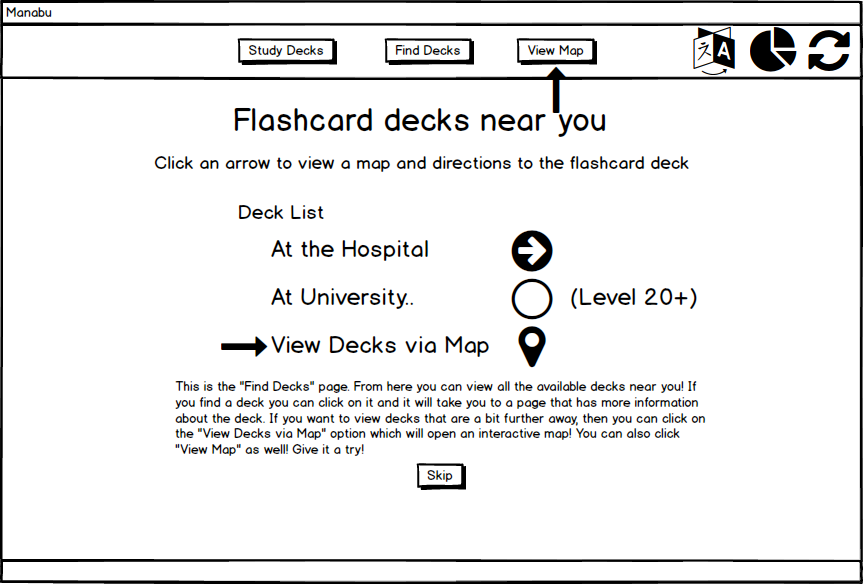
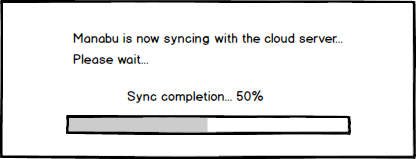
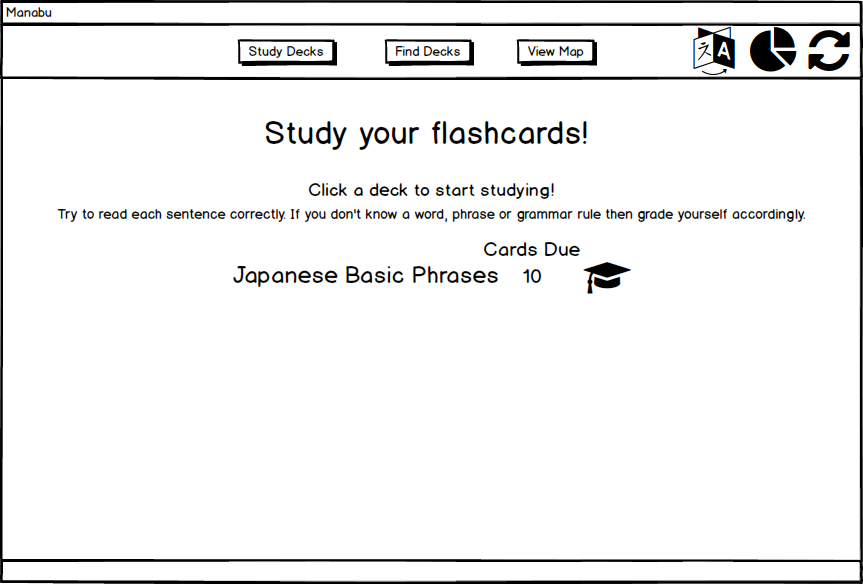
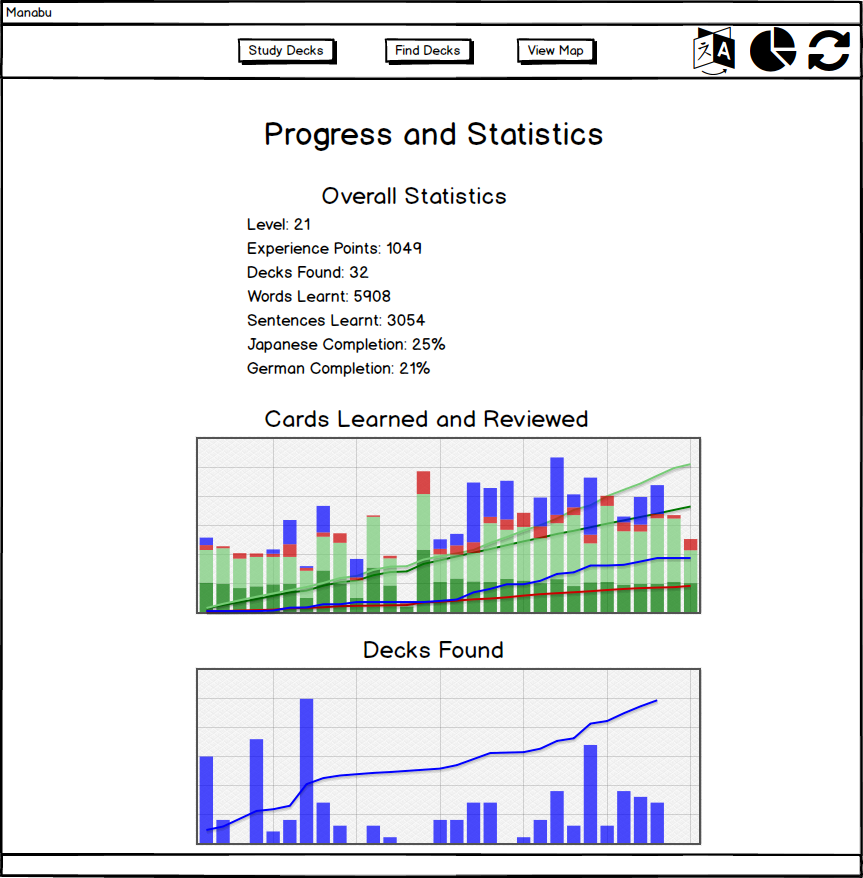
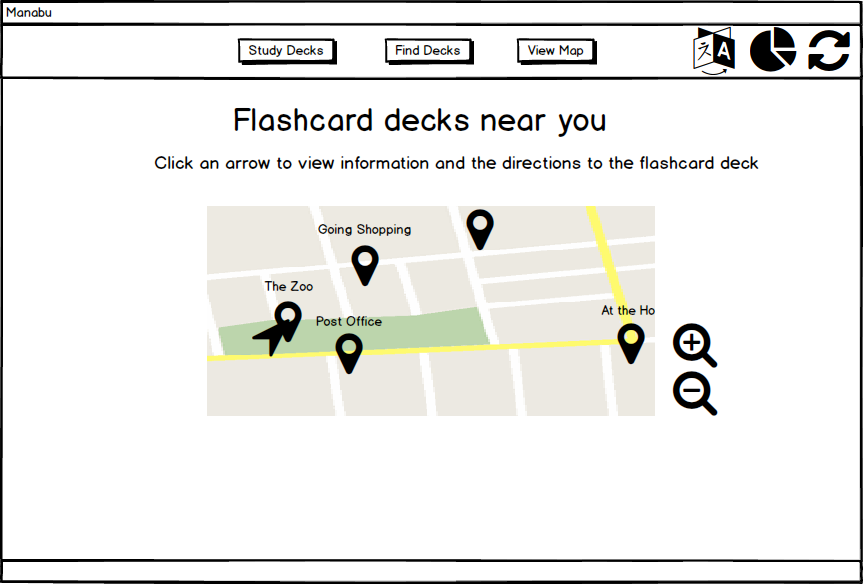
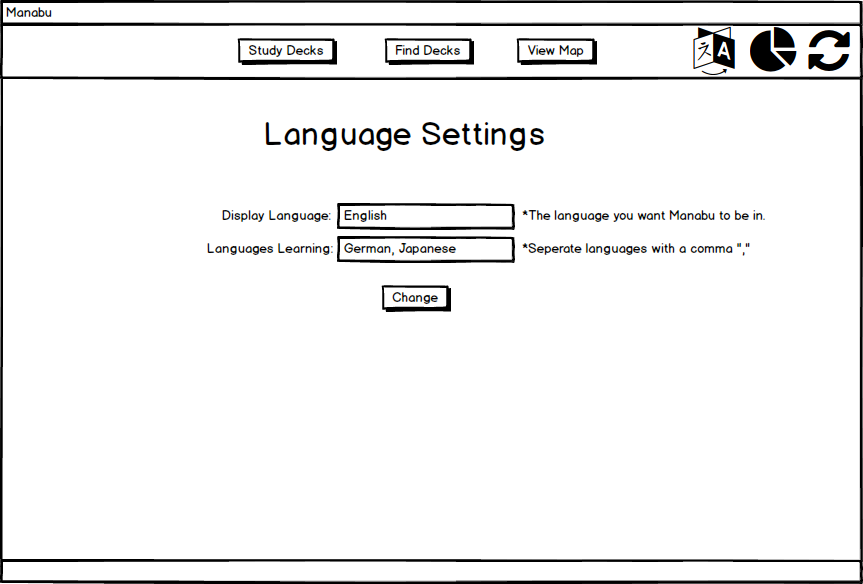
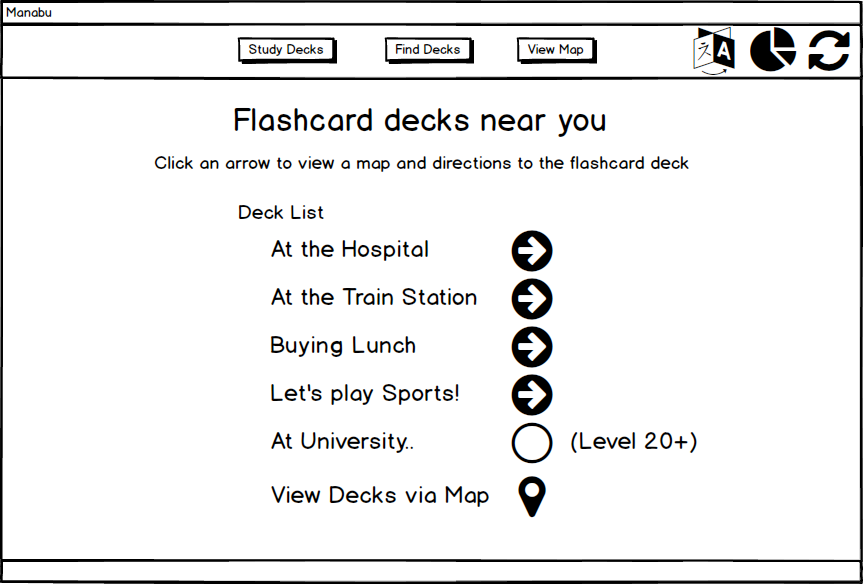
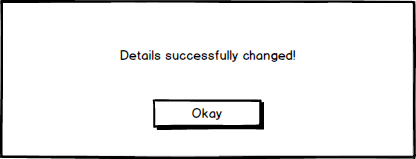
"Ben Shneiderman." *Ben Shneiderman*. Web. 01 Mar. 2017.

"Ben Shneiderman." *Ben Shneiderman*. Web. 03 Mar. 2017.

"Usability Metrics." *Usability Metrics*. Web. 03 Mar. 2017.

**Appendix**

**The Rest of the Desktop Low Fidelity Designs**



**The Rest of the Desktop High Fidelity Designs**

