Study and Repeat
A desktop application for spaced repetition

Marco Benelli
marco.benelli@stud.unifi.it

Luca Bindini
luca.bindini@stud.unifi.it

January 17, 2022

Abstract

Study and Repeat is an application that aims to integrate the spaced repetition technique into your study method. Through it, a student or any person wishing to learn a certain topic can do it in a simple and intuitive way, optimizing time and avoiding long study sessions in the last days before the exam.

In this article we will illustrate the entire development process of the application itself, starting from the preliminary phase of needfinding up to the implementation and the various test phases.

1 Introduction

When preparing to study for any test or exam, most people tend to leave a heavy workload in the final part of the preparation, making the days before the exam itself extremely loaded in terms of the amount of knowledge to review or, even worse, having to study for the first time.

1.1 Spaced repetition

Spaced repetition is a memorization technique that involves studying a certain concept and repeating it in several separate instants of time. This technique can be applied to any form of education, from middle school students, starting their first years of study, to university students preparing for an exam.

Spaced repetition is extremely effective in improving long-term memory compared to other traditional study methods which tend to make you forget the concepts you have just learned very quickly. But the most important aspect is that thanks to this technique the total study time for a given topic is greatly reduced.

To use this technique, the most historically used method is the paper one, also known as the *flashcard method*.

1.2 Flashcard method

The flashcard method consists of a double-sided deck of cards, where on one side there is the question while on the back side there is the corresponding answer.

Once a card has been drawn from the deck, the user tries to answer the question and if he answers correctly he will enter it in the list of cards that he will have to review the following day, if he still answers well he will enter it in the list of questions to review in a more "distant" day and so on. If, on the other hand, the answer to the question is incorrect, it will be put back in the current day queue. In other words, the questions we know how to answer will be asked less and less frequently, while the cards we don't know very well will be drawn more often.

In *Study and Repeat* what we wanted was to replicate this manual system in a digitized form that would introduce some comforts due to the fact of using a digital and paperless system.

2 Needfinding

The first thing we should do is the needfinding. This means interviewing potential users of our application and studying their habits to be able to suit their needs. From the interviews we will arrive at categorizing our users into a small set of personas, which we will find scenarios for. Once we have the scenarios, we can derive the requirements for the program, so that we will have a guideline to follow when developing the application.

2.1 Initial survey

The first thing we did was a survey to understand how our potential users currently study and whether they would benefit from an application like ours.

The survey was administered to 16 people in total. All of them were students, 13 of which were in college, meaning they would have to memorize a lot of concepts for each exam, which means they could probably take advantage out of a spaced repetition approach. This was also evident by the fact that 38% of the interviewed people study more than 8 hours a day when nearing an exam, with another 38% of them studying around 6 hours a day.

Another thing we asked in the survey was how frequently the students took exams where they were evaluated based on solving exercises. This was a crucial question, since an app based on fixed questions and answers would not have been sufficient for that use case. However, we saw that none of the participants ranked this method of evaluation as the most frequent and only 31% of them ranked it in second place (out of three options). The other two options were asking whether students were asked to memorize concepts or analyze things critically, both of which were more popular than the exercises option.

Next we asked about what device people are most comfortable using for studying. We found out that 88% of respondents prefer a laptop, 6% prefer a desktop, 6% a tablet, and no one voted for a smartphone. To us, this was a clear indication that we had to target a desktop operating system. We can explain this result by the fact that for study sessions a small screen would not be comfortable after a while.

When we asked the students if they had heard of the spaced repetition method, only 19% of them said they had, with only 6% saying they actively use it. This was not a surprise to us, since this method is still not mainstream; however, we were happy to see that many people were willing to say that they could benefit from a different method than their current one, as we will see shortly.

The four next questions were 5 point choices, meaning that the respondents would have to say how much they agreed with a sentence on a scale from 1 to 5 (both included). Here are the sentences with the arithmetic means and standard deviations of the answers:

- I am happy with the effectiveness of my study method. (mean: 3.56, deviation: 0.89)
- I am happy with the efficiency of my current study method. (mean: 3.06, deviation: 0.85)
- My study method could benefit from the use of a digital device. (mean: 3.88, deviation: 1.15)
- I could benefit from integrating spaced repetitions in my study sessions. (mean: 3.81, deviation: 0.66)

From these questions we can see that people are moderately happy with the effectiveness of their study methods, however they are not really satisfied by their efficiency, meaning they would like to study for less time to get the same grades.

As a last question, we asked what features people thought were the most important for our application. The available options were: the possibility to share a deck with a friend, the possibility to add images to an answer, the possibility to add audio to an answer, and a way to view some sort of study statistics for a selected deck. The answers to this last question are summarized in figure 1. From these results it was clear to us that the most urgent features were the first two.

2.2 Personas

From the interviews taken we have obtained two different kinds of users summarized by the following personas.

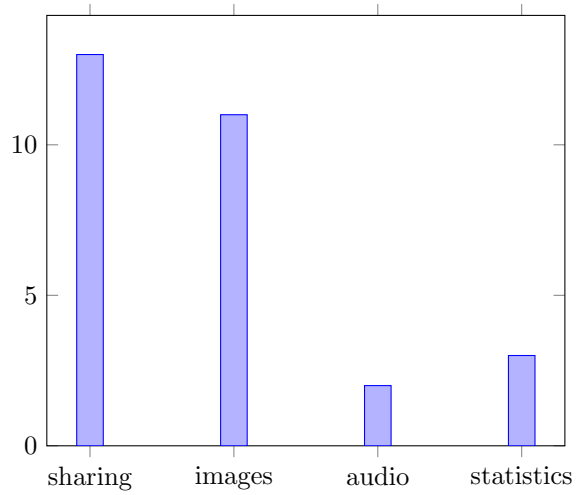

Figure 1: Histogram of the most requested features

Figure 2: Giulia

2.2.1 Giulia

Giulia, 22 years old, is a college student studying economics. The days before an exam she has long study sessions of up to eight hours a day. She isn't very satisfied with her study method, she would prefer to study less hours a day, over a longer period, but in a more efficient way.

Giulia loves using her laptop, which she often carries to university. She thinks that this is the most convenient digital device, especially for long work sessions. Notwithstanding the love for her laptop, she doesn't integrate it in her study method, which is primarily focused around her notes taken during classes.

Figure 3: Francesco

2.2.2 Francesco

Francesco, 24 years old, is a master degree student in engineering. His grades are well above average and this is mostly thanks to his many hours spent studying each day. He studies with his desktop PC from the professors' slides and notes but without following any particular study method and that's why he's looking for a better alternative.

He usually tries to help his fellow students but he doesn't have enough time for everyone and that's why he would like to help all of them in one fell swoop.

2.3 Scenarios

From thinking about what we want the main uses of our application to be, and keeping in mind the initial survey, we got to the following scenarios.

- Giulia is studying for an upcoming exam. She opens the application *Study and Repeat* and responds to the questions she is asked, some of them are new to her and so she has to look at the answer, however she confidently answers the other ones.
- Giulia needs to study for a new exam. The exam requires her to remember a lot of things. She decides it would be best to use the application *Study and Repeat* in her study sessions. She opens the application and creates a new deck of question-answer cards related to the subject. She adds cards to the deck, the answers are extracts from the professor's slides.
- Giulia has a new exam to study for, but she doesn't have the time to build her own deck. She decides to study from her friend's deck that she saw yesterday. She therefore asks him to export the deck and send it to her and, after importing it herself, she starts studying right away.
- Francesco has just taken an exam and got a really good grade. He thinks this is thanks to his well built deck on *Study and Repeat* and is sure that many other students would benefit from it. This is why he decides to export the deck and send it to his friend group, since he knows that many of his friends will be studying for the exam he just took.

2.4 Requirements

The requirements that we got from analysing the scenarios are the following.

- Users are asked a set of questions to which they have to respond. After responding on their own, they can reveal the correct answers. They have to let the application know the correctness of their answers. Depending on this the application will decide when to ask the questions again.
- The application must have the possibility to create new decks, with textual question-answer cards, and store them locally.
- Users can add images (e. g. professors' slides) to an answer.
- Users can share a deck simply by transferring the deck file to the other person's computer.

3 Implementation

Once we had a clear idea of the platform we were targeting and the basic requirements, it came time to actually code the application and design its user interface. Right before starting to code, we need to design some mockups for our interfaces and decide on the tools we should use for the job.

3.1 Mockups

We decided that before writing any code, we would also create a mockup, which is a realistic representation of the final interface. The mockup is often times underrated in its usefulness, since it can seem like a waste of time, however, it's quite the opposite, because when done well it can actually be a time saver.

We did all of our mockups in Qt Designer[1], a graphical user interface designer for Qt applications. We took this decision because Qt Designer can output a so called "Designer UI file", which is a file containing the information of the designed graphical user interface with the `.ui` extension, which can then be converted into Python code. The conversion is done using `pyuic`[2], a utility that is part of PyQt and allows us to generate `.py` files from `.ui` ones. In the end however, we decided against using `pyuic`, because we thought it would be more beneficial to us to learn how to write a widget structure on our own without the use of external tools.

We will now show the mockups for all the various interfaces. In figure 4, we can see the home screen. As you can tell, there are two decks in the application

[1] `https://doc.qt.io/qt-5/qtdesigner-manual.html`

[2] `https://www.riverbankcomputing.com/static/Docs/PyQt5/designer.html#pyuic5`

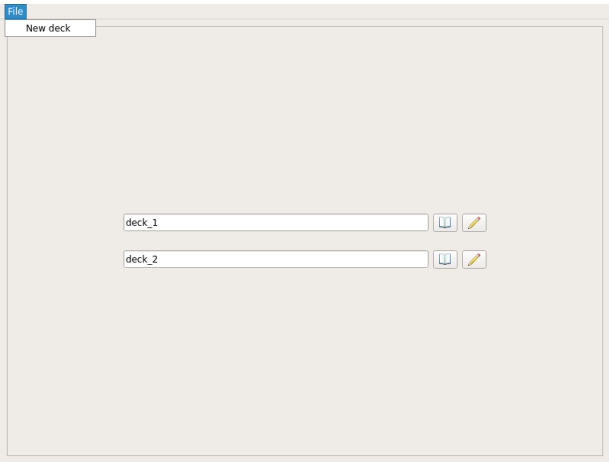

Figure 4: The mockup of the home screen of the application

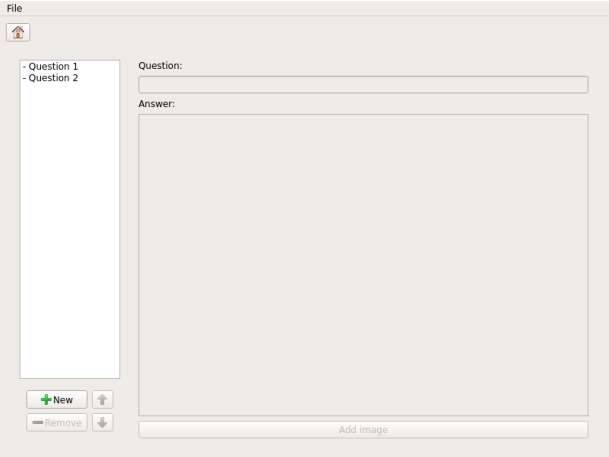

Figure 5: The mockup of the edit screen of the application

which have been named "deck_1" and "deck_2" for the sake of the example. Besides each name we have put two buttons, the first one is to study the deck and the second one is to edit it.

Next we have the edit screen in figure 5, where we can see the cards list on the left and the interface to edit each one of them on the right. On the left we also have many buttons for all of the various operations we may want to do, such as removing a card, creating a new one, or moving a card up or down.

The next two mockups are for the study screen. When you start studying a deck you will get the interface in figure 6, where you will be able to read the question and think of the answer. After you've done that, you will have to click on the "Show answer" button, which will take you to the interface in figure 7, where you can look at the correct answer and understand whether the answer you had thought of was right or wrong and communicate that to the application using the appropriate buttons. At that point you will be brought back to the previous interface with another question.

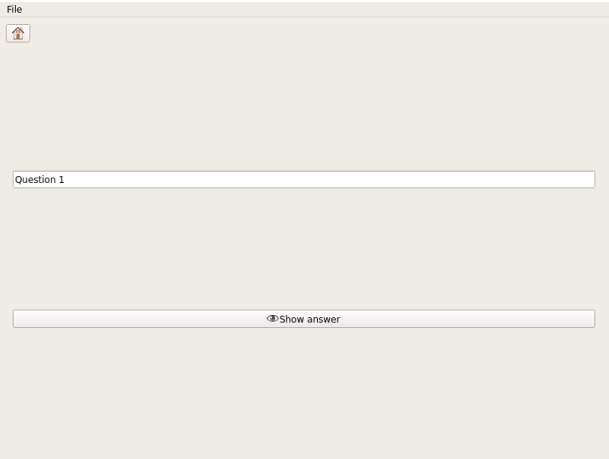

Figure 6: The mockup of the study screen of the application before the answer is shown

3.2 Used tools

The whole project was written in the Python language in its 3.9 version. In addition to the standard

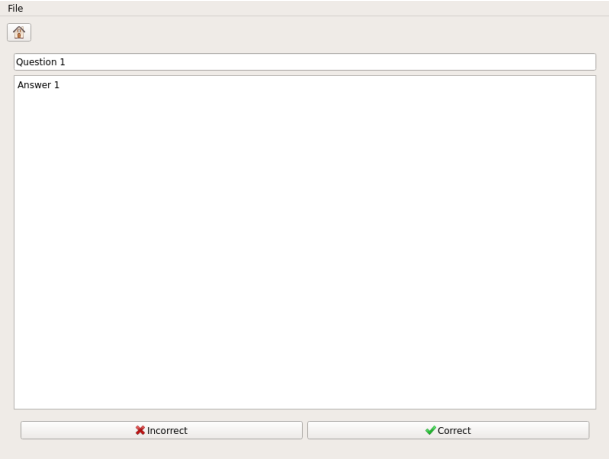

Figure 7: The mockup of the study screen of the application when the answer is shown

Python libraries, other external libraries have been used which are useful for the graphics part and beyond.

- `PyQt5`[3] is a comprehensive set of Python bindings for the famous C++ library Qt and in this project we used it to build the whole application GUI.
- `QDarkStyle`[4] is a complete dark/light style sheet for Qt applications and in this project we used it to have the dark style of the interface.
- `pyinstaller`[5] bundles a Python application and all its dependencies into a single package. The user can run the packaged app without installing a Python interpreter or any modules. In this project we used it to build the executables for all the main operating systems (GNU/Linux, macOS, Windows).

3.3 Code structure

The code is structured following the MVC (model-view-controller) structural design pattern, however the controller part is integrated directly into the view, effectively making the code structure into a model-view.

In particular, the code source is structurally divided into two different subdirectories and two other single `.py` files.

- `src/model`: it contains all the parts of the application domain, i. e. the classes and functions that define the behavior of the application itself.
- `src/view`: it contains all the parts of the GUI with the main window of the application and all the widgets in it.
- `config.py`: it contains all the declarations of global variables useful throughout the application (e. g. save directory, path where application icons are present etc.).
- `study_and_repeat.py`: it's the main script of the application and is responsible for instantiating the main window.

3.4 Functionalities

Study and Repeat aims to take the spaced repetition technique with the help of flashcards (as we saw in the introduction) to the next level. In addition to having the basic functionality to use the technique itself, there are other additions aimed at making the user experience more convenient and simple.

Below is a list of all the functionalities of the application:

Create deck in your local machine with the possibility to edit it whenever you want.

Add card to a desired deck. Cards can have an answer not only formed by text but enriched by images.

Move card to change the card order in a deck.

Remove card if it's incorrect or no longer needed.

Rename deck if it has an incorrect or outdated name.

[3] `https://pypi.org/project/PyQt5`
[4] `https://pypi.org/project/QDarkStyle`
[5] `https://pypi.org/project/pyinstaller`

Delete deck if it's no longer needed.

Reset deck if you want to clear spaced repetition temporal statistics.

Export deck in a `.tar` archive file. During the export phase you can choose whether or not to preserve the temporal information of the spaced repetition technique.

Import deck from a `.tar` deck file previously exported.

View statistics in particular the number of new cards (never studied) and the number of cards to be reviewed today.

3.5 User interface

The graphical interface of the application, created entirely through the `PyQt5` library, is the result of our initial ideas (exposed in section 3.1) modified and expanded thanks to the results obtained from the usability tests (in section 4).

The application is divided into 3 main screens:

Home is the main screen of the application (in figure 8). In this screen you can see the list of decks present locally with three buttons to start studying, modifying or possibly deleting it and two numbers representing the number of new cards and the number of cards for today. There is also another button to create a new deck. Through a double click on the name of a deck it is possible to change its name, both during the creation and modification of the name of a deck, it must comply with certain requirements (such as not containing special characters, spaces or dashes) that are verified through the use of a regular expression. At the top there is a menu bar through which the user can create, remove, reset (clear spaced repetition statistics), import and export a selected deck. During the removal of a deck the user will obviously be asked for confirmation, being a critical operation, the same thing happens when we try to import a deck that has a name of a deck that we already have.

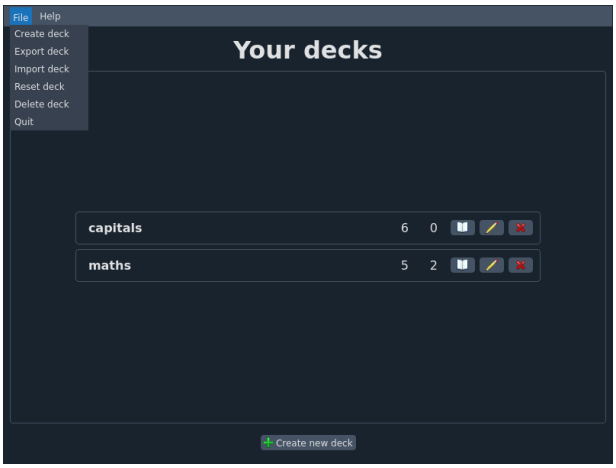

Figure 8: The home screen of the application

Edit deck is the screen where we edit a certain deck (in figure 9). We can then add cards, remove them and change their order in the deck through buttons placed at the bottom. The cards can be consulted through the side list. When we are in the phase of creating a card we can insert, if we want, images in the answer which will then obviously be displayed when we go to study it.

Study deck is the screen where we study a certain deck (in figure 10). It contains the question of the current card and through a button we can show the answer once we have answered to see if we have guessed right or not. Based on this you press the corresponding button "correct" or "incorrect" (in figure 11). Once the cards to be studied for the current day have been finished, the application will inform us that we have finished studying that deck for today.

4 Usability tests

The final step of any engineering process should be the testing phase. In fact, it shouldn't just be the final step, since the development should follow a cycle, alternating designing, prototyping, and testing. This is why we decided to do two rounds of testing instead

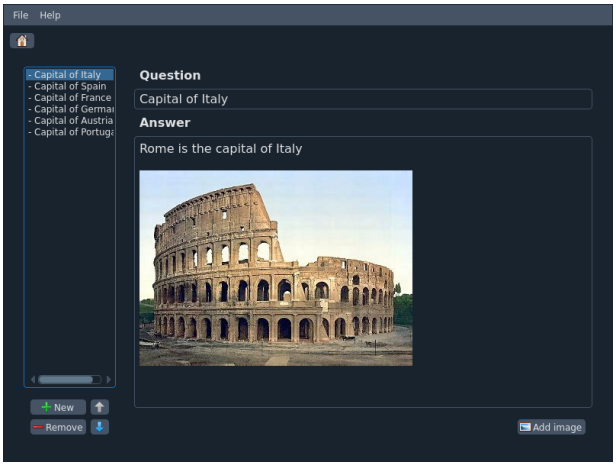

Figure 9: The edit screen of the application

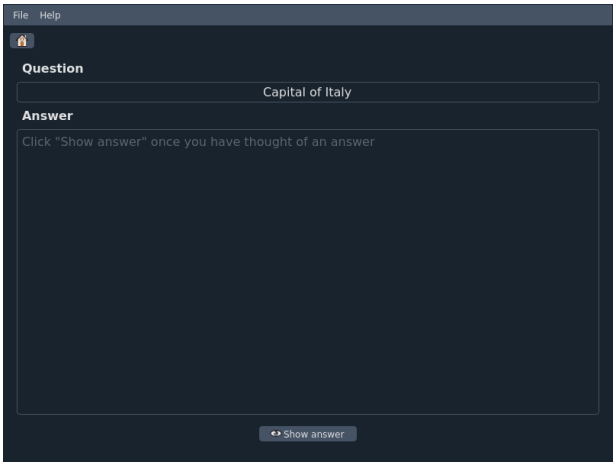

Figure 10: The study screen of the application before the answer is shown

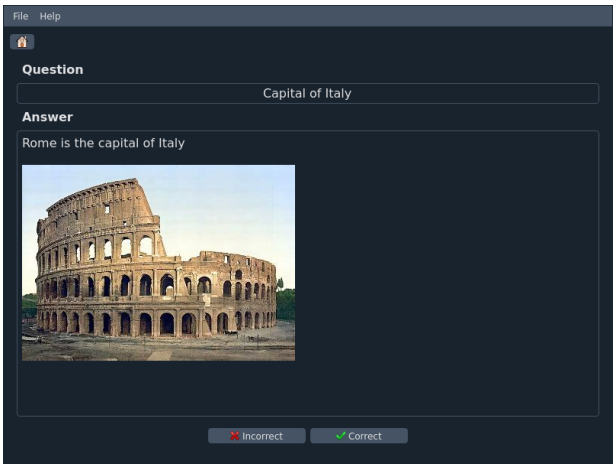

Figure 11: The study screen of the application when the answer is shown

of just one.

The test was structured as a series of tasks that the participants should complete (or at least try to). After each task, the participants would have to answer a SEQ, i. e. a single ease question, about how difficult or simple they found the assignment. Single ease questions are questions to which the respondent can reply with an integer from 1 to 7 (both included) that are often used to measure the user experience. They are a simple metric both for the respondent to understand easily and for developers to analyze the results and gives us the same quality of results as other more complex metrics.

After having completed all the tasks, the respondents were asked a few closing questions, again following the SEQ style, about the overall experience.

4.1 First test

When we administered the first test to our participants, we used a version of the application that closely resembled the mockups we showed in section 3.1. For the test, we used a deck that we created, that way we wouldn't waste too much time making the participant create a deck during the test.

As you will see from the tasks, the application at this point was also lacking some features like the abil-

ity to export and import decks. This was a problem because we wanted to give participants a pre-made deck. We therefore had to do it by placing the deck in the appropriate directory in the file system in between the second and the third task. However, as we found out during the pilot test, we had an issue with decks with images, since they referred to a path that on the respondent's PC was not the same as in ours (this was because of different account names). This was therefore fixed before doing the first actual test, but is a clear indication that pilot tests can often be a life-saver.

The first test round was administered to 7 people in total (excluding the pilot). Participants were asked to complete a total of 9 tasks and rate their difficulty and in the end they were asked 3 more general questions. In table 1, we have shown the all of the statements testers rated as well as the average and standard deviation for all of the tests. You might have noticed that some statements are written in a positive sense, while others in a negative one. Mixing up the questions like this is something that can sometimes help with the accuracy of the results.

From the table we can deduce that most of the tasks were pretty easy, but a few of them were not. In particular we see that the most critical tasks were executing the application, deleting a deck, and creating a new deck. In addition to those, the graphical appearance of the program was not well received, even though we were comforted by the high variance.

The critical tasks that were just mentioned were all addressed in the final release, taking into account suggestions from the testers. To make the application execution easier, we decided to ditch the portable program on Windows in favor of an installable implementation. This was done because almost all of the respondents were using Windows and were puzzled by the `.exe` file that would immediately launch the application. The installer was created using Inno Setup[6], a free installer for Windows programs.

Deleting a deck was made easier by adding a new button for each deck on the home screen that would delete the corresponding deck, instead of forcing people to go to the File menu. On a similar vein, we also added a single button to the home screen for the deck creation.

Finally, to make the application more pleasing to the eye, we put more care into the widget placement and dimensions, we changed the font sizes depending on the text's importance, and started making use of `QDarkStyle`.

4.2 Second test

The second test was done with the application which you've seen in section 3.5. In this final round we only tested the things that were critical in the first version as well as new features. The new features mainly involved the exporting and importing of decks. This new test round was administered to a total of 5 people.

In table 2, you can take a look at the statements and aggregate statistics of their ratings. This time we reduced the number of tasks to six, with three of them being entirely new. We were very glad to see that the old tasks were correctly addressed. In addition, the new ones got really good grades too, with the only exception of the exporting of a deck. This was not because of a difficulty in exporting itself, but rather because of the check box that the participants were faced by asking whether they wanted to retain the scheduling information. This was seen as confusing or unclear by many.

As with the first testing round, we finally asked the three general questions. Once again, we were pleased to see the grades improve, especially in the case of the graphical appearance. We are therefore confident that we were able to improve on all of the critical aspects.

5 Conclusion

In conclusion, thanks to the usability tests we were able to understand the critical issues of the application and in addition to those already listed in section 4, other small changes were made, such as inserting tooltips on the labels and buttons that give the user a further indication (in addition to the visual one of the

[6] `https://jrsoftware.org/isinfo.php`

icons) on what represents a label or what a certain button will do.

The development of an entire application guided by the user experience has allowed us to understand that many times the problems that the user poses are much different from those that the developers had anticipated.

Statement	mean	σ
Executing the application once downloaded has been easy.	4.86	1.25
Creating a deck with a few cards has been easy.	6.00	1.07
Studying a deck has been hard.	1.43	0.49
Editing a deck's card has been easy.	6.29	1.39
Reordering the cards in a deck has been hard.	2.57	1.29
Adding images to a card's answer has been hard.	1.43	0.49
Deleting a few cards from a deck has been easy.	7.00	0.00
Inserting a card in the middle of an existing deck has been easy.	6.00	1.07
Deleting a deck of my choice has been hard.	3.00	1.41
I found it easy and intuitive to interact with the application.	6.14	0.35
I didn't appreciate the graphical appearance of the application.	3.57	1.50
I think I could benefit from using this application.	6.14	0.64

Table 1: The summary of the results of the first test. For each statement we are showing the average rating as well as the standard deviation (σ).

Statement	mean	σ
Executing the application once downloaded has been easy.	7.00	0.00
Creating a deck with a few cards has been easy.	7.00	0.00
Deleting a deck of my choice has been hard.	1.00	0.00
Exporting a deck of my choice has been hard.	2.20	0.40
Bringing a deck back to its original state has been easy.	6.40	0.80
Importing a deck from a file has been easy.	6.80	0.40
I found it easy and intuitive to interact with the application.	6.80	0.40
I didn't appreciate the graphical appearance of the application.	1.60	0.49
I think I could benefit from using this application.	6.20	0.40

Table 2: The summary of the results of the second test. For each statement we are showing the average rating as well as the standard deviation (σ).