Master Thesis

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Computer Graphics and Interaction

Platform for fast selective A/B testing

Bc. Sebastian Voráč

Supervisor: doc. Ing. Zdeněk Míkovec, Ph.D.
Field of study: Open Informatics
Subfield: Human-Computer Interaction
May 2019

ČVUT
ČESKÉ VYSOKÉ
UČENÍ TECHNICKÉ
V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Voráč** Jméno: **Sebastian** Osobní číslo: **420366**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačové grafiky a interakce**

Studijní program: **Otevřená informatika**

Studijní obor: **Interakce člověka s počítačem**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Platforma pro rychlé selektivní A/B testování

Název diplomové práce anglicky:

Platform for fast selective A/B testing

Pokyny pro vypracování:

Vytvořte platformu, která umožní uživatelům platformy získat zpětnou vazbu od uživatelů odpovídajícím vybrané cílové skupině. Získávání zpětné vazby bude zajištěno metodou A/B testování prostřednictvím webové aplikace. Během návrhu platformy se zaměřte se na to, aby byl celý proces získávání respektive dávání zpětné vazby velice rychlý, jednoduchý na použití a dostupný komkukoliv na světě, kdo má připojení na internet bez technických znalostí či předchozích zkušeností s uživatelským výzkumem. Navržený systém implementujte ve formě vysokoúrovňového prototypu. Celý proces návrhu a implementace systému bude mít iterativní charakter a bude průběžně ověřován formou uživatelského testováním v souladu s metodikou UCD (User Centered Design).

Seznam doporučené literatury:

Dan Siroker and Pete Koomen. A/B Testing: The Most Powerful Way to Turn Clicks into Customers
Cooper, Alan, Robert Reimann, David Cronin, and Christopher Noessel. About face: the essentials of interaction design. John Wiley & Sons, 2014.
Bill Buxton: Sketching User Experience, Elsevier Inc., 2007
Colin Ware: Visual Thinking for Design, Elsevier Inc., 2008

Jméno a pracoviště vedoucí(ho) diplomové práce:

doc. Ing. Zdeněk Míkovec, Ph.D., Katedra počítačové grafiky a interakce

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **18.02.2019** Termín odevzdání diplomové práce: **24.05.2019**

Platnost zadání diplomové práce: **19.02.2021**

doc. Ing. Zdeněk Míkovec, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Ing. Pavel Ripka, CSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

.
Datum převzetí zadání Podpis studenta

Acknowledgements

I would like to express my sincere appreciation to doc. Ing. Zdeněk Míkovec, Ph.D. for his valuable suggestions during the time of leading my work. His quick constructive feedback helped me to keep moving forward. I would also like to thank my family and friends for an enormous amount of support they gave me.

Declaration

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškerou použitou literaturu.

V Praze, 1. května 2019

Abstract

This master thesis deals with design and implementation of the platform which enables to get quick feedback from other users corresponding to the selected target group. Obtaining or giving feedback is provided by the A/B testing method. The platform is implemented as a single-page web application using the React.js javascript library. This thesis has a great emphasis on user experience (UX) and usability on mobile devices. These goals have been achieved through the development of the platform in accordance with the UCD (User-Centered Design) methodology.

Keywords: platform, A/B testing, user feedback, UX, web, single-page application, user centred design, javascript, React.js

Supervisor: doc. Ing. Zdeněk Míkovec, Ph.D.

Abstrakt

Tato diplomová práce se zabývá návrhem a implementaci platformy, která umožňuje získávat rychlou zpětnou vazbu od ostatních uživatelů odpovídajícím vybrané cílové skupině. Získávání respektive poskytování zpětné vazby je zajištěno metodou A/B testování. Platforma je realizována jako webová single-page aplikace pomocí javascriptové knihovny React.js. V práci je kladen velký důraz na uživatelskou zkušenost (UX) a použitelnost v mobilních zařízeních. Těchto cílů je dosaženo díky vývoji platformy v souladu s metodikou UCD (User-Centered Design).

Klíčová slova: platforma, A/B testování, uživatelská zpětná vazba, UX, web, single-page aplikace, user centred design, javascript, React.js

Překlad názvu: Platforma pro rychlé selektivní A/B testování

Contents

Figures

Tables

Chapter 1

Introduction

Receiving constructive feedback is part of our everyday life. Let us just focus on the case, where we want to hear the feedback and specifically ask for it. In the business field, marketers know that the success of the product respectively sales depends on what their potential customers think about it. They invite the group of selected people matching their potential clients and asks the group for some feedback. This is not very fast and often even not so cheap process [23]. Therefore receiving feedback by this method is used mainly by companies and more mature start-ups. When we just cannot decide if X is better than Y, the feedback we want to get is pretty straight-forward and there is no need for more complex user questionnaire.

That is a common case even in our personal life. We might want to know what the best dress for an important meeting is or to choose the best logo of our new company out of these four variations. Unlike in the business field, we ask our friends and relatives rather than a selected unknown user group. The question is simple - are they really the best user group to provide us the most valid feedback? Is there a better way?

This work will examine the need for a solution to the problem of quickly providing and receiving valid feedback. The output presents what the potential users want from the platform and the way how they want to interact with it. Based on the criteria gained, the platform will be designed and developed with continuous validation. The validation ensures that the platform is being developed in a way users want to use it. This approach is called User-Centered design and is an essential basement for every successful software product with frequent user interaction.

1.1 Motivation

This work is a response to the demand I have observed among my friends and people in groups on social media sites. I have also experienced this need for fast valid feedback on my own when I wanted to choose between my logo designs or picture filters. Shortcomings that the current platforms provide did not seem to be a satisfying solution to the problem. To involve also public opinion, I have consequently made couples of interviews to confirm there is actually a real demand for the platform I was considering to create.

This also seemed to me like a great opportunity to demonstrate the influence of fast involving javascript language along with the React.js javascript library [3], Node.js environment [12] and related modern technologies.

Finally I wanted to improve my skills in the aforementioned technologies, so that I would be able to use these skills in practically.

1.2 The aim of this work

The aim of this work is to provide a solution for fast selective user testing. This solution is meant to be an alternative for using current modern applications like Instagram or Facebook which are able to handle this task but are not suited for this purpose. Therefore they will unlikely to provide satisfactory test results 2.2.2.

One of the aim of this work is to be an example that with the use of the latest modern web technologies, it is possible to create a responsive web application that looks like and responds like mobile application wrapped in the web browser. This can be quite useful to realize. When we talk about React.js and React Native [13] domain, then developing a mobile react native application is in general more troublesome than developing a web React.js application, therefore is more expensive. This work shows that if you do not necessarily need a native mobile functionality which latest modern web technologies along with an HTML5 API [8] can not provide, you can just make a web application instead of the native one. Another point of the view at the subject would be if someone considers developing both React.js web application and React Native mobile application of his product. Then, developing a mobile-feels like web application can serve similar way prototypes do. It can be validated with the users and all findings could be incorporated in web application development. When the web application is proven to be good enough, the HTML components need to be edited into native components along with adding a purely mobile-dependent functionality. I believe this approach of the developing both applications at the same time can save some amount of time to the developers and money to the companies.

The reader of this work can get acquainted with the complete development process of the platform from the idea to the application creation and the deployment ready for an immediate use by its new users. Even technically more experienced reader might enhance his awareness within the use of some modern technologies for the development of SPA [18].

1.3 Strucutre of this work

In chapter 2, there is a deeper look at the problem of this topic following by introduction of related applications that are currently being used. The related work is examined in the context of receiving user feedback and from its conclusions, qualitative user research is taking place to find out a better way of solving the topic problem. Based on the researched results, requirements for the platform are set along with the choice of core technologies for the development. Chapter 3 consists of two general parts. The first part is important for platform functionality. There is a draft of platform information architecture with a class and object diagrams which will serve as a template for the implementation part. The second part is more about designing the user interface with the aim of achieving the best possible user experience resulting from the qualitative user testing described in chapter 5. Chapter 5 also considers testing of the platform implementation. The penultimate chapter 4 describes the whole complex implementation process of the application development. Last chapter 6 summarizes achieved results and compares them with the original goals of this thesis. In the very end, the possible future work on the developed application is outlined.

Chapter 2

Analysis

This chapter defines the problem at the very beginning in section 2.1 and breaks down the issues that two currently most used competitive solutions have in following section 2.2. Then the quantitative user research is made in section 2.3 to finds out a better solution that eliminates the mentioned problems. Based on the user research, the user scenarios are being described in detail in section 2.4. From the described user scenarios, the requirement for the platform are being constructed in section 2.5. Then the advantages and disadvantages of using Single-page over Multi-page application are being discussed in section 2.6 along with a brief introduction to the possibilities of application rendering. The section also contains core technologies which are analyzed and selected correspondingly to the defined requirements and rendering method chosen.

2.1 Problem description

People have so many options and possibilities and sometimes it is hard to decide the best one so they ask other people for their advice. When there are only two options to choose between, the inquirer can take notes about how many people prefer a particular option. Then he simply counts them and the option with the most votes wins. The problem becomes more complex when there are more than two options to decide. With each option added, it is harder for respondents to answer. Moreover, if everyone is just suggesting the best option to choose, two or more options can have a very close number of votes in the end.

The assumption is that people want to know what is a better option in general for all respondents and they are not finding one option which will amaze the largest group of respondents but the remaining groups of the respondents would think it is an extremely poor option. Therefore, choosing the option with the most first-votes just because it won for example by one

	v_1	v_2	v_3	v_4
$Option_1$	10	0	0	14
$Option_2$	9	15	0	0
$Option_3$	2	4	10	8
$Option_4$	3	5	14	2

Table 2.1: Example instance of test results with 24 respondents

vote does not seem to be the best approach as it will be explained right away.

More sufficient approach is to let the respondents sort all options from the best to worst. In this case such a scoring system can be introduced that gives the points to the options with the respect to the number of votes for particular ranks. The total score S of option x with the number of the total tested options n would be computed as:

$$S(x) = \sum_{i=1}^{n} v_i * q_i \tag{2.1}$$

where v_i is the number of votes for rank i and $q_i = n - i$ is the multiplying quotient for rank i. Such formula was constructed so it can represent the distance between ranks uniformly thus points between individual ranks are also being split uniformly. Let us just remind the assumption, that the users want to know, which option is the best in general for all users and not which option amaze the major group of the users while everyone else considers it the worst option among all. Therefore no rank has nonuniform point advantage over others.

In the table 2.1, there is an example of the test results where v_i is the number of votes for rank i. This concrete instance was selected to prove, that even option with most first-place votes can be in general third with a nearly double loss to the winner position, as it can be seen on the corresponding graphs in figure 2.1.

However, the decision process for respondents becomes even harder at the moment, when they have to sort all options instead of just choosing the best one. It would be nice to have a scoring system based on sorting while retaining the simplicity of the easy choice for the respondents.

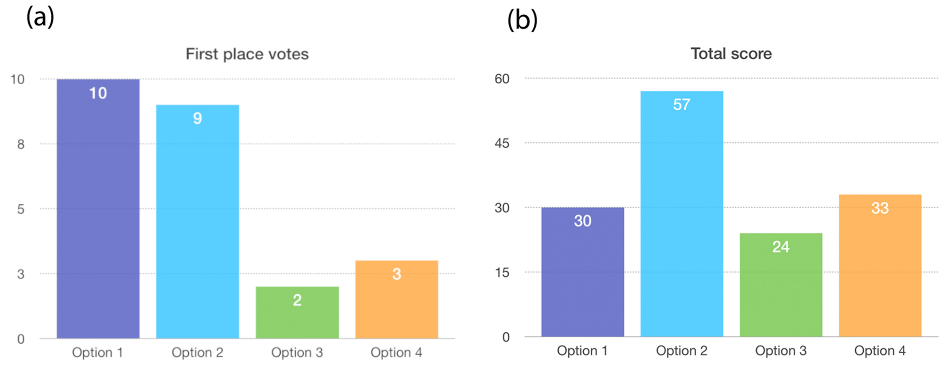


Figure 2.1: Instance from table 2.1 (a) first place votes (b) total score

This is where the standard A/B test method can be involved. A/B testing is widely-used for gaining simple user feedback based on selecting the better option from two. Some kind of modification has to be made to be able to use this method for more than two options. What seems to be interesting at first glance, is considering the composition of several A/B tests. Such approach is described and designed in section 6.

2.2 Related work

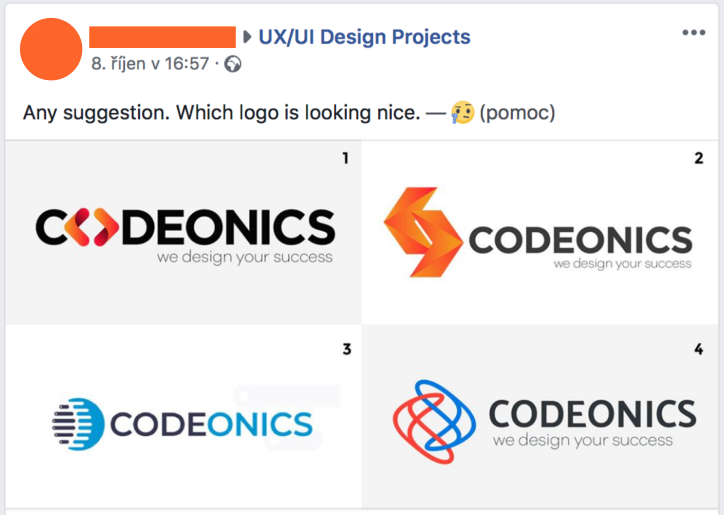
The software product that does not really bring anything new than competitive products is pretty much useless and it will inevitably fail like the Google+ social network did [14]. The design with the implementation can be stunning, but when nobody will use the software, all the effort spent on it will be wasted. Therefore the analysis of similar solutions is necessary.

The aim of this work is to provide fast, selective user feedback for everyone with no need for technical knowledge or previous experience with the user research in any form. Applications that do not meet these criteria will be not considered. Currently, most used ways that are being used for the solution of the researched problem and meets defined criteria, are Facebook posts and Instagram stories. Brief introduction of both will be provided together with the found issues.

2.2.1 Facebook posts

Facebook provides a way to share the question in text form along with only one image, so all images have to be merged in one and labelled with its number. Post can be shared with friends, with a subset of your friends or in such a specific facebook group which can be public or private. Respondents then comment under the post, which variation is the best in their eyes. Then Inquirer manually counts votes in comments to find out the winner.

- need for graphic editing of images
- inquirer have to manually count votes
- question has to be deleted to close the study
- limited subsection selection
- votes only for first places problem 2.1



[redacted] ▸ UX/UI Design Projects

8. říjen v 16:57 ·

Any suggestion. Which logo is looking nice. — 🤔 (pomoc)

Figure 2.2: Real facebook post into a group of UX/UI Design Projects

2.2.2 Instagram stories

Instagram allows sharing a disappearing image post called Instagram story. It is visible for exactly 24 hours to all Instagram users (or to their selected subset) who follows (subscribes) profile of the inquirer. Text can be added to each story as well as the Yes/No survey voting. The range-slider can be also added to the Instagram story, but the inquirer can only see the median point graphically in the slider without any numerical value provided.

For two images it works perfectly fine but when we talk about 3 or more images it will become troublesome. Users can share for example 4 image stories (one image in each story) and put a survey in each story determining if the actual image option is good or not (instead of asking which option is the best compared to the remaining options). Inquirer then needs to compare Yes/No percent ratio between each image story to determine the winner. Similarly for the range-slider where the inquirer also has to graphically compare the sliders from all stories between each other and decide the winner.

It is important to say that the number of respondents for each story is always a bit different. It is caused by voluntary voting in each story and by the delay of the inquirer in sharing stories. The time window between sharing individual options is usually at least one minute. With each image story, it is much more difficult for the inquirer to extract, recompute and correctly evaluate all data from the respondents.

- limited target group selection
- misleading form of questioning when there are more than two options
- inquirer has to recompute the results accordingly to the number of respondents voted for a particular option (story)
- inquirer has to compare stories results to each other to determine the winner
- range slider does not provide result in numerical value
- range slider is prone to the motor ability of the finger
- range slider is often placed on different position in each story, thus motoric possibilities of the respondent are also different

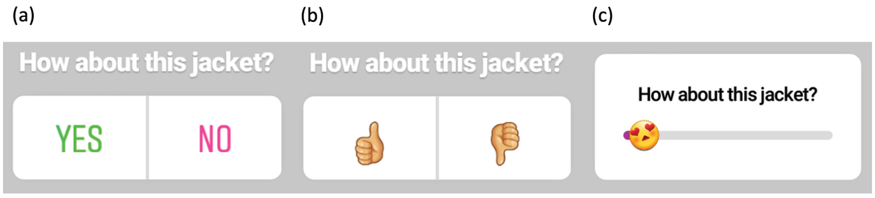
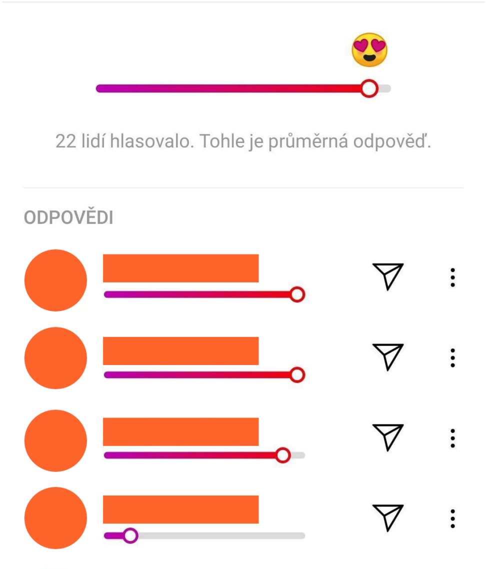


Figure 2.3: (a) standard survey (b) emoticons survey (c) range slider



22 lidí hlasovalo. Tohle je průměrná odpověď.

ODPOVĚDI

Figure 2.4: Instagram range slider results

2.3 User research

In the previous section, considerable issues were found. Before the decision for making a better solution, verification of the need for this application is obliged to be done with the real users. The research can also provide a rough idea about what main functionality would the potential users use in the application. In order to save valuable time, it is possible to find out both at the same time in one research. It is important to admit that what is important for the developers and stakeholders does not need to be important for the users and vice versa. Not only the functionality that users do not want to use will not be implemented at all, but this will also help to determine the priorities in application development. The focus will be on the functionality of the application which is most wanted among the users.

2.3.1 Target group

When developing applications, it is almost impossible to accommodate everyone. The user group for the application needs to be selected. While doing the research with the users, it is important that respondents are fitting to the user group defined for the application. Furthermore, the selected group of users for research needs to equally represent the whole target group for the application. In other words, it has to be a balanced mixture of different groups of users with no major representation of a specific group of users.

Specified age was set to 20-35. It is a group of frequent users of the Instagram filtered from teenagers and kids. It is believed that the filtered group has uncomprehensive views that can change in a few years due to adolescence.

Due to my limited possibilities, for the purpose of this thesis, I was not able to find out such a perfectly balanced user group. In practice, each participant would receive an adequate reward for his participating, so proper user group mentioned above would be found much easier.

Participants, I was able to find, can be divided into two user groups:

- friends and family (20 - 27 years)
- company colleagues (26 - 35 years)

It is assumed that my friends and my colleagues are technically much more experienced that the average group of people in terms of whole population but I was able to get at least an acceptable subset of ideal target group.

2.3.2 Research method

Since there is a need for knowing major opinion among the users, a quantitative study has been selected for this mater. The quantitative study is done via Google questionnaire form [2]. I was able to collect the questionnaire form from 61 respondents (24 respondents from friends and family group and 37 respondents from company colleagues). Since the characteristics of the two groups differ, it would be unwise to merge them into one. For the purpose of observing both groups separately, a separate questionnaire form was made to persist the difference between the groups in the questionnaire results.

2.3.3 Focused areas

Questionnaire structure was made based on the previously mentioned need of knowing possible application usefulness. The quantitative study needs to find out, if such situations as asking about the feedback happen among the people and if they happen not just rarely. Then the user preferences about the application functionality have to be found.

One extra question asking about the usage of Instagram will take place at the end of the questionnaire. When a large representation of respondents are active users of Instagram, the design and user experience of the application will be inspired by the design and user experience of Instagram. Based on these criteria, the questionnaire structure was made.

The questionnaire structure consists of:

- 2 questions determining the frequency of being asked and asking others
- 7 questions determining stance of the users to the application and its functionality
- 1 question determining usage of Instagram application

Subsequently, the concrete questions asking about the preferences of the users, need to be made aside with the question on the probability of use of the application. In the process of creating questions, several assumptions were considered.

The assumptions influencing application functionality:

- people can feel discomfort when asking others for help
- people can be annoyed when someone is asking them for help
- someone can feel he or she might not be the right person to ask
- people might want to display their question to the experts
- people might want to select the target group
- people might want to help others

Instead of the questions, concrete statements for the questionnaire has been made. Respondents then select on the Likert scale [4] how likely they agree with each statement. The statements are:

S1 When I ask for feedback, sometimes I do not feel that comfortable.

S2 When someone asks me for the feedback, it annoys or delays me sometimes.

S3 Sometimes, I feel like I am not the right person to ask this concrete question.

S4 I would use the application for giving or receiving quick user feedback if I have a chance.

S5 I would like to display my question to the experts in the selected field.

S6 I would like to select the age and gender of my target group.

S7 I do not mind helping others to answer their questions if it feedback would be given really quick and in a simple way (by a few clicks).

	Group 1	Group 2
Asks few times a week	12.5%	10.8%
Asks few times a month	33.3%	16.2%
Gives feedback few times a week	41.7%	29.7%
Gives feedback few times a month	33.3%	21.6%

Table 2.2: Frequency of giving and receiving feedback

2.3.4 Research results

In the table 2.2 the results of both questions that are asking about the occurrence frequency of the phenomenon can be found. It can be seen, that for example 41.7% of respondents from the first group (friends and family) experiencing the phenomenon of giving simple feedback a few times a week and 33.3% of respondents a few times a month. An interesting observation is that both groups give feedback much more often than receive it. It appears that both groups are more experts in general than the average group in the population of the Czech Republic. This is caused by not ideally selected target group as was mentioned in 2.3.1.

The other information that was gained was the usage of Instagram. In the first group (friends and family) there are 70.8% of active Instagram users. In the second group, there are 40.5% of active Instagram users along with 16.2% people with an experience of using Instagram, but not using it actively anymore.

The last part of the questionnaire with the S1-S7 statements and respondents identification with each of the statement with the use of Likert scale is shown trough diverging stacked bar charts [26] on figure 2.5 for the Group 1 and figure 2.6 for the the Group 2.

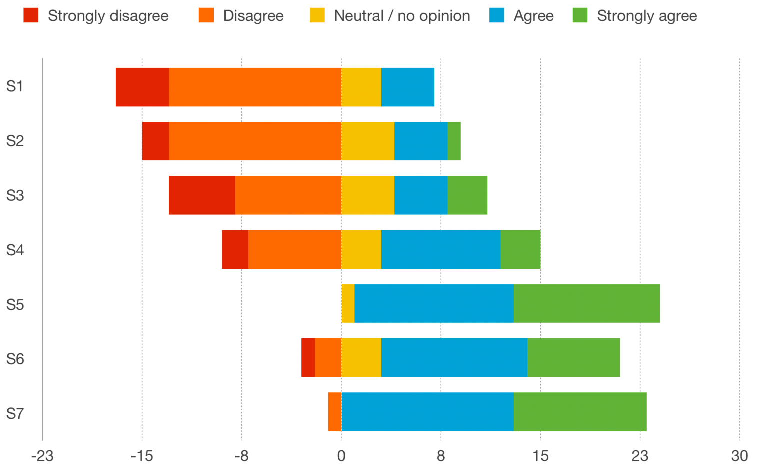


Figure 2.5: Group 1 identifying with statements S1-S7 defined in 2.3.3

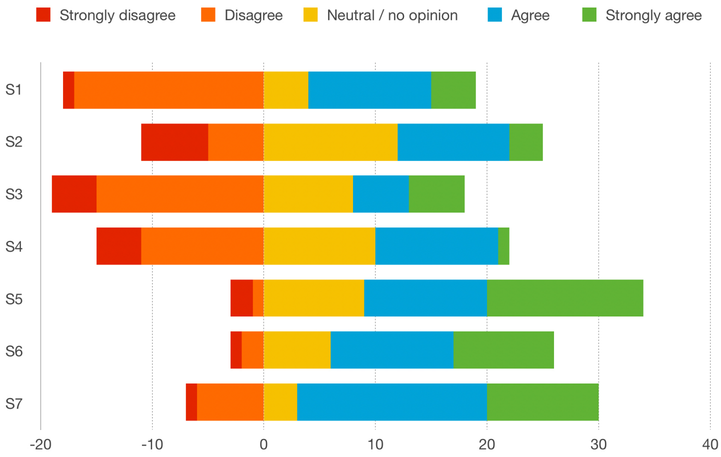


Figure 2.6: Group 2 identifying with statements S1-S7 defined in 2.3.3

2.3.5 Research conclusions

The first result of the research is based on questions considering the occurrence frequency of the feedback phenomenon. The percentages in the table 2.2 shows that the occurrence frequency is not just occasional. Moreover, the memory factor should be considered too, as respondents might forget being asked and so there is a certain probability they were subjected to the question more times than they reported. Therefore, there is a need for giving and providing user feedback.

Second important outcome relates to the usage of the application, namely to the statement S4 mentioned in subsection 2.3.3. More than 50% of the users from each group expressed their preference in using the application as it can be seen from the diverging stacked bar charts in figure 2.5 and figure 2.6.

These figures also consist of the six remaining expressions of the affection to the previously defined statements S1-S7. From the charts, it is observable that respondents do not really identify themselves with the statements S1-S3. It means they are not feeling discomfort while being asked nor while answering questions, so there is no need for caring about the mater of user discomfort in the design process of the application. The fact there is an unusually high occurrence of participants in the group expressing their comfortance can mean that the target group of respondents was not correctly selected because the specific group with a certain characteristic predominates in the group of the participants. That happened very likely to the target group selected in subsection 2.3.1. It is understandable that expertise users (proven to be questioned more often as shown in table 2.2 shows) will be more comfortable than the average users.

On the other hand, participants strongly identify themselfs with the statements S5-S7, which means they would like to select the age and gender of a question target group. They would also like to show the question to the expert in the question field. Finally, they expressed interest in helping other users by providing them their feedback if the whole process of providing feedback will be really simple and fast.

2.4 User scenarios

To be able to construct user requirements correctly, the formal description of user needs is required. The user needs are presented in scenarios typical for the selected target group of the application. The major interest is in the gaining and providing user feedback which means the scenarios 3 and 6 are key scenarios and comes with their own storyboard for each.

User scenario 1: Registration and login

A student has been informed about this new application by his roommate. He was curious and he decided to visit the application page. He was looking at the login page and found a button for creating an account. He easily found the button, clicked on it and was redirected to the registration form. He filled his full name with email and date of birth. Then he typed his password which consisted of 4 characters. As he clicked out of the password field, wanting to select the gender, the red error message under the password field appears. He is reading the message warning about the fact that the password length has to be at least 6 characters long. He recreated his password with the respect to this requirement and selected his gender. Then he submited the form and waited for the response. A popup informing him about the success of creating a new user was displayed to him. He proceeded to the login screen where he was filling the newly created email and password credentials. However, he misspelled his email and he proceeds to the password field which he also accidentally misspelled. He was trying to submit the form but the application shows him warning about the non-existing user with such an email address. He corrected the misspelled email and is trying to submit the form again. Now the application showed him error message refering to the invalid password. He is retyping the password into the correct form and trying to login again which is finally a successful login recognized by the application.

User scenario 2: Filling up the expertise

A new user of the application looked to his questions dashboard. Besides seeing the questions which are not expertise-dependent he saw an advice about filling up his expertise in the profile section. He went to the profile section and saw the field for selecting the expertise along with the information of the minimum and the maximum number of expertise. In order to add his fishing skill as expertise, he opened the select box with the options. There are so many options and he could not find his desired expertise, but he noticed the blinking cursor in the select box field and started typing the first letter of his expertise. Immediately as he types in the first letter, the options of the select box are filtered to the ones starting with the first letter provided to the field. In the filtered options he can now see his desired expertise in fishing. He clicked on the expertise he found in options and the expertise is added visually to his select box field. He added five more expertises but as soon as the last one is added visually, the error message under the select box is informing him that the limitation for expertise is 5. He removed one expertise in which he believed is the worst among all selected and he submitted the form. He is being informed that all changes have been saved by the application.

User scenario 3: The feedback needed

A graphic designer in a well-known company has to deliver a new logo for a very important client of the company. He is sitting in his office being aware that the strict deadline is within a day and a half and he has still not decided what version of the logo would be the best choice. He realized that he can use the application for this matter. He logged into the application and began to create a new question. He is filling up the question field with his question and subsequently selecting a graphic design category. The need for expert advice leads selecting the box "expertise dependent only". He chose the target group from 25 years to 35 years hoping for a young yet experienced judgment in graphic design. Then he is setting the question duration to 24 hours so he can manage to send the best-selected design to the client in time. While he is uploading the image he accidentally chose differently than he wanted. So he changed the selection of his third image to the proper one. He is looking at the hint telling him that the question costs 10 credits so he checked his credit amounts in the application. Luckily he had enough credits so he is posting the newly created question and is waiting for the confirmation. The confirmation of the success is being received. With a lighter conscience, he is going for a cup of coffee expecting to receive test results over time.

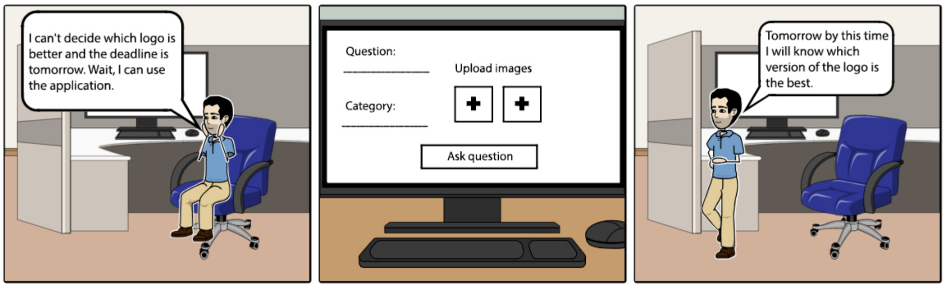

I can't decide which logo is better and the deadline is tomorrow. Wait, I can use the application.
Question:
Upload images
Category:
Ask question
Tomorrow by this time I will know which version of the logo is the best.

Figure 2.7: User scenario 3: Demand for feedback in graphic design

User scenario 4: End of the question

A graduate student just received the message that tomorrow business meeting has been cancelled. He already asked three questions in the application related to the meeting that has been canceled. He does not want the users to provide needless votes, so he clicked on the button to end questions and choose those three questions one by one. Therefore one by one question is being terminated and is disappearing from the dashboard of the other users.

User scenario 5: Change of the question

A stressed women marketer is looking at the results of her newly created question. In the results summary of the question, she is realizing that she checked to target men instead of women. The question only makes sense for women to ask so there is no reason to let the question continue nor allow users to provide their feedback. She considered deleting the wrong question and was thinking about creating a whole new one, but she realized she can just change the question. That means all parameters of the previous wrong question can be transmitted into the form for creating a new question. She selected the question she wanted to change and checked the women checkbox instead of the men. Then she submited the edited question.

User scenario 6: The feedback provided

A father of two kids has some free time while waiting for his friend to arrive. He is thinking about using his time in a more useful way than just reading the news as he often does. Therefore, he opens the application to see, if there are some new questions which he is able to give feedback to. On the dashboard, he sees there are 3 new questions. Two from the graphics category and one from the fashion category. He has just a bit of free time, so he does not scroll down in the application to find out if there are some new questions not dependent on the users' expertise or questions listed as Uncategorized. He opens the first question thumbnail from the graphic design and reads the actual question. The question is about the particular graphics studio which he does not know much about. For that reason, he is skipping the actual question preview to the next preview of the question and reads the new question. It is about the 2D design which he is really comfortable to provide feedback with. Therefore, he started the question test. He is switching between the options and he keeps selecting the better one while observing the status progress of the question down below on the question test. When he is selecting the last picture from the A/B comparison, the application informs him about successfully finishing the test. He immediately sees he just received credits for the help. Having enough credits to ask tomorrow about his question he closes the application with a positive feeling of helping someone in just a moment of free time he had.

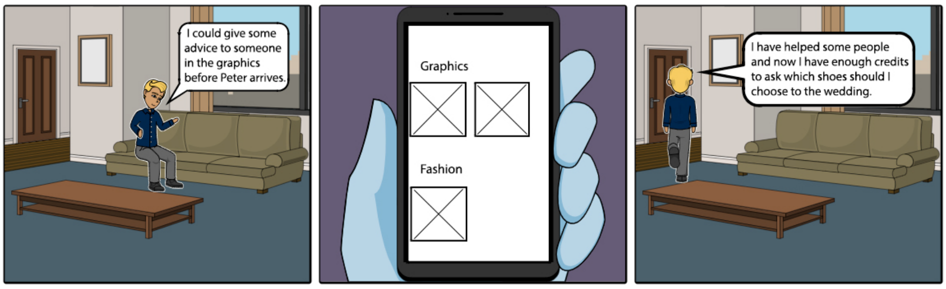

I could give some advice to someone in the graphics before Peter arrives.
Graphics
Fashion
I have helped some people and now I have enough credits to ask which shoes should I choose to the wedding.

Figure 2.8: User scenario 6: The advice in graphic design provided

2.5 Requirement analysis

The user scenarios provide detailed formal description not suitable for the use of developers and related stakeholders who would need to find between the lines the desired application functionality. There is a need for extracting the requirements for the application from the user scenario. Requirements can be assumed and formed as actions, objects or contexts. It is important to realize requirements are not the same as tasks or features [6]. The requirements were defined accordingly to the analysis, observations, and conclusions made in the above chapters.

2.5.1 Data requirements

Data requirements can be described as the objects and adjectives related to those objects [6]. Data flow diagram is used to visually represent the flow of the data in business information system. Data flow diagrams describes what processes are involved in order to transfer data from the input to the storage [28]. On the figure 2.9 the flow of the data from the input to the storage can be seen. The flow from input to the storage is provided by application processes.

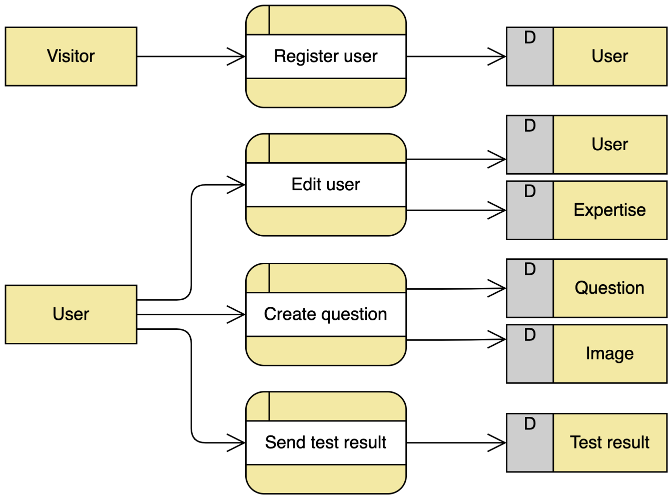


Figure 2.9: Data flow of the application

2.5.2 Use cases

The support for the user scenarios by the application is described by use cases. Use cases are technique based on exhaustive descriptions of functional requirements of the system. It is focusing on the low-level user action and associated system response [7]. All actions in the application are being triggered by actors. Actors are external objects exchanging their information with the application [15]. It is not good to generalize actors, so the range of users has been split to the cases when the application does and does not provide them one major functionality - creating the questions.

The user roles are:

- **Visitor** is any person who visits the site of the application
- **User** is a registered **Visitor**
- **User with credits** is a **User** with a minimal value of credits needed for asking a question

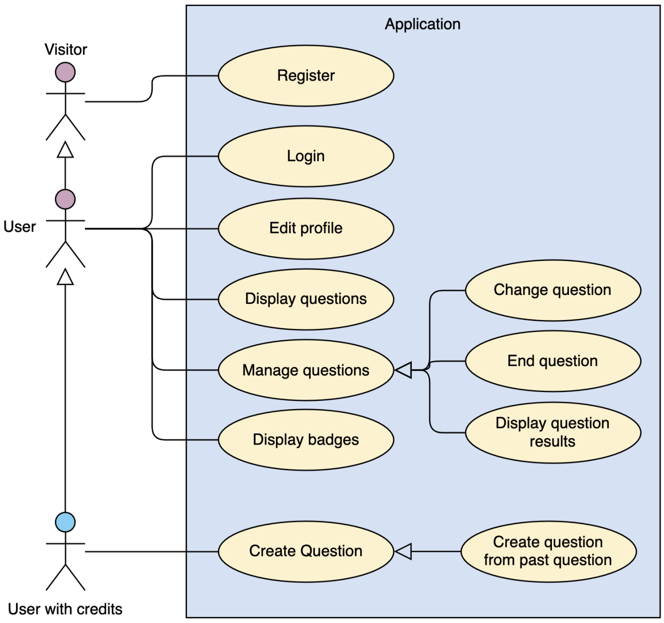


Figure 2.10: Diagram of application use cases

2.5.3 Functional requirements

Functional requirements define a function of the application or its component. Functional needs are typically some actions or operations that needs to be performed on the system objects [6]. Functions can be described as a behavior between outputs and inputs.

Interface requirements:

1. Forms allow autofill
2. Forms allow easy orientation in form fields by tabulator key
3. Input fields are being validated according to their type (email, number, etc.) by HTML 5 validation
4. Input fields are being validated by its type with external javascript serving like a backup client validation for browsers with no or partial support of HTML 5 validation
5. All required input fields are being validated by HTML 5 and external javascript (for the same reason as mentioned above)
6. Registration email needs to be unique in the application
7. Date of birth has to be in the year range 1900 - 2018

8. Question duration should be at least one hour and a maximum of 24 hours
9. When the question is being created with the "Uncategorized" or "other category", user cannot select expertise audience of the question.
10. There have to be four inputs for file upload and provides a thumbnail of each image loaded
11. File inputs are ensure that user loads at least 2 images (no matter to what file inputs)
12. File inputs allow up to 1MB per image as maximum and provide information about which file inputs exceeded the file size limit
13. Select box with existing expertise has to allow only up to 5 expertise with the minimum of one
14. Select box with existing expertise has to be able to search expertise according to provided input letters
15. Select box with existing expertise has to suggest the creation of searched input string if the string does not perfectly match one of the existing expertise
16. When there is an error on some input field, the red error message describing the violation of the input rules is displayed to the user right under the input with the incorrect user input provided.
17. Input field validation errors are being displayed right after the user leaves the input where is it possible, otherwise the remaining errors are displayed after submitting of the form
18. Submitting the forms is time-outed on the client to prevents double click by mistake or brute force attack on the client side

Security requirements:

1. User password is saved in database hashed and salted by a method commonly used in the production environment
2. All API routes of the application are secured except the login and registration
3. Server will not make database record if the data send to the server do not match their data type expected defined on the server
4. Server automatically logs out the user when he closes the browser session
5. Application will not display any error to application console or to the application itself, that would reveal sensitive information about the server or database

Other functional requirements:

1. Visitor of the application can register an account
2. Registered user can log in
3. User can view thumbnails of other users' questions on the main dashboard only if the users' profile settings match the questions' target group
4. User can edit his name, email address, expertise and turn on or off the notifications of a new questions
5. User can create a question with the possibility of selecting the question category, target group gender and age, question duration and images to test
6. User can create a new question from his past questions
7. User can terminate or change his question
8. User can display test results of his question
9. User can display a preview of any question thumbnail where he can see the question
10. User can switch between the previews in the same category by navigation buttons
11. User can start and pass any selected question test
12. User can switch between A/B variant in the test and by selecting one, he can see the test progress bellow
13. User cannot vote in the test for the image A if he has not seen image B yet
14. Question categories are being loaded from all existing user expertise and option "other/uncategorized" is provided below all question categories options in the select box of creating a new question
15. Question categories are being created dynamically when any user creates (and adds) new expertise to his profile
16. Question categories are being removed dynamically when the last user with the expertise removes the expertise from his profile
17. User is being informed about the new questions matching his profile setting by a web notification (if enabled)
18. User can undo his choice in the question test within 3 seconds.

19. Dashboard with the questions automatically reloads every 30 seconds and provides the most recently created questions by adding them into the dashboard in the way the user is not being moved on the page or interrupted. The load process of the questions is presented to the user with a loading status bar on the top corner of the dashboard

Informational requirements:

User is informed about:

1. Maximum and minimum values in question duration, expertise selection, and files to upload
2. Maximum and suggested size of the files to upload
3. Absence of questions from the users on the questions dashboard
4. Absence of his questions to choose from in all "manage questions" operations
5. Absence of test results when question displayed is not yet answered by any user
6. Absence of his expertise while the application suggest their addition
7. Functionality of filling the form for new question with the values of the past question
8. Number of total answered questions
9. Number of total badges collected
10. The amount of virtual money user has

The application informs the user by displaying a popup message when:

1. User is creating a new account with the same email already existing in the application
2. The incorrect email is provided in the login
3. Incorrect password is provided in the login
4. Profile is successfully edited
5. Question is successfully created
6. Test result is successfully sent
7. Question is terminated
8. Question is changed
9. There is an error and the request could not be processed

2.5.4 Non-functional requirements

As it is apparent from the title, non-functional requirements are not concerned with the functions of the system. They define the criteria which the application is expected to conform to. They can include for example response time and reliability and other requirements for sufficient quality [9]. In the process of creation non-functional requirements, the aim mentioned in section 1.2 about creating the mobile-friendly web application was considered. The application also needs to support modern widely-used browsers. Support for the Internet Explorer and Edge was not put in the requirements for know issues in these browsers when they are used for working in a modern javascript application. The third non-functional requirement prevents the user with bad intent overloading the memory of the server with the huge files sent in the post requests directly to the servers' API handling question creation.

The non-functional requirements of the application:

1. All application screens are responsive, so they display correctly in any device type with any screen size
2. All input fields should be easily filled from a mobile device
3. The server needs to save only images in the maximal size of 1MB
4. Application should work correctly in the web browsers:
    - Google Chrome
    - Safari
    - Firefox

2.6 Technologies and methods

Before designing the application, the technical possibilities of the implementation of the application have to be explored in order to see the big picture. In subsection 2.6.1 the two main concepts of the application are being compared and the better approach for the use of developing an application is chosen. In subsection the core technologies for the application development are being briefly described. They have been selected accordingly to the requirements of the application.

2.6.1 SPA vs. MPA

In web application development, there are two main design patterns: Single-page application (SPA) and multi-page application (MPA). Both design patterns have their advantages and disadvantages and it is always important to realize what the requirements of the application are and based on them decide the most suitable design pattern [21].

A multi-page application design pattern is the old approach widely used in most of the applications based on the PHP for example. Every change on the page triggers a request which is sent back to the server. Then, the server process the request, build the new page based on the application logic and send it back to the particular user. AJAX [29] is used to support dynamics of the MPA applications by sending an asynchronous request to the server which returns part of the page content or data back to the browser without additional reload of the page [29]. The major advantage of MPA is proper SEO [30] management. The disadvantage is complexity caused by multiple levels of view and the long response time of the server.

The single-page application works inside the browser and does not demand page refresh when the user makes interaction with the application. On the server, there does not need to be any code related to the application. SPA is easy to debug with Chrome extension and it is much easier to get started with the development. SPA can only send requests to the server to demand the data. After obtaining the data it can use this data and works even offline. Because of the javascript rendering nature, Single-page applications have poor SEO management and results in the browsers' rankings. Sometimes they are not even listed in the search results, because web crawlers [17] cannot extract the correct information about the application. However, this is already solved by requiring rendering of the particular application parts (related to the SEO) on server-side. Popular React framework for server-side rendering is Next.js [22]. The goal of this application is not in the marketing field, so the SEO nor server-side rendering is actually needed. However, one of the goals of developing an application is to provide fast user interactions and that is exactly the case where Single-page application design patterns excel.

2.6.2 Analysis of core technologies

Yarn (development)
Yarn is favorite javascript package manager [32]. Libraries, frameworks or components are easily managed by Yarn. The advantages over npm package manager [24] are that it is a little bit faster, it has better security and provides better collaboration in teams.

Webpack 4 (development)
Webpack [31] is a configuration module bundler with a lot of features. Its main purpose is to bundle JavaScript files for usage in a browser so it is capable of transforming, bundling or packaging resources and assets.

Babel (development)
Babel is a JavaScript compiler for ECMAScript 2015 and newer versions [5]. It makes new JavaScript code backward compatible with older browsers or environments. It also includes a polyfill feature that allows using promises and other new features available in the new versions of the javascript.

javaScript ES6 (development)
ECMAScript 6 [1] is a new version of JavaScript bringing a lot of new features for developers to use like promises, arrow functions and much more making the code more readable and less prone to error.

HTML 5 (development)
HTML stands for Hypertext Markup Language and is made for creating the structure of a website. The newest HTML 5 was released in 2014 and came with many features like Drag and Drop API, session storage and new element tags like "nav" and "section". HTML 5 is cross-platform, so it does not matter whether tablet, smartphone or desktop computer is being used as long as the device web browser has HTML 5 support.

React.js (client-side)
React.js is a JavaScript library for creating website components developed by Facebook. It is focused on user interface and interactivity. The angular.js framework is taking care of the backend too and demanding to do things in one defined way. Unlike Angular, React allows more freedom in the development process, so the other libraries frameworks and packages can be used freely to solve part of the application functionality.

Styled Components (client-side)
When it comes to the component-based applications, Styled Components [27] is the alternative version of the popular Sass (Syntactically Awesome Style Sheets) preprocessor for creating CSS styles. The main idea is that every component has its style separated from every other component. Instead of React.js inline styling, it allows you to do external styling in separate components, which is way more easy to read. Styled components do not use class names because it goes against the component logic. Dynamic styling allowing to use react component properties can be very helpful when designing the component view based on the component internal state.

Redux (client-side)
Redux [25] was made in order to easily manage the state of the application. As application increases its size and complexity, it is much harder to pass properties between components. Redux represents application state container which is available from every component connected to the redux reducer so even unrelated components far away from each other in the components tree can be connected directly and their properties do not need to send through the component tree hierarchy.

Formik (client-side)
Formik is a javascript library that takes care of the logic behind forms and things closely related to it [11]. For example, handling the form values, errors, touched fields, validation and submission handling.

Yup (client-side)
Formik has the open support for Yup [33] which cares only about form validation. Yup provides creating of own validation schema which is then used to validate all user inputs according to the validation rules defined in the Yup schema.

Node.js (server-side)
Node.js is an open source server environment based on the Javascript. It can run on most platforms like Windows, Mac, and Unix. Thanks to its asynchronous principle, Node.js is used to process server requests fast and so it is widely used for real-time hight applications with the high data and request intensity.

Express.js (server-side)
It is a Node.js framework that makes working with the node.js easier for example in creating server API and handling the requests and sending the responses [10].

MongoDB (server-side)
MongoDB [19] is a flexible and scalable document-oriented (NoSQL) database with JSON-like format for high volume data. mLab server provides free usage of MongoDB so it can be used for the purpose of storing application data which are meant to preserve.

Mongoose (server-side)
Mongoose [20] is an object data modeling library which manages the relationship between data. It helps to easily interact with MongoDB database and Node.js. It includes data types, validation, business logic hooks, simple queries, and provides faster development of the application data-based models.

2.7 Chapter summary

In this chapter, the problem has been described along with the suggested approach to the problem in section 2.1. Then the related work has been analyzed and related issues were described in section 2.2. After that, the quantitative user research took place in section 2.3, where the whole process of the research has been described and presented along with the research conclusions. Based on the user research, detailed user scenarios have been made in section along with the storyboards for the two key scenarios 2.4. The work continues by extracting the application requirements from the scenarios and describing them in section 2.5. Last but not least, the advantages and disadvantages of using Single-page over Multi-page application have been discussed in section 2.6 and the SPA design pattern has been selected. Finally, core technologies have been analyzed and selected for the purpose of the development of the desired application.

Chapter 3

Design

This chapter describes the design process. The application has been developing through the iterative UCD, which has three phases in each iteration - design, evaluate, implement [16]. In section 6, the scoring system for ranking the images is being presented. Each image gets its rank based on the score obtained from the user comparisons. In the subsequent section 3.2,the information architecture is being discussed.The discussion about the information architecture has been split into 3 individual parts. System architecture, information architecture, and website architecture. The individual iterations of the design process are being described in the final sections of this chapter. It starts with sketches in section 3.3, progress to the low-fidelity prototype in section 3.4 and finish off with the high fidelity prototype in section 3.5. All of the described stages of the design process are tightly coupled with chapter 5 since every design iteration must be evaluated with the users.

3.1 Options ranking system

With the respect of the problem solved in this thesis and described in detail in section 2.1, appropriate ranking system has been designed. The ranking system has an aim to provide 3 or 4 images to the user while receiving the images back from the user sorted by their rank accordingly to preferences of the user. The whole process of obtaining the image ranks from the user is being ensured by the series of A/B testing, which figures in the standard knockout duel system known from sports from example.

The situation with the two images is trivial. Therefore, the algorithm has to be demonstrated on the test case with 3 and 4 images. On figure 3.1, there is an example of algorithm functionality. The emoticons represent, how the user likes the image so they predetermine the winner.

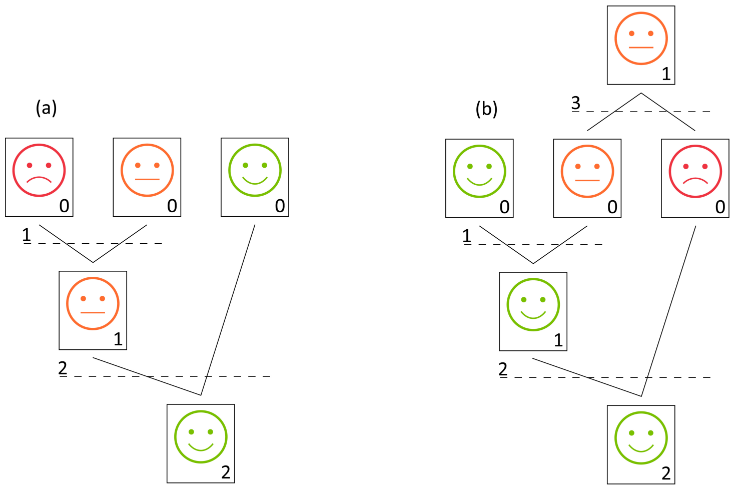


Figure 3.1: test with 3 images (a) best case (b) worst case

The algorithm works in general as follows::

1. Initialize the score of each image to 0
2. Let the user compares the image A and B
3. The winning image gets the score of the losing image + 1
4. Move on the next A/B pair and continue from point 2.

Now, when the algorithm is done it has to be ensured if there is a need for another comparison. It can be simply detected by having the same score of two images. To be concrete in the case of 3 images it is score 0 and in the case of 4 images, it is score 1. In the case of four images, additional comparison can be skipped if the images with the same points have been compared in the first level of the image tree as shown in figure 3.3. It is important to mention that winner of the additional comparison in case of 4 images does not get one extra point for moving itself somewhere in the images tree, thus the image just takes the score just from the picture he lost against him.

When the algorithm ends, it has images with their score. The rank r of the tested image x is then computed as:

$$r(x) = n - s(x) \tag{3.1}$$

where n is the number of tested images and $s(x)$ is the score if the tested image x.

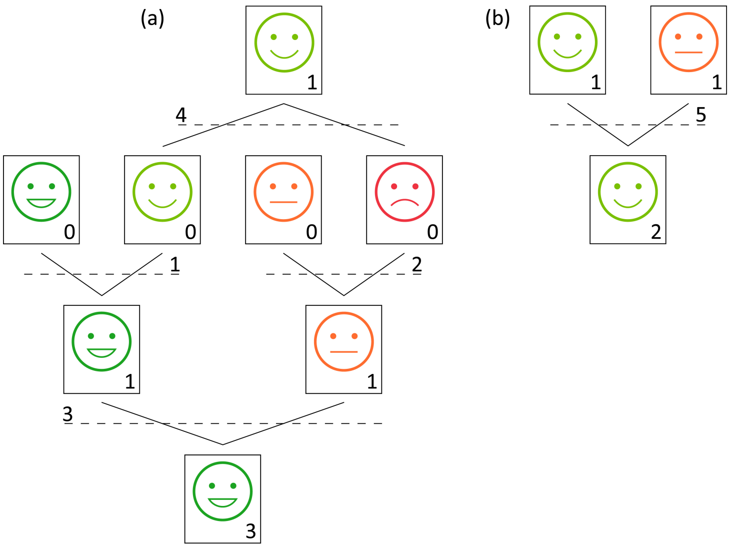


Figure 3.2: test with 4 images, an additional comparison needed

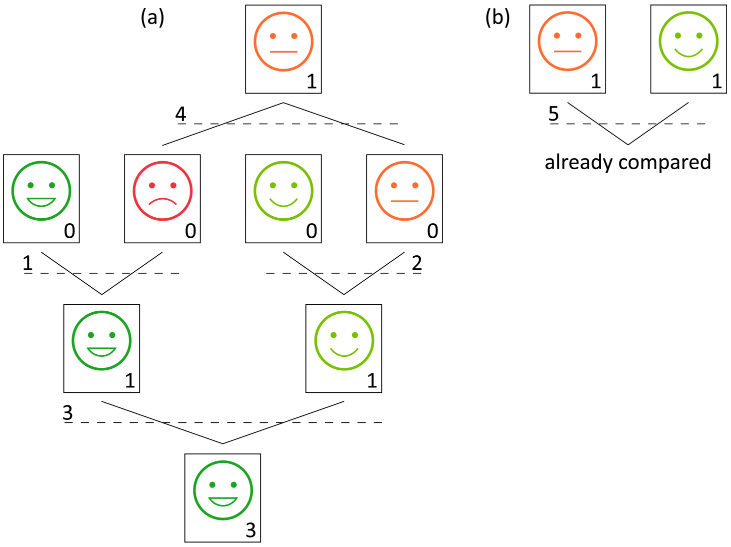


Figure 3.3: test with 4 images, no need for additional comparison

This ranking system has been designed for the purpose of this application and may require some modification when it is desired to be used in a different context. But for the purpose of this work is fully sufficient because there is always a limitation of comparing 4 images in maximum. That has two reasons. If someone cannot decide between more than four options he should consider rethinking his criteria and motivation. Secondly, the application does not limit the user in this matter. He can do a series of tests and then compare the winners in another test. In this way, even the user absolutely unaware of which version of the 20 options is the best, can be satisfied.

3.2 Information architecture

Information architecture forms a foundation for user experience design. It is a structural design of shared information environments. To be able to correctly represent Information architecture is an important skill within the user experience design. UML diagrams are also important in order to remain company know-how in the one place neatly. It is essential especially for the new people on board and even more when architects of the system suddenly leave the company.

3.2.1 System Architecture

On figure 3.4 the system architecture can be seen. Application is hosted on the Heroku server that provides a node.js container. The database is hosted on the mLab server so it can be used from local development server independently of the server. The application logic on the client is taking care of fetching respectively posting data form respectively to the servers' Rest API which handles all requests from the client, process them and sent them to the mongoose driver model methods. The model methods then querying the mongo database with the specific requirements deriving from the nature of the action.

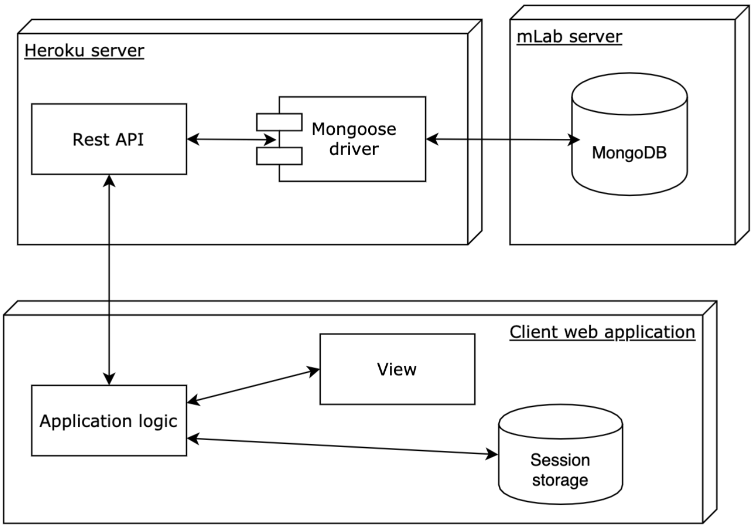


Figure 3.4: System Architecture

3.2.2 Aplication Architecture

On figure 3.5 Redux data flow with the interaction of the React components is being presented. View components like inputs and other small visual elements record an interaction triggered by the user (like click to the button). They pass interactions into the higher-order smart components with the internal component logic. They process the interactions and eventually triggers the call for actions so action can be fired in action creators. The actions are being dispatched to the store from where they are sent to the reducers along with the previous state. Reducers process the action in the context of state provided and creates a new application state which they send to the store. In other words application state immutable objects and is being replaced with the whole application state produced by reducers each time the application state needs to be changed. This core principle is the basement of whole application logic.

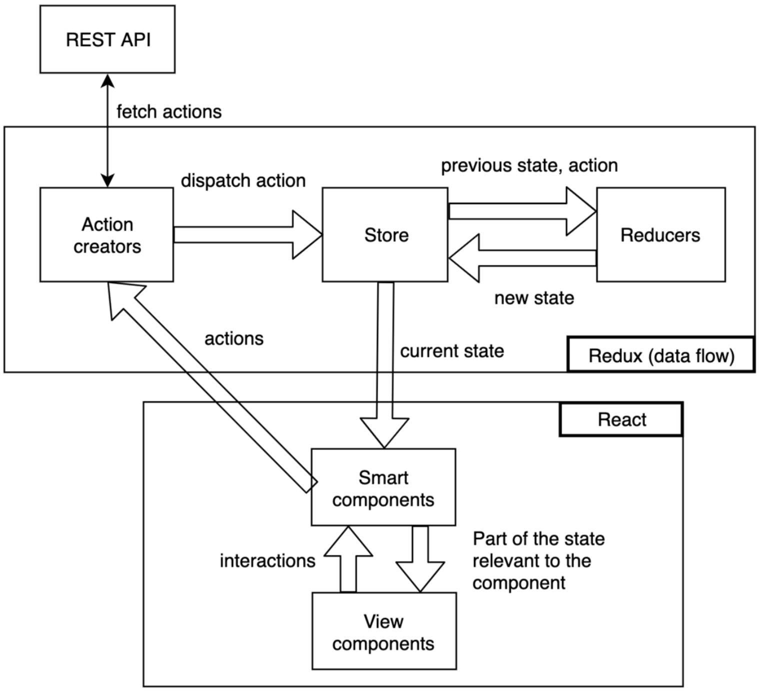


Figure 3.5: Application Architecture

3.2.3 Website Architecture

Website page structure is captured on the figure 3.6. Application has a login as a homepage. Then user can proceed in the root pages off the application, namely **Questions dashboard**, **Manage questions** and **Profile**. These 3 root pages do not have the back arrow on the mobile phones because coming back to the login page is not supported by this way (user has to log out to see the log in page again). In the case of the remaining pages, the back button serves for the purpose to be navigated one step back (to the pages' direct parent page).

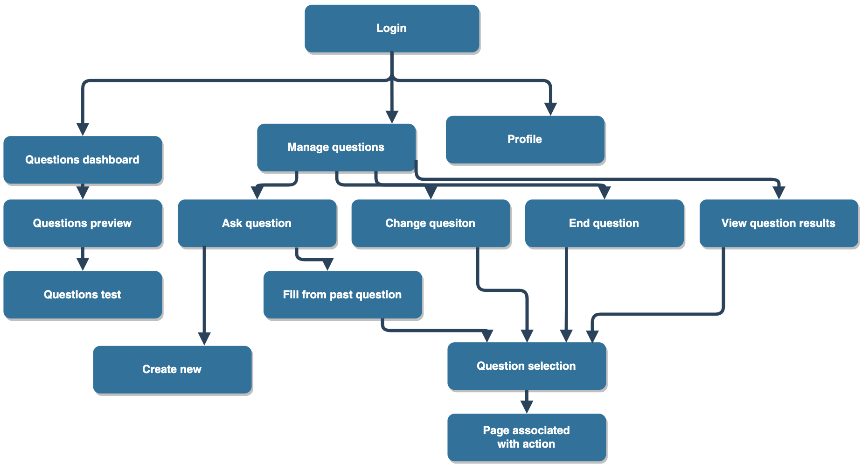


Figure 3.6: Website Architecture

3.3 Sketches

In the very beginning of the design process, the creation of sketches has been taking place. It was necessary to construct a basic graphic concept of the interface according to previously defined requirements and Website (page) architecture. The main focus has been set on sketching the dashboard with the questions. Details about the testing of the sketches are described in related section 5.1. Visual outputs of the sketches are often not presentable to the outer world, so that is the case.

3.4 Low-fidelity prototype

Sketches have been useful for bringing a brief idea about the layout of the main components like a dashboard with the questions. It was needed to take the design further with the more detailed concepts of the design, functionality, and layout of all application and its elements. The low-fidelity prototype have been made with respect to application requirements. The main focus in low-fidelity prototype testing was to test answering the question, creating the question, viewing the results of the question and editing a profile. Detailed test scenarios are being described in section 5.2. After qualitative testing with the users, the results and recommendation have been processed into the new version of the low-fidelity prototype.

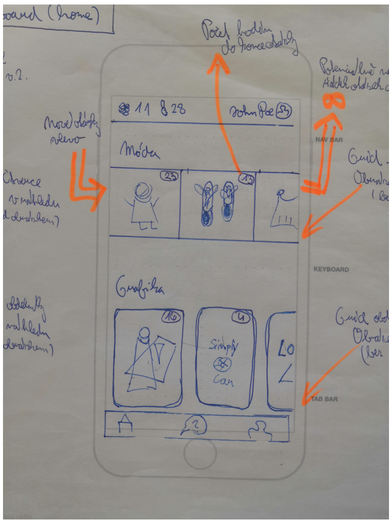


Figure 3.7: Low-fidelity prototype of questions dashboard

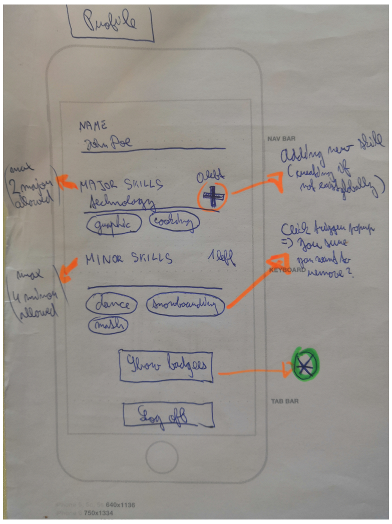


Figure 3.8: Low-fidelity prototype of profile

The nature of goal-aimed test scenarios where users had to imagine their interactions with the system led to the discomfort of some participants. They have been claiming about the excessive need to imagine the reaction system. Therefore, clickable application wireframes have been created from the low-fidelity prototype using the Balsamiq application and the user testing have been done again using the same test scenarios mentioned in section: 5.2

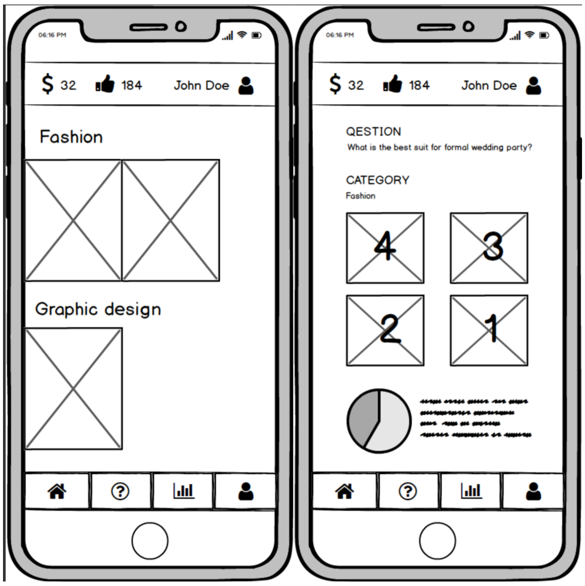


Figure 3.9: A low-fidelity prototype of questions dashboard and question results

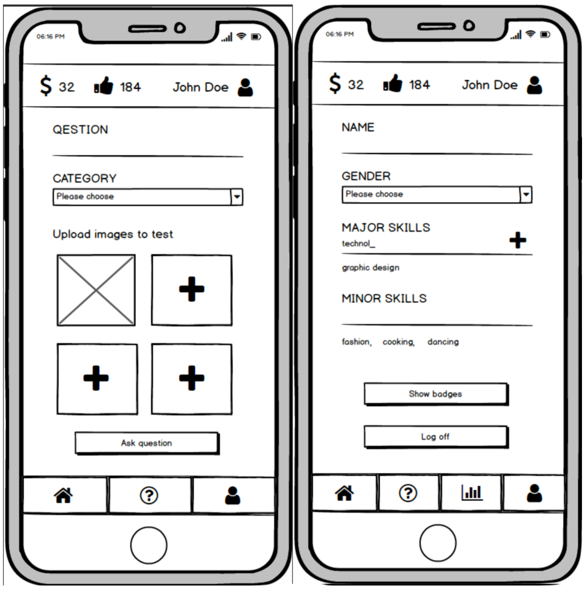


Figure 3.10: A low-fidelity prototype of create quesiton and profile

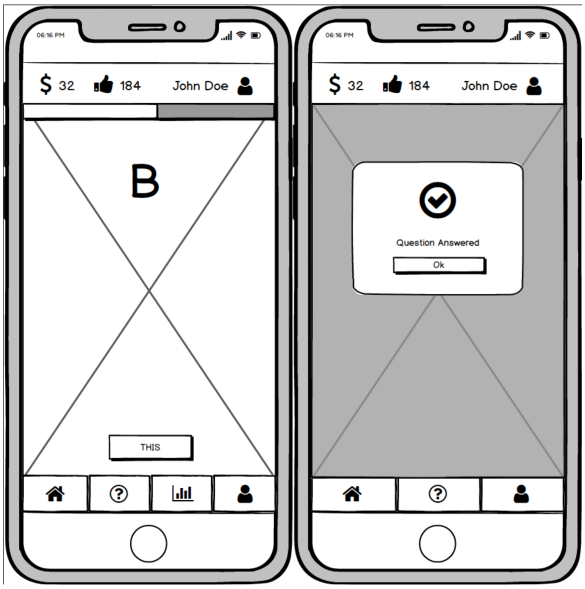


Figure 3.11: A low-fidelity prototype of question test

3.5 High-fidelity prototype

When all the recommendations from both low-fidelity testing described in section 5.2 have been processed to the new design, it was about the time to make a high-fidelity prototype. Invision studio has been chosen over other applications because it provides easy to learn environment with the great support for modeling animations and user interactions. The prototype has been validated with the users as section 5.3 describes. The final results of the high-fidelity prototype can be seen in the figures 3.13, 3.14 and 3.12 below.

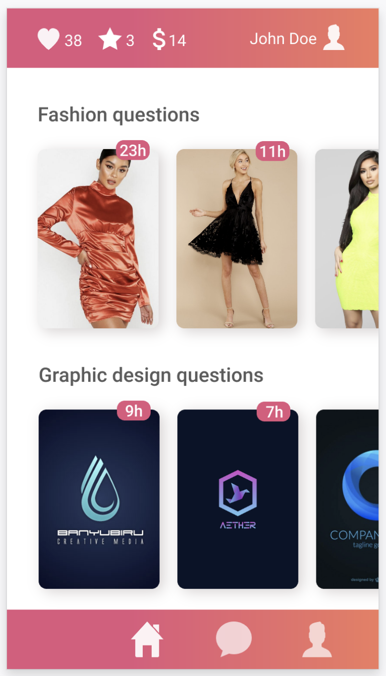
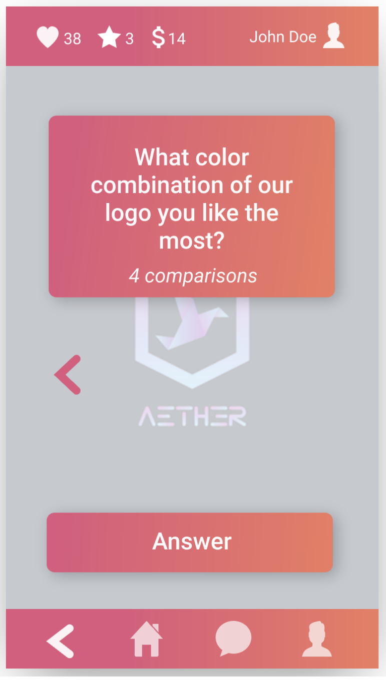


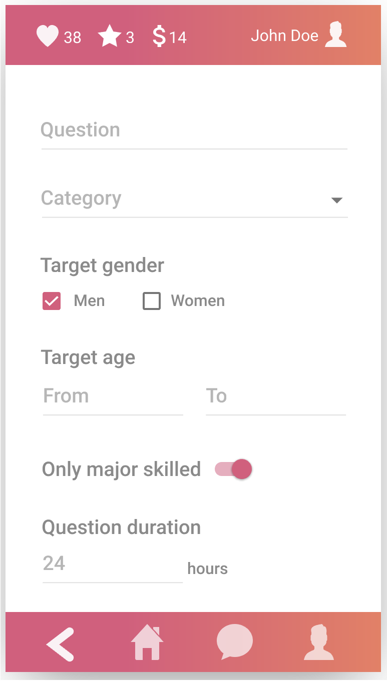
Figure 3.12: A high-fidelity prototype of questions dashboard



What color combination of our logo you like the most?

4 comparisons

Figure 3.13: A high-fidelity prototype of quesiton test



Question

Category

Target gender

☑ Men ☐ Women

Target age

From To

Only major skilled

Question duration

24 hours

Figure 3.14: A lhigh-fidelity prototype of profile

3.6 Chapter summary

In this chapter, the design of the application has been described, together with the iterations went through. In the first section, the scoring system determining ranks of individual images from the question tests with A/B comparisons have been introduced. In the following sections, the information architecture is being presented thus providing a good basis for the implementation part. There was a focus of the design process in the final sections. Sketches have been described first followed by the low-fi prototype which has been made based on the sketches. Similarly for the following hight-fi prototype which has been made with the respect of low-fidelity prototype and its findings and recommendations gained in the testing phase of the low-fi prototype mentioned in the section 5.2. After user evaluation of the high-fidelity prototype described in section 5.3, the changes have been reflected in the high-fidelity prototype, which has been evaluated with the users again. Since no more design iteration have been planned, the findings obtained from the last evaluation with the users have been written down and kept in mind in the phase of implementation of the application described in chapter 4 placed below this chapter.

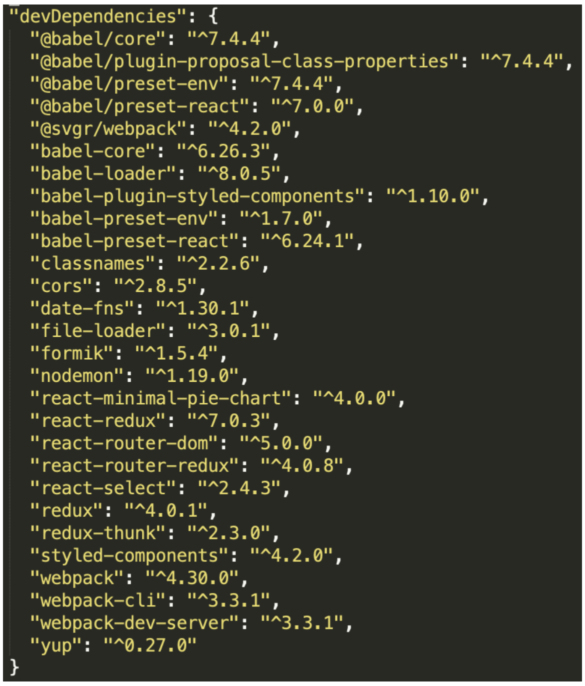
Chapter 4

Implementation

Implementation has been done with the respect of all defined functional requirements in section 2.5 and non-functional requirements defined in section 2.5.4 and according to all diagrams designing the implementation in this thesis.

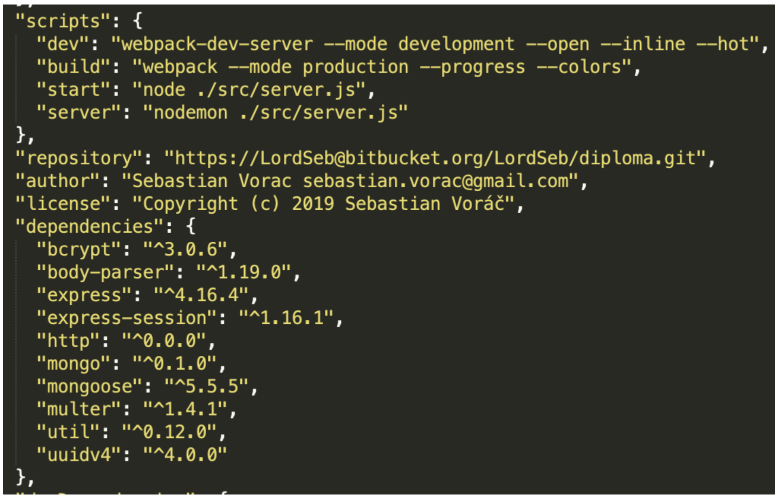
4.1 Development environment

First of all, the developer environment has been set. With the usage of yarn, javascript package managing has been really easy. I could just add and remove development dependencies or server dependencies at all packages used in the application are being reflected in the one file called **package.json**. Some packages have been added immediately because I knew I will be using them and some have been added when needed. In the package.json the task runners can be also set. Task runners I did set was one for the development of the client application, one for the starting a virtual server and one for the build of the application - making javascript bundle.js file and other assets ready to deploy on a server. The Webpack setup is needed to be made in order to make task runners, a compilation of assets and javascript, hot-reloading of the browser while developing the application, etc. works. The setup of Webpack can take a couple of hours or a couple of days depending on your skills, luck and demand functionality.



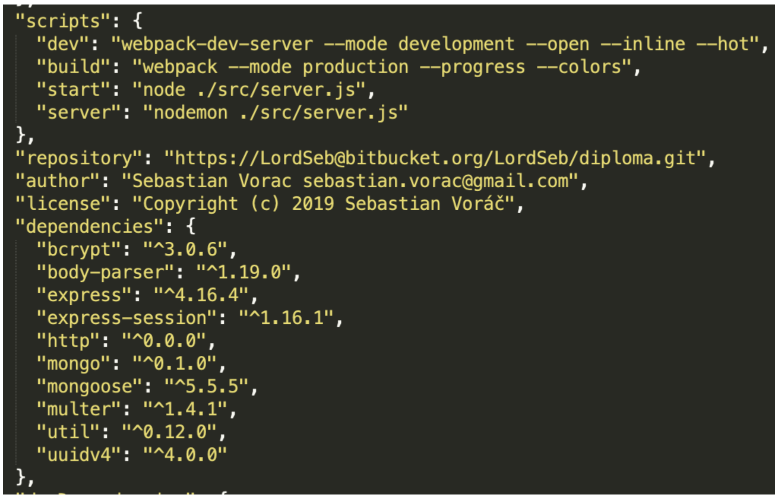
```
"devDependencies": {
  "@babel/core": "^7.4.4",
  "@babel/plugin-proposal-class-properties": "^7.4.4",
  "@babel/preset-env": "^7.4.4",
  "@babel/preset-react": "^7.0.0",
  "@svgr/webpack": "^4.2.0",
  "babel-core": "^6.26.3",
  "babel-loader": "^8.0.5",
  "babel-plugin-styled-components": "^1.10.0",
  "babel-preset-env": "^1.7.0",
  "babel-preset-react": "^6.24.1",
  "classnames": "^2.2.6",
  "cors": "^2.8.5",
  "date-fns": "^1.30.1",
  "file-loader": "^3.0.1",
  "formik": "^1.5.4",
  "nodemon": "^1.19.0",
  "react-minimal-pie-chart": "^4.0.0",
  "react-redux": "^7.0.3",
  "react-router-dom": "^5.0.0",
  "react-router-redux": "^4.0.8",
  "react-select": "^2.4.3",
  "redux": "^4.0.1",
  "redux-thunk": "^2.3.0",
  "styled-components": "^4.2.0",
  "webpack": "^4.30.0",
  "webpack-cli": "^3.3.1",
  "webpack-dev-server": "^3.3.1",
  "yup": "^0.27.0"
}
```

Figure 4.1: All packages used for development and client-side use



```
"scripts": {
  "dev": "webpack-dev-server --mode development --open --inline --hot",
  "build": "webpack --mode production --progress --colors",
  "start": "node ./src/server.js",
  "server": "nodemon ./src/server.js"
},
"repository": "https://LordSeb@bitbucket.org/LordSeb/diploma.git",
"author": "Sebastian Vorac sebastian.vorac@gmail.com",
"license": "Copyright (c) 2019 Sebastian Voráč",
"dependencies": {
  "bcrypt": "^3.0.6",
  "body-parser": "^1.19.0",
  "express": "^4.16.4",
  "express-session": "^1.16.1",
  "http": "^0.0.0",
  "mongo": "^0.1.0",
  "mongoose": "^5.5.5",
  "multer": "^1.4.1",
  "util": "^0.12.0",
  "uuidv4": "^4.0.0"
},
```

Figure 4.2: All packages used for the application server-side use along with yarn task ranners

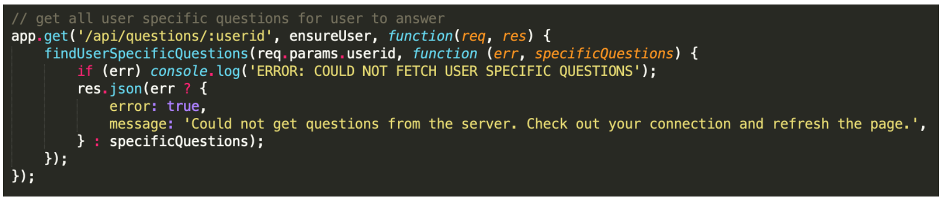


```
"scripts": {
  "dev": "webpack-dev-server --mode development --open --inline --hot",
  "build": "webpack --mode production --progress --colors",
  "start": "node ./src/server.js",
  "server": "nodemon ./src/server.js"
},
"repository": "https://LordSeb@bitbucket.org/LordSeb/diploma.git",
"author": "Sebastian Vorac sebastian.vorac@gmail.com",
"license": "Copyright (c) 2019 Sebastian Voráč",
"dependencies": {
  "bcrypt": "^3.0.6",
  "body-parser": "^1.19.0",
  "express": "^4.16.4",
  "express-session": "^1.16.1",
  "http": "^0.0.0",
  "mongo": "^0.1.0",
  "mongoose": "^5.5.5",
  "multer": "^1.4.1",
  "util": "^0.12.0",
  "uuidv4": "^4.0.0"
},
```

Figure 4.3: All packages used for the application server-side use along with yarn task ranners

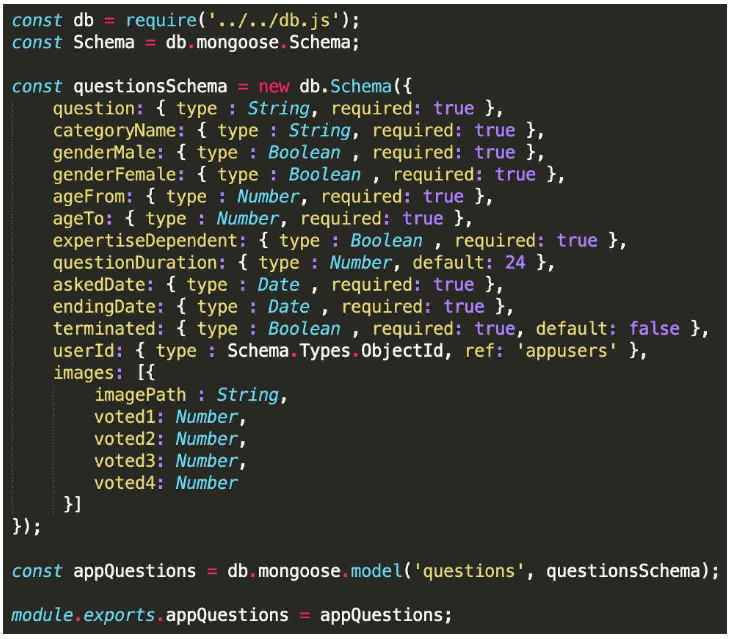
4.2 Server and Database

The node.js server have been created with the help of express.js and the proper API routes for the fetching and posting user data have been created. The server API takes the client request and ensures if the client has a permission to do such a request by middleware callback method **ensureUser**. The server has a session generation implemented. Server sends the session to the client when the client log in successfully. For this mater only API routes for register and login are open and do not demand to ensure if the user is logged in. When the user has permission to call API request, then the server sends it to the corresponding mongoose methods to process the request while he is waiting for a callback with the data returned or error reported. The mongoose method will process the request with the data and accordingly to defined database schemas on the server makes a particular operation in the database.



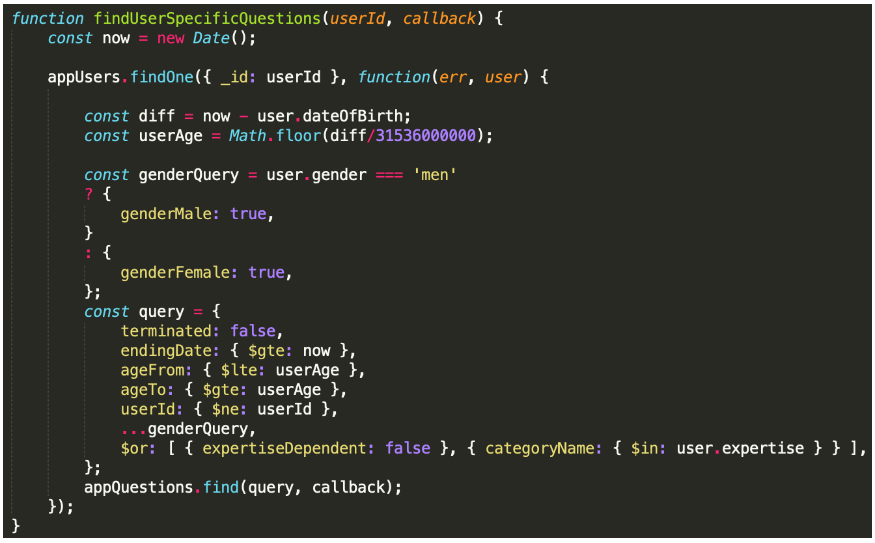
```
// get all user specific questions for user to answer
app.get('/api/questions/:userid', ensureUser, function(req, res) {
    findUserSpecificQuestions(req.params.userid, function (err, specificQuestions) {
        if (err) console.log('ERROR: COULD NOT FETCH USER SPECIFIC QUESTIONS');
        res.json(err ? {
            error: true,
            message: 'Could not get questions from the server. Check out your connection and refresh the page.',
        } : specificQuestions);
    });
});
```

Figure 4.4: Server API for fetching user questions example



```
const db = require('../../db.js');
const Schema = db.mongoose.Schema;

const questionsSchema = new db.Schema({
    question: { type : String, required: true },
    categoryName: { type : String, required: true },
    genderMale: { type : Boolean , required: true },
    genderFemale: { type : Boolean , required: true },
    ageFrom: { type : Number, required: true },
    ageTo: { type : Number, required: true },
    expertiseDependent: { type : Boolean , required: true },
    questionDuration: { type : Number, default: 24 },
    askedDate: { type : Date , required: true },
    endingDate: { type : Date , required: true },
    terminated: { type : Boolean , required: true, default: false },
    userId: { type : Schema.Types.ObjectId, ref: 'appusers' },
    images: [{
        imagePath : String,
        voted1: Number,
        voted2: Number,
        voted3: Number,
        voted4: Number
    }]
});

const appQuestions = db.mongoose.model('questions', questionsSchema);

module.exports.appQuestions = appQuestions;
```

Figure 4.5: Database schema example

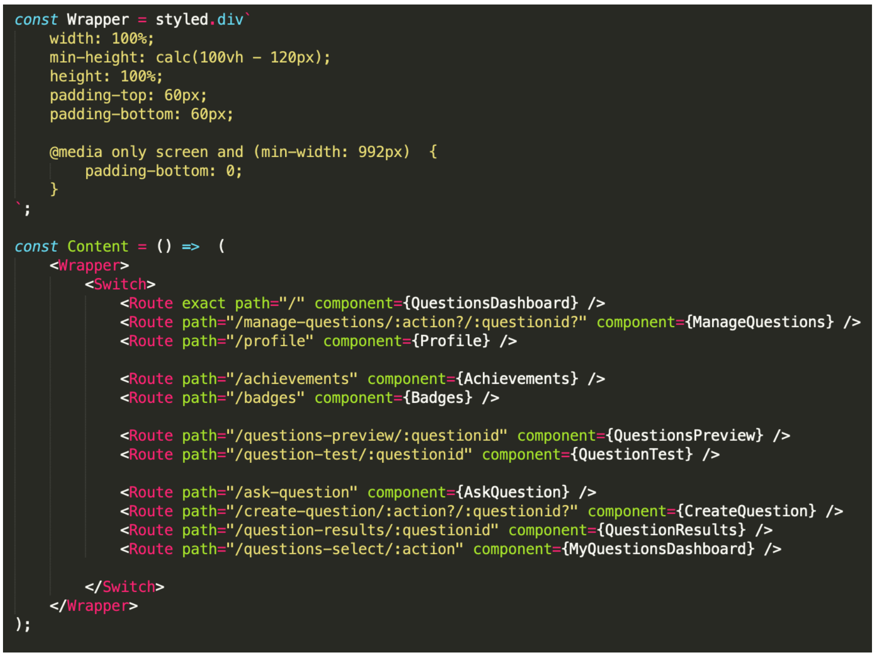


```
function findUserSpecificQuestions(userId, callback) {
    const now = new Date();

    appUsers.findOne({ _id: userId }, function(err, user) {

        const diff = now - user.dateOfBirth;
        const userAge = Math.floor(diff/31536000000);

        const genderQuery = user.gender === 'men'
        ? {
            genderMale: true,
        }
        : {
            genderFemale: true,
        };
        const query = {
            terminated: false,
            endingDate: { $gte: now },
            ageFrom: { $lte: userAge },
            ageTo: { $gte: userAge },
            userId: { $ne: userId },
            ...genderQuery,
            $or: [ { expertiseDependent: false }, { categoryName: { $in: user.expertise } } ],
        };
        appQuestions.find(query, callback);
    });
}
```

Figure 4.6: Mongoose model method example

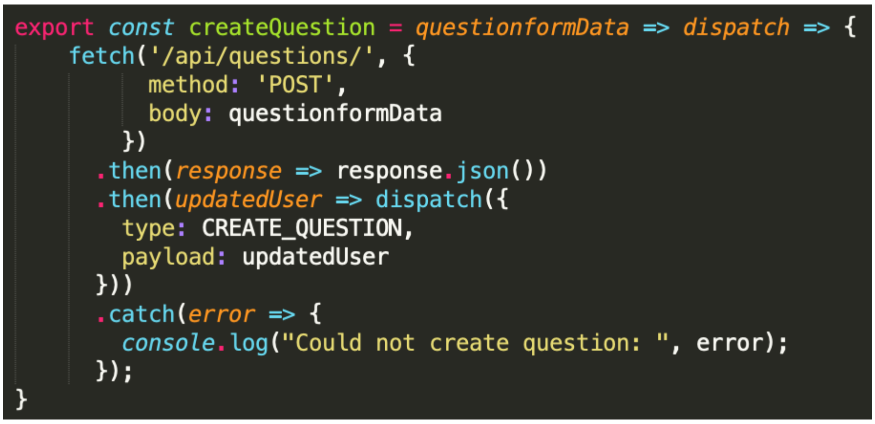
4.3 Client-side of the application

The client side of the application is implemented with the respect of the principle and functionality described in subsection of application architecture 3.2.2. This main approach is used in the whole client application. Each even tiny piece of the client application React or Styled component depending if it demands some functionality or the component only needs to be styled. There are two types of React components. One type of component is purely functional without own logic and just passing the properties to other components or respond on change of the properties by changing their view.



```
const Wrapper = styled.div`
    width: 100%;
    min-height: calc(100vh - 120px);
    height: 100%;
    padding-top: 60px;
    padding-bottom: 60px;

    @media only screen and (min-width: 992px)  {
        padding-bottom: 0;
    }
`;

const Content = () =>  (
    <Wrapper>
        <Switch>
            <Route exact path="/" component={QuestionsDashboard} />
            <Route path="/manage-questions/:action?/:questionid?" component={ManageQuestions} />
            <Route path="/profile" component={Profile} />

            <Route path="/achievements" component={Achievements} />
            <Route path="/badges" component={Badges} />

            <Route path="/questions-preview/:questionid" component={QuestionsPreview} />
            <Route path="/question-test/:questionid" component={QuestionTest} />

            <Route path="/ask-question" component={AskQuestion} />
            <Route path="/create-question/:action?/:questionid?" component={CreateQuestion} />
            <Route path="/question-results/:questionid" component={QuestionResults} />
            <Route path="/questions-select/:action" component={MyQuestionsDashboard} />

        </Switch>
    </Wrapper>
);
```

Figure 4.7: React functional component with styled component example



```
export const createQuestion = questionformData => dispatch => {
    fetch('/api/questions/', {
          method: 'POST',
          body: questionformData
        })
      .then(response => response.json())
      .then(updatedUser => dispatch({
        type: CREATE_QUESTION,
        payload: updatedUser
      }))
      .catch(error => {
        console.log("Could not create question: ", error);
      });
}
```

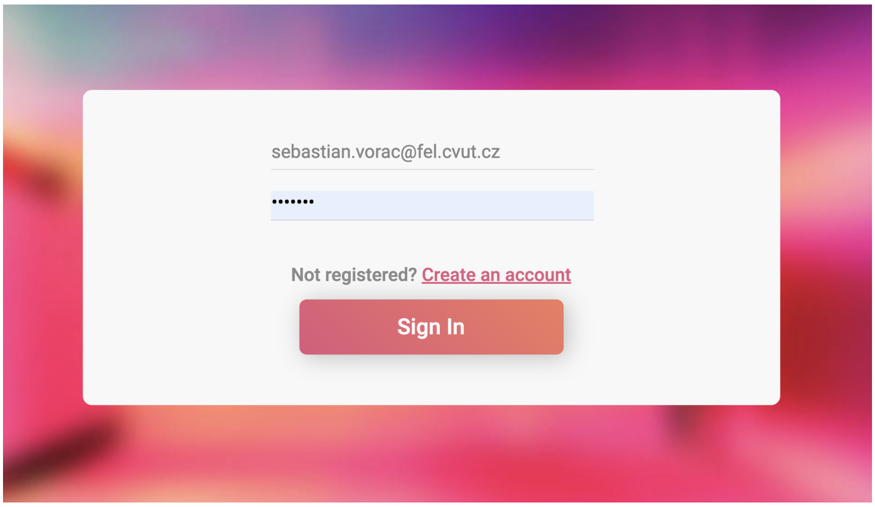
Figure 4.8: Redux action creator method example - fetch

4.4 Implementation results

All defined functional requirements in section 2.5 and non-functional requirements defined in section 2.5.4 have been implemented with the exception of these two:

- User is being informed about the new questions matching his profile setting by a web notification (if enabled)
- User can undo his choice in the question test within 3 seconds.

Implementation met all diagrams defined in this Thesis. Also all principals and best practice in web application development have been met as it can be seen in the application source code. The result of the application is shown in the figures below.

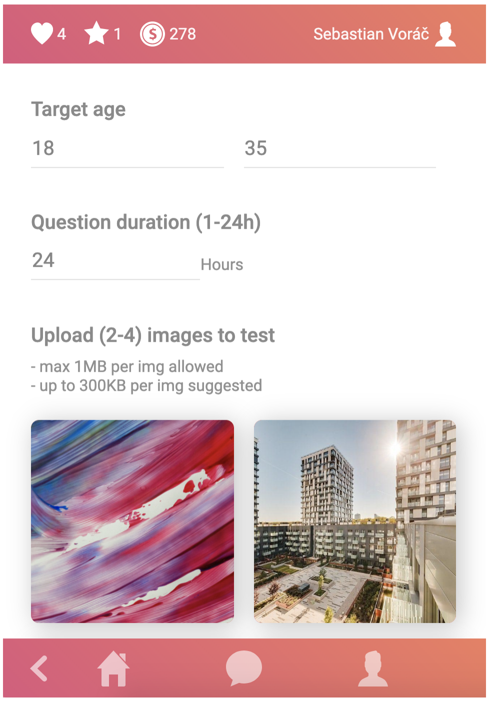


sebastian.vorac@fel.cvut.cz

•••••••

Not registered? Create an account

Figure 4.9: Login page of the application



Target age

18 35

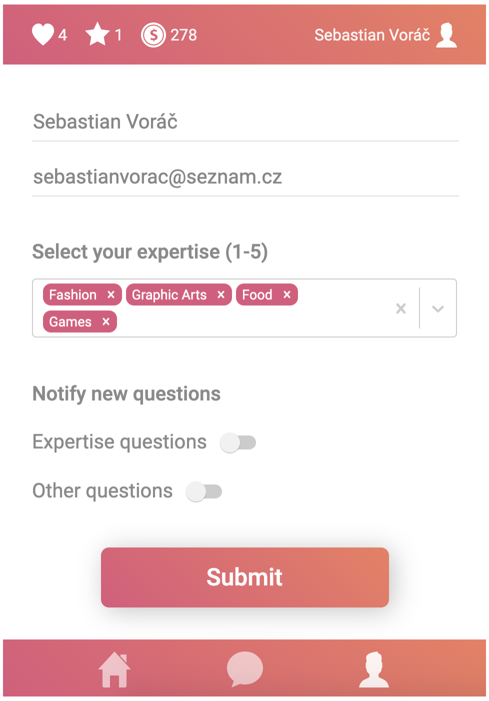
Question duration (1-24h)

24 Hours

Upload (2-4) images to test

- max 1MB per img allowed
- up to 300KB per img suggested

Figure 4.10: Creating a question



Sebastian Voráč

sebastianvorac@seznam.cz

Select your expertise (1-5)

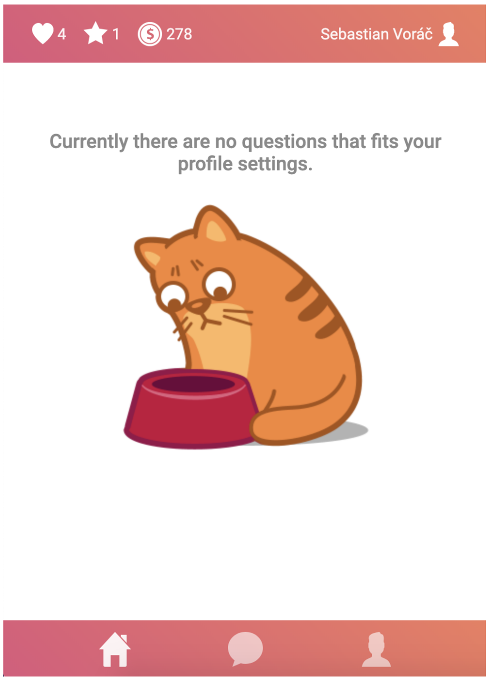
Fashion × Graphic Arts × Food × Games ×

Notify new questions

Expertise questions

Other questions

Figure 4.11: Profile of the user



Currently there are no questions that fits your profile settings.

Figure 4.12: Questions Dashboard

Chapter 5

Testing

In this chapter, there is enhanced the process of user testing with the user testing scenarios and results with recommendations which is the key information in UCD. All testing has been made qualitatively with individual users one by one. The user scenarios have been followed in the testing process of the prototypes.

5.1 Sketches

The test goal was to provide a simple dashboard layout, where users can clearly see the questions from other users filtered from each other by question categories.

I had peace of paper representing the questions thumbnails on the dashboard along with a few versions of the sketched questions dashboard designs.

The testing procedure was informal with 6 participants. The quick validation of sketched design was needed so the testing was conducted with the family and closest friends. The environment was home or school.

The sketched version of the design where questions were filtered by buttons with the name of the categories provided itself to be suboptimal on the small mobile devices with more categories so this design variant was not considered. On the other hand, a variant where the questions were sorted to the categories and were horizontally scrollable within the category was successful with one recommendation - it must be visible that there is a content on the right side of the displayed question thumbnails. The recommendation was processed to the design by always showing part of the question thumbnail on the mobile devices, so the users can immediately realize, there is another content they can scroll to and display.

5.2 Low-fidelity prototype

The test goal was to get a better idea about how the users want to interact with the application, so the main focus in low-fidelity prototype testing was to test answering the question, creating the question, viewing the results of the question and editing a profile on the paper prototype according to the test scenarios described below.

The paper prototype was provided to 11 users in the office. After realizing the fact that some participants are not feeling comfortable when they have to imagine interactions with the system on the paper I came to the conclusion that making a clickable wireframe would be a better solution in order to properly go through the test scenarios described below. The user testing was then repeated with the wireframes together on 7 participants.

Test scenario: Providing User information

1. Open your profile
2. Fill up your name and other information
3. Set your major Fashion and Graphic design
4. Remove the ability graphic design
5. Add minor ability
6. Display your badges

Test scenario: Test scenario: Answering a question

1. Move questions more to the left
2. Open a question on a fashion topic
3. Get to the other questions detail
4. Answer to some question
5. View all possible picture answers
6. Select one

Test scenario: Question submit

1. Type the actual question
2. Select its category

3. Upload 3 images to test
4. Remove the last 3rd image
5. Submit the question

Test scenario: Displaying test result

1. Go to the results tab
2. Click on particular fashion question
3. Display the images order

Results and recommendations from the user testing:

- Results of the test should display images sorted according to their rank obtained in the testing process, not in the order the user inserted them.
- Geometric shapes, which aid the navigation of the users with impaired color recognition ability, are not taken advantage of by the majority of the users. However, the option to enable them should be placed in the profile while retaining the colors as a default.
- The user should not view images in a cycle. The first image should not have the arrow to the left, and the last image should not have the arrow to the right.
- The user knows the number of images. However, only the upper indicators indicate the number of comparisons. I should give it clearly.
- I should replace the graph icon with the image icon in the menu. The manages questions apart from displaying images. The icon is, therefore, misleading.
- Three seconds is not sufficient time for the undo action. I should increase the time for the action to five seconds.
- The user should recognize the return button as disabled. It should not be present in any root category, only in nested categories. It should navigate back to the root categories, but no further except for the upper legend icons, where it should lead back to the dashboard.
- I should display both types of results (first-place votes and overall) with icons providing a thorough explanation of the metric upon click.

5.3 High-fidelity prototype

The main goal of the testing was to ensure that the 3 most complex and possibly more problematic parts of the application are designed well and it is ready to be implemented.

The testing environment was again a quiet room in the office. There were three iterations of the user testing with the 7, 9, 11 and 9 participants for particular tests. The participants have been told to follow the testing scenarios down below and share loudly their actual thoughts as well as providing feedback after the test procedure.

Test scenario: Set up your profile

1. Click on the profile tab
2. Add minor skill fishing
3. Remove major skill Graphic design
4. Turn on the minor questions' notification
5. Save changes

Test scenario: Answer the question

1. On Dashboard swipe left on fashion topic row to see 3rd question preview
2. Click on some question in the Fashion category
3. Look at the all possible question previews
4. Select one question preview to answer
5. Select one question preview to answer
6. Look at the both A / B variant
7. Hold tap to see the full picture and then hold tap again to get back
8. (Read till the end before you do) : Select one answer but then undo your choice and select the other one with "This".
9. Now just select the first option (there are no other options for prototype simplicity)
10. Return to the dashboard

Test scenario: Manage questions

1. Go to manage questions section
2. End one of the question
3. Go back and create a new question
4. Look to the available categories (without closing it)
5. Upload Image in there (don't worry about the change of fields to invalid etc.)
6. Remove the third image
7. Submit the question
8. Go to the question results and determine / describe how the picture testing ended up

Results and recommendations from the user testing:

- The participant wanted to click at the upper left menu with the hear icon and the dollar icon.
- The participant wanted to know the meaning of the hearts and what can be done with the dollars. Solution: Introduce a new page explaining the meaning of the virtual currency, which the user can use to ask a question, and the hearts, which indicate how many people the user has helped.
- Lower case letter informing about how many skills remains to be added is not well visible.
- The participant would like a "tiled" display of the skills in case of only two major and four minor skills.
- The should badges button appears to the participant as a save button.
- The participants would expect a save button or a least a notification that their input was saved. Solution: Introduce the save changes button, which saves the whole profile page to the server.
- Some participants were not instantly aware that individual question thumbnails are questions.
- The numbers at the question thumbnails confused the participants. Some of them thought they denote the number of images present in the question, not the time remaining. Solution: add hour unit("h") behind the number, so that the meaning is clear. Establish the question thumbnails as questions and the numbers as the time at the first app load.
- Participants did not find it apparent that horizontal swipe between the question thumbnails is possible. Solution: indicate the functionality explicitly in the design of the app.

- The participants would like to return to where they left off before answering a question, not browse the questions from the beginning after they left. Solution: Save the position where the user was on the page and redirect them to it after a question is answered.

5.4 Final Application

Thanks to the application deployment to the server, the application could be tested in the form of participants "standard" usage. The application has been sent to the friends and relatives with freedom of feedback provided as well as it was tested 6 times - each time with one user who has been observed as he has been fulfilling the assigned tasks. A lot of needless to mention technical issues were discovered and fixed until nobody has been experiencing any bug while using the application. Manual testing of the functionality can be more troublesome when the application grows, so in future work I would suggest to implementation of the automized unit test created by delightful JavaScript Testing Framework - Jest.

5.5 Chapter summary

This chapter provided a detailed view of user testing with the user scenarios along with the results and recommendations which have been processed in the next iterations of the UCD methodology. Automatized unit testing of the application has been suggested in order to be able to test and maintain the correctness of the application main functionality.

Chapter 6

Conclusion and future work

The aim of this Theses was to design and develop a platform that would provide simple fast user feedback with the respect of UCD. In section 2.3 I was able to do effective user research that led me to the right path of designing and specifying the platform. In section I developed own scoring system for gaining user test feedback. Then I was able to create low-fidelity and high-fidelity prototypes of the application and I was ready for the implementation of the web application. However, in the subsection 2.5.3 I defined almost 60 functional requirements for the application. It has been done with good intentions to do the platform the best way I could possibly imagine, but the implementation of the application with so many requirements became incredibly time-consuming. I managed to implement them all in the end and made the application ready for the commercial use so I finally fulfilled the goal of this work. I hope my solution will help thousands of users in the future to provide and receive simple user feedback.

Possible future work on this application would be for example automatic size reduction of all images uploaded on the server instead o allowing the maximum size of 1MB per image and suggesting user to reduce image size on his own. Because gamification is already implemented in the form of badges and virtual money receiving, the next possible extension could be monetization for companies and possible money gain for the users. The idea could be that companies would pay users to give feedback on their business questions. The last thing that could be implemented would be undone button in the test while voting. However, the application is well prepared for this feature but it was not implemented due to time pressure as well as the notifications on the new questions which appeared to be not very useful when the questions with dashboard reload itself every 30 seconds. Therefore, the implementation of notifications has been considered as a minor feature and have not been implemented.

Bibliography

[1]

[2] Formuláře google – zdarma vytvářejte a analyzujte průzkumy. URL: `https://www.google.com/intl/cs_CZ/forms/about/`.

[3] React – a javascript library for building user interfaces. URL: `https://reactjs.org/`.

[4] Gerald Albaum. The likert scale revisited. *Market Research Society. Journal.*, 39(2):1–21, 1997.

[5] Babel · the compiler for next generation javascript. URL: `https://babeljs.io/`.

[6] Alan Cooper, Robert Reimann, and David Cronin. *About face 3: the essentials of interaction design*, volume 3. John Wiley & Sons, 2007.

[7] Alan Cooper, Robert Reimann, and David Cronin. *About face 3: the essentials of interaction design*, volume 3. John Wiley & Sons, 2007.

[8] Rob Crowther, Joe Lennon, Ash Blue, and Greg Wanish. *HTML5 in Action.* Manning Publications Co., 2014.

[9] M. Jose Escalona and Nora Koch. Requirements engineering for web applications-a comparative study. *J. Web Eng.*, 2(3):193–212, 2004.

[10] Node.js web application framework. URL: `https://expressjs.com/`.

[11] Overview · formik. URL: `https://jaredpalmer.com/formik/docs/overview`.

[12] Node.js Foundation. URL: `https://nodejs.org/en/`.

[13] Node.js Foundation. URL: `https://nodejs.org/en/`.

[14] Why did google plus fail? a google autopsy, Oct 2018. URL: `https://edgy.app/why-did-google-plus-fail-a-google-autopsy`.

[15] Hana Kanisová and Miroslav Müller. *UML srozumitelně*. Computer press, 2006.

[16] Aaron R Lyon and Kelly Koerner. User-centered design for psychosocial intervention development and implementation. *Clinical Psychology: Science and Practice*, 23(2):180–200, 2016.

[17] Filippo Menczer, Gautam Pant, and Padmini Srinivasan. Topical web crawlers: Evaluating adaptive algorithms. *ACM Transactions on Internet Technology (TOIT)*, 4(4):378–419, 2004.

[18] Michael Mikowski and Josh Powell. *Single page web applications: JavaScript end-to-end*. Manning Publications Co., 2013.

[19] The most popular database for modern apps. URL: `https://www.mongodb.com/`.

[20] Mongoose. URL: `https://mongoosejs.com/`.

[21] Neoteric. Single-page application vs. multiple-page application, Dec 2016. URL: `https://medium.com/@NeotericEU/single-page-application-vs-multiple-page-application-2591588efe58`.

[22] Next.js. URL: `https://nextjs.org/`.

[23] Cost of user testing a website. URL: `https://www.nngroup.com/articles/cost-of-user-testing-a-website/`.

[24] npm. URL: `https://www.npmjs.com/`.

[25] Redux · a predictable state container for js apps. URL: `https://redux.js.org/`.

[26] Naomi B Robbins, Richard M Heiberger, et al. Plotting likert and other rating scales. In *Proceedings of the 2011 Joint Statistical Meeting*, pages 1058–1066, 2011.

[27] Styled-Components. components. URL: `https://www.styled-components.com/`.

[28] Libuše Svobodová and Miloslava Černá. Project management model with designed data flow diagram: The case of ict hybrid learning of elderly people in the czech republic. In *International Conference on Computational Collective Intelligence*, pages 399–408. Springer, 2018.

[29] Techopedia. What is asynchronous javascript and xml (ajax)? - definition from techopedia. URL: `https://www.techopedia.com/definition/24402/asynchronous-javascript-and-xml-ajax`.

[30] Susan Ward. What is search engine optimization?, Dec 2018. URL: `https://www.thebalancesmb.com/search-engine-optimization-2948419`.

[31] webpack. URL: https://webpack.js.org/.

[32] yarn. URL: https://yarnpkg.com/lang/en/.

[33] yup. URL: https://www.npmjs.com/package/yup.