SmartSparks - A Crowdsourcing App for Organisations

Friday 22nd January, 2021 - 21:28

Iris Kremer
University of Luxembourg
Email: iris.kremer.001@student.uni.lu

Steffen Rothkugel
University of Luxembourg
Email: steffen.rothkugel@uni.lu

***Abstract*—This project seeks to explore the domains of crowdsourcing and gamification. It also includes the development an application called SmartSparks, that enables organisations and companies to perform crowdsourcing among their employees and customers. In addition to its crowdsourcing features, the application uses gamification concepts to motivate people to contribute. Therefore, the scientific deliverable of this project is a discussion on crowdsourcing and gamification, while the technical deliverable is a prototype of the application SmartSparks.**

1. Introduction

Crowdsourcing is a powerful and sometimes underestimated technique to gather data from the crowd. We often do not realise that it is used everywhere, and that we are part of it as we correct a mistake in an article on Wikipedia, rate a movie on Netflix or train the YouTube algorithm while watching videos. But while the rise of the internet made some crowdsourcing tasks much easier, like those presented just before, it is still hard to motivate people to contribute when it comes to less interesting topics. That is a pity, because crowdsourcing is very convenient for instance in scientific research to gather necessary scientific data through or from the entire population, or within companies or groups of people to gather new ideas more efficiently. We therefore need to encourage people to contribute to such crowdsourcing problems, and an efficient solution is gamification, where the crowd is motivated to contribute by triggering similar drives and pleasures than when playing games.

The detailed project description, including domains and target deliverable, can be found in section 2, while the prerequisites for the project are described in section 3.

The sections 4 and 5 then present the concrete scientific and technical deliverables. Each of these two sections is subdivided into requirements, design, production and assessment, to cover the entire development process of these two deliverables.

Finally, we conclude on the project and present future perspectives in section 6.

2. Project description

2.1. Domains

2.1.1. Scientific. The two scientific domains of this BSP are crowdsourcing and gamification. These two domains, although reaching beyond strictly computer science, are very relevant in computer science, especially in the context of this project, were we focus on data and ideas collection.

Crowdsourcing, sometimes called **Human Computation** [1], aims to use the innovation and computing power of the crowd to solve problems which are difficult to solve with only a small number of specialists. The concerned crowd is a large group of people who usually have limited or no knowledge of the concerned issue (they don't need any), and problems addressed using crowdsourcing can for instance be collecting a large amount of data for research purposes, coming up with solutions to specific issues - like in this project - or even training an artificial intelligence.

Gamification is the process of using game elements to turn a problem into a game, with the goal to increase the interest and motivation of people approaching this problem [2]. In this project, gamification concepts are used to make the app feel closer to a game and motivate the users to contribute.

2.1.2. Technical. Since an prototypical app is developed in this project, the technical domains concern the engineering technologies used in the app development: Flutter and Dart for the frontend and Firebase for the backend.

Flutter is a software development kit (SDK) created by Google. It was made to develop applications with neat user interfaces (UIs) that can be deployed on several different platforms from a single codebase, written in the programming language **Dart**.

Firebase is also a platform developed by Google, offering several types of backend services, such as authentication and NoSQL databases, which are used in this project.

2.2. Targeted Deliverables

2.2.1. Scientific Deliverables. The scientific deliverable of this project is a discussion of crowdsourcing in general,

together with the potential and the impact gamification concepts provide in this context. It comprises the following elements:

1) A definition of crowdsourcing, scientific explanations of how and why it works and some real or fictional examples
2) A definition of gamification, a presentation of its core concepts, and how they can be applied effectively in a crowdsourcing application

2.2.2. Technical Deliverables. The technical deliverable of this project is the prototype of an application, SmartSparks, primarily designed for organisations, such as a company, to perform crowdsourcing among their members, employees, customers or even just anyone if they wish to, but equally useful also in other, similar contexts. The application uses gamification to make the user experience more enjoyable, and to increase the engagement of the crowd, the goal being to promote innovation. The application also addresses issues such as how to select the best solutions in the pool of ideas. The application is intended to showcase the most essential features. The limited time does not allow to develop a fully operational product.

3. Pre-requisites

3.1. Scientific pre-requisites

The scientific pre-requisite for this project is to get familiar with the state of the art of crowdsourcing and gamification.

3.2. Technical pre-requisites

This project did not have particular pre-requisites, but previous knowledge in some domains were useful to learn and understand technologies used faster and better.

Previous knowledge on object-oriented programming was useful for learning and using Dart. The experience of using React, a JavaScript library for building user interfaces, contributed to a faster understanding of Flutter. Having worked previously with NoSQL databases and, in particular, previous experience with Firebase were also very useful in this project. And finally, since we used GitHub to share and backup the code, experience with this technology was also helpful.

4. On the Use of Gamification in Crowdsourcing

4.1. Requirements

The goal here is to present the domains of crowdsourcing and gamification and to highlight how gamification benefits crowdsourcing. Therefore, we define the following requirements on the content of the deliverable:

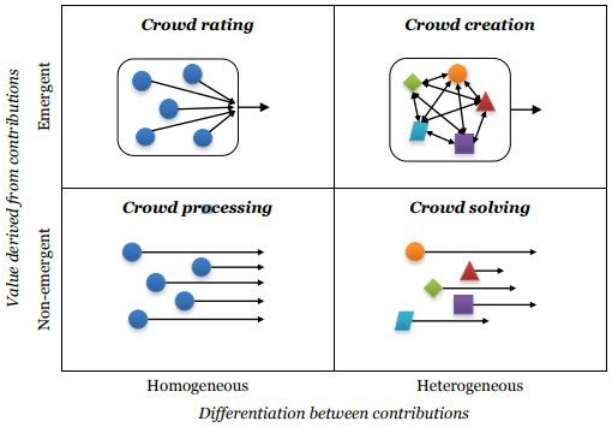


Figure 1: Four different types of crowdsourcing [3].

1) To have a basis for context, the scientific deliverable must contain a definition of crowdsourcing, to which is always referred to in this report whenever we mention crowdsourcing. Some examples of crowdsourcing methods or applications may be presented to support this definition.
2) For the same reason, it must also provide a definition of gamification, to which is referred to whenever we mention gamification in this report.
3) The deliverable must present some general concepts of gamification. Examples may be used to illustrate these concepts easier.
4) It must explain why gamification works and how it can be used in crowdsourcing.
5) To support the use of gamification in crowdsourcing, some concrete examples of crowdsourcing games and gamified crowdsourcing applications must be presented.

Additionally, some non-functional requirements have been set as well:

1) The deliverable must present scientific evidences such as references to support the definitions and explanations.
2) The author may include some own critical thinking in the reflections.

4.2. Design

The scientific deliverable, which will be presented in section 4.3, is composed of two sections. The first section will introduce the reader to the domain of crowdsourcing, and the second deals with gamification and its applications in crowdsourcing.

4.2.1. Crowdsourcing Section. Four different types of crowdsourcing were identified in article [3], as depicted in figure 1: Crowd processing, crowd solving, crowd rating and crowd creation. Each of these types have their specific criteria and address other kinds of problems with different

crowds. Therefore, after introducing the subject with a short definition of crowdsourcing, we will present these four types of crowdsourcing and their differences. We will then give some real examples of crowdsourcing applications in computer science and other domains. Finally, we will explain the benefits of such applications and their issues, to transition over to gamification.

4.2.2. Gamification in Crowdsourcing Section. Just like the crowdsourcing section, we also start here with a definition of gamification. We then present some examples of gamification concepts, alongside with explanations of why they work and how they increase people's motivation to participate. We will then derive from these explanations how gamification can be applied and benefit crowdsourcing, using some examples of gamified crowdsourcing applications as support for our explanations.

4.3. Production

4.3.1. Crowdsourcing. The term "**crowdsourcing**" is composed of the words "crowd" and "to source".

In the context of this domain, a crowd is referring to "any large and possibly random group of people, often gathered online". It does not follow entirely a standard definition of the word crowd as described in [4] due to the possible online aspect of our crowd definition, but the concept is very similar. To source means "to get something from a particular place" [5]. In this context, what exactly can be obtained can be all types of information. Therefore, we define crowdsourcing as "*obtaining information from a large, possibly random group of people*".

Many definitions of crowdsourcing given in scientific articles such as [6] and [7] include that the crowd of people is an online crowd, since the rise in popularity, attention and effectivity of crowdsourcing in the last decades has a lot to do with the rise of the internet itself. While it is often the case that crowdsourcing is performed online, we believe that restricting crowds to be online in its definition is not necessary. Procedures like brainstorming with a large group of people at the office or conducting interviews on the street are therefore also included in our definition of crowdsourcing.

In computer science, crowdsourcing is sometimes referred to by another name: "**human computation**". This term reflects better the usage of crowdsourcing in computer science applications, which is to outsource certain computation tasks to humans, because they are easy to solve for humans, but very difficult to solve by a computer. Some examples of such human computation games developed by Luis von Ahn, one of the pioneers of crowdsourcing, are presented in section 4.3.2.

As already mentioned in section 4.2, article [3] distinguishes four different archetypes of crowdsourcing systems, depicted in figure 1, and all crowdsourcing applications correspond more or less to one of these types. They are the result of different combinations of two criteria: homogeneity of contributions and emergence of value from the contributions. Here is a shortened version of the explanations of article [3]:

- A **homogeneous** system will value all contributions equally, because these contributions are identical in terms of quality. On the other hand, a **heterogeneous** system values each contribution differently, as these contributions can be seen as alternative solutions or complements to each other.
- Emergence is a term that denotes a property that is not possessed by any individual component, but results from the relationships between these components. Therefore, a system seeking **emergent** value will derive the value from the relationship of its contributions, while a system that derives **non-emergent** value will get its value directly from the individual contributions.

Crowdsourcing is useful whenever we want to gather a large amount of data that can be provided easily by a random or a specific crowd of people, and it is very effective when correctly performed. According to [8], "85% of 2014 Best Global Brands have used crowdsourcing in the last 10 years", which shows a clear tendency of good and big companies to use crowdsourcing.

Our interconnected world greatly eases the process of crowdsourcing by providing means to communicate with and reach millions of people very easily. In fact, the internet itself is more or less a giant crowdsourcing platform. We can also consider individual entities on the internet such as Wikipedia or YouTube as places where crowdsourcing is performed - these two would fall into the category of heterogeneous and emergent, so crowd creation systems [3].

Another perhaps more obvious example of crowdsourcing is the "Completely Automated Public Turing test to tell Computers and Humans Apart", better known by its acronym "CAPTCHA". These mini tasks or puzzles we must occasionally solve to prove that we are humans when surfing on the internet are actually also a mean for collecting valuable data for research purposes. In particular, the human responses obtained by the system "reCAPTCHA" are used as training data for machine learning projects [9].

However, while users feel a natural interest for contributing to systems such as Wikipedia and YouTube and are forced to solve CAPTCHAs to prove they are humans, there are also many application where crowdsourcing would be very effective, but where users do not voluntarily contribute, simply because they have no reason to. In these cases, the users need to be motivated through other means, and the simple purpose of the data collection, even an excellent one, is often insufficient. One solution to this problem is gamification.

4.3.2. Gamification in Crowdsourcing. As already defined in section 2.1.1, **gamification** is the process of using game elements to turn a problem into a game [2]. The goal is to increase the interest and motivation of people approaching this problem, which ultimately will also increase creativity

and innovation, thus the likelihood of solving the problem. Similarly to crowdsourcing with the internet, gamification is often associated with digital support, but nothing in its definition excludes the possibility of non-digital gamification approaches.

"Intrinsic reinforcement" and the "winner effect" are the biological and psychological effects of achieving goals we set to ourselves, and the main sources of pleasure in video games, which in turn result in increased motivation to play more [10]. Depending on the problem, other psychological effects, such as "easy failure" that gives a sense of safety for instance, will also benefit the users. Gamification aims to reproduce these effects on us when we are dealing with non-game problems by making them appear or feel like games [11].

The term gamification includes a wide range of techniques. One common approach is to include game elements, such as points, levels and rankings, or only a subset of them, in the process used to solve the problem (e.g. an application), to make it more similar to standard combat or adventure games [12]. This gamification technique will most likely not mask the target problem that is to be solved, but will still increase people's motivation to contribute. Another is to design an entire new game specifically for the problem, so-called "Games with a purpose" [13]. There, players might not even be aware of the problem they are contributing to solve, they may just play the game for the fun it provides them. A third way is to incorporate the problem into an existing game, like "project discovery" from the video game "Eve online"[1], in which case player may or not be aware of the problem they are helping to solve.

The techniques mentioned can also be combined and of course, this list is not exhaustive. Depending on the problem and target application of gamification, the most efficient techniques will vary as well.

Gamification has already proven to be effective in numerous applications, such as raising awareness about and getting people to recycle their waste [14], increasing motivation in e-learning [15], [16] (which can be especially interesting in the current COVID-19 pandemic context) and the numerous examples given in [10].

In crowdsourcing, gamification is also very effective to solve lack of motivation issue, and there are several successful examples of gamified crowdsourcing application. "Games with a purpose" [13] were mentioned, and in particular, the ESP game (later renamed into Google Image Labeler), PeekaBoom and Verbosity are good examples of crowdsourcing games with a purpose [1]. FoldIt[2] [17] is another example, famous for having enabled scientists to solve a very difficult protein structure problem that they had been working on for over 15 years within less than 10 days [18]. There are more such games that help advance science: We already cited "project discovery" previously, and EteRNA[3], Phylo[4], Eyewire[5] and Quantum Moves[6] are further examples. Aside from games with a purpose, there are also crowdsourcing applications where some game elements were incorporated, such as the example described in [10] from the department of work and pension in the UK, where they built a "stock market for ideas".

Through all these examples, we see that gamification, if implemented correctly, benefits crowdsourcing a lot by attracting and motivating the crowd to contribute, which is essential for crowdsourcing to be effective. Furthermore, according to [6], most scientific research reached the same conclusion, so we can safely conclude that crowdsourcing and gamification work well together.

4.4. Assessment

4.4.1. Fulfillment of the Requirements. The requirements for the scientific deliverable are elicited in section 4.1, and the first part of the assessment consists of validating the requirements.

For the functional requirements, the first is largely validated, because an entire section is dedicated to defining crowdsourcing in details through different angles, giving examples and even distinguishing between different crowdsourcing systems. Requirements 2 and 3 are also validated, as a definition of gamification and some techniques are presented. Requirement 4 is satisfied through an explanation of how and why gamification works, and the examples of gamified crowdsourcing applications are fulfilling requirement 5.

Both non-functional requirements were also fulfilled, because the material presented in the scientific deliverable contains several references and some elements of critical thinking.

4.4.2. Value for the Project. The second part of the assessment is to analyse the value of the scientific deliverable for the project.

The knowledge acquired during the process of making this scientific deliverable is, in the first place, useful for the design of our application SmartSparks. Being aware of the different crowdsourcing systems and gamification approaches enables to make adequate decisions regarding design of SmartSparks and have scientific support for these decisions. Another interesting remark is that the last example in section 4.3.2, i.e. the "stock market for ideas", was the gamified crowdsourcing application that initially inspired SmartSparks.

Of course, the knowledge is also valuable beyond the project scope, because it gives us an insight of the domains

1. https://www.eveonline.com/discovery
2. https://fold.it/
3. https://eternagame.org/
4. https://phylo.cs.mcgill.ca/
5. https://en.wikipedia.org/wiki/Eyewire
6. https://en.wikipedia.org/wiki/Quantum_Moves

of crowdsourcing and gamification, which are two interesting and useful topics to know about.

5. SmartSparks

5.1. Requirements

The aim of this project is to develop a prototypical application called SmartSparks, which will enable organizations to perform crowdsourcing. Since it is a prototype application which is only intended to showcase the main features of the app, the requirements listed here will not correspond to a possible final version of the app. However, to engineer these requirements, we first thought about the features of the final application, which will justify our requirement choices for the prototype requirements. Therefore, we will start with presenting our vision of the final application in section 5.1.1, then describe the requirements for the prototype in 5.1.2.

5.1.1. Vision of the final application. The main idea of the application is to have users create what we called "topics", where they describe an issue for which they need a solution, then other users can suggest solutions to the topics, which are called "sparks" in the app.

However, having such a plain system is not specific enough for our purpose. There are several issues and points of interest that we need to think about to improve this initial concept, some of which are listed below, alongside with our thoughts on how to address them. Since the prototype is primarily going to showcase features, we mainly thought about functional requirements the final application and left out most non-functional requirements.

1) Of course, the first issue is that plain crowdsourcing is not sufficiently exciting to keep people interested in contributing.
    - Therefore, we introduce gamification into the application to make it more engaging and motivate people to contribute.
2) The next problem is that the app in our project is intended for organisations to perform crowdsourcing among a selected crowd such as employees or customers. Therefore, we need users that manage the application, the topics and who is allowed to register and contribute.
    - We introduce two different kinds of users: admin and normal users.
    - Admins must validate new accounts of normal users.
    - Admins only are the ones who create, manage and close topics, while normal users are the ones who can post sparks on these topics.
3) The next important issue identified is that selecting the best ideas among a large amount of them is difficult and tedious.
    - Luckily, we can here again use the power of the crowd to help us with the work, by letting them vote for the sparks they think are the most promising. A spark with a high number of votes is then much more likely to be a good solution.
4) Another problem is that when we only have users posting sparks, these sparks cannot be improved by other users.
    - Here, we decided to introduce a form of collaboration between contributors: comments on sparks. Any normal user can comment on a spark to suggest improvement and point out issues they would detect in a spark.
    - Of course, just like for sparks, users can like comments and comments with many likes are much more likely to be pertinent.
5) We also realised that contributors, i.e. the normal users, can be biased into supporting ideas of their superiors, friends or people they think have more knowledge than them
    - Our first idea was to make normal users remain anonymous to other users at all times. However, this alone is risky, because it will drop the motivation of the contributors, since they will not get recognized and rewarded for their investment. Therefore, we thought of a more elaborated solution for this point.
    - The first thing is that normal users will appear by their usernames to admin users, who, in a real scenario, probably are managers or people with some sort of power in the organisation. This means that for instance, when an employee suggests great ideas to drive the company forwards, he knows that the admins will see his investment and probably reward him. On the opposite, employees posting weak contributions are also directly visible.
    - Secondly, instead of appearing completely anonymous to other normal users, normal users will appear by their rank, which is determined by the amount of points they gain through contributing. This way, they obtain some recognition from their fellow contributors who see their rank appear on sparks they suggested, while still remaining anonymous. Of course, this may bias the contributors into supporting ideas from people with a high rank, but at least, the rank reflects the user's skills and involvement in the crowdsourcing process, which is a much more appropriate measure than someone's status in the company or personal relations with this person.

6) Finally, we thought that new sparks and sparks with few or no votes may be disadvantaged in favor of sparks which already have many votes, but some of these may be excellent ideas. These ideas would therefore be lost due to a lack of visibility.
    - The solution we found for this issue is to promote new and unpopular sparks by making appear by default 2-3 of these, randomly chosen, at the beginning of the list of sparks for each topic, before displaying all other sparks.

5.1.2. Prototype requirements. As already mentioned in the section 5.1.1, a prototype is meant to focus mainly on features, so we only defined functional requirements for it.

We used our vision of the final app as basis to engineer the prototype requirements. The goal was to demonstrate as many of the most important features as possible. The following requirements resulted from our reflection:

1) Gamification concepts must be implemented in the application in form of a ranking system with points that can be gained through contributing to the crowdsourcing process in the app.
2) Unauthenticated users must be able to register or login.
3) Instead of having two different user status (”normal” and ”admin”), we chose to combine both user status into one which we will name “authenticated users”, to contrast with unauthenticated users. An authenticated user is therefore not representative of either roles, but it showcases the features of both of them.
    a) All important features of both normal and admin users must be implemented and available to authenticated users. These include creating and closing topics, posting sparks and comments, voting for sparks and liking comments, and viewing the profile.
    b) The admin feature of validating new user accounts is not a main feature and can therefore be left out.
    c) To showcase point 5) of section 5.1.1, authenticated users will appear by their username on topics and by their rank on sparks and comments.
4) The users must have some flexibility in the format when creating topics and sparks, such as the possibility to include headings, bullet points and code.
5) The user interface of the application must be intuitive and neat.
6) Point 6 of section 5.1.1 must only be implemented in the application if time allows for it.

5.2. Design

5.2.1. Gamification. The first part of the design was to define how gamification would be implemented. In the requirements, we mentioned the idea to use a ranking system, but we needed to make justified choices when it comes to the design of our application.

We wanted to consider the type of crowdsourcing system our application belongs to for justifying our chosen gamification implementation. In section 4.3, we have seen an introduction to the different types of crowdsourcing systems, and SmartSparks falls into the category of crowd solving (heterogeneous and non-emergent), which are, among others, very suitable for problems that have no objectively optimal solutions [3], such as those that can be addressed with SmartSparks. However, according to the survey article [6], no real pattern of gamification techniques can be identified to work best for one specific crowdsourcing type or another. They only conclude that a vast majority of systems use points, in conjunction with various other mechanisms such as leaderboards, time constraints etc. Our initial idea to use a ranking system with points in SmartSparks therefore seems to be a good idea, but we cannot provide more specific reasons than the general ones.

We implemented our ranking system in conjunction with votes for sparks and likes for comments to make use of the “wisdom of the crowd”. But to make the experience more interesting and encourage users to contribute on all levels, we constructed a slightly more advanced system of points and ranking.

Crowdsourcing requires two main skills from its participants: intuition and innovation. These two skills are incorporated in the name of the application as “smart” for intuition and “spark” for innovation. To encourage users to develop both skills while contributing, we made a category of points for each of the two skills: a “smart points” and a “spark points” category. Each user has one points score for each of the categories, and different actions will earn them points in one or the other category. Points are awarded following this schema:

- Smart points
    - Having voted for a spark that got picked: 20 points
    - Posting a comment: 10 points
    - Getting an own comment marked as useful: 50 points
    - Having liked a comment that was marked as useful: 10 points
- Spark points
    - Posting a spark: 10 points
    - Having posted a spark that got picked: 100 points

Since smart points are associated with intuition, the actions that can earn a user smart points are actions requiring mainly intuition, such as voting for the appropriate sparks and comments, and writing insightful comments on sparks.

On the other hand, spark points are awarded for actions that require innovation skills, which are posting qualitative sparks.

The exact amount of points for each action was determined without any specific criteria and would, in a real life application, need to be adjusted through testing.

Points are awarded either just after the action was performed (posting sparks or comments), or when an admin user performed an action enabling to establish the final points distribution (when an admin picks a spark during the closing process of a topic or when an admin marks a comment as useful).

The amount of points of a user determines its rank, an since there are two point categories, we opted for a rank composed of two parts. The amount of smart points will determine the first part of the rank:

- $[0, 10[$ points: "Novice"
- $[10, 200[$ points: "Active"
- $[200, 500[$ points: "Experienced"
- $[500, 1000[$ points: "Expert"
- $[1000, +\infty[$ points: "Smart"

While spark points give the second part of the rank:

- $[0, 10[$ points: "Reader"
- $[10, 200[$ points: "Contributor"
- $[200, 500[$ points: "Brainstormer"
- $[500, 1000[$ points: "Innovator"
- $[1000, +\infty[$ points: "Spark"

Therefore, a new user who has never contributed will start with a "Novice Reader" rank, while a user having more than 1000 points in both categories will have the rank "Smart Spark". A user who has 330 smart points and 190 spark points will have the rank "Experienced Contributor".

The requirements specify that users will appear by their rank on sparks and comments, to showcase the anonymity of normal users in the foreseen final application. It is important to note here that we chose to indicate the rank of the user at the time they post their content on the sparks and comments. Therefore, when a user gains a new rank, it is not updated on this user's sparks and comments. This way, the rank is always representative of the intuition and innovation levels of the user at the time they share their content.

5.2.2. Interface Design. The interface was first designed on paper through UI mockups, because it is fast to make and gives a good basis to work on when we arrive at the coding state. The two UI Mockup versions are shown in the appendix section.

The first version includes only the crowdsourcing features of the application, because the gamification implementation was not yet defined when this first version was drawn. The second version on the other hand also includes the main gamification design: votes, likes, a profile page with the points displayed and the selection of sparks when a topic is closed by an administrator. Both UI mockup versions have views for normal and admin users separated, as the decision to merge these two views into an "authenticated user" view was only made later, during the coding phase. However, the second mockup is already close to the final appearance of the application.

The final design of the interface can be seen on the screenshots presented in section 5.3.3.

5.3. Production

SmartSparks was developed using Visual Studio Code and tested in an emulator of the phone Nexus 6. This section will present the coding process and the final prototype obtained.

5.3.1. Frontend. The application was coded in the programming language Dart and using the Flutter framework for the frontend. Dart is an object-oriented and class-based programming language, which means a vast majority of the code is structured into classes. Flutter makes use of widgets which are structured in trees in the code. Therefore, the app has a root widget that is an ancestor of all other widgets. To be even more precise, every element of the application interface is a widget in Flutter, even things like colors. This allows for a very modular and reusable code, but results in loads of nested code.

Flutter has a number of built-in widgets for all sorts of basic elements (Text, Icon, Padding, GestureDetector, Container...) and some more elaborated ones as well (Scaffold, ListTile, AppBar...), which relieves the programmer from a lot of designing work. These basic widgets will be assembled into more elaborate ones to create the application we want to build.

Additionally to Flutter's built-in widgets, packages can be imported as dependencies in the project and provide additional, pre-made widgets. In our project, we used the packages "cloud_firestore", "firebase_auth" and "provider" for the implementation of backend communication and features. We also used "cupertino_icons", "flutter_svg", "syncfusion_flutter_gauges" and "flutter_spinkit" for some design elements, and finally "flutter_markdown" to implement markdown support for topics, sparks and comments.

Figure 2 shows the navigation between different screens of the application.

Only "wrapper" and "authenticate" are not screens, but parent widgets which are there to show either of their children widget, according to a boolean variable. This enables to toggle between authenticated and non-authenticated user view by displaying a different widget. The same mechanism is used to toggle between login and register screens. Therefore, all these connections are widget tree connections. A similar mechanism could also be used in the final version of the application to determine whether to show the admin or normal user view.

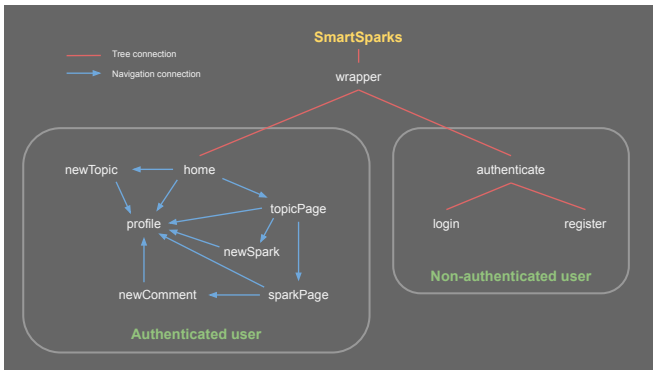


Figure 2: Graph showing widget tree and navigation graph.

Within the authenticated user view, the navigation between different screens was too complicated to implement with boolean variables. Furthermore, we wished to have a history of navigation, enabling to return to the previous page whenever we would navigate to a new page. Therefore, we used the built-in "Navigation" widget of Flutter to implement the navigation between these different screens. These are the blue arrows in the figure. Each blue arrow can be traversed in it's direction and back, but cannot be traversed in its opposite direction if it has not been traversed in it's normal direction first. For instance, navigating from home to topicPage to profile, then back to topicPage, then back to home is possible. But navigating from home to profile then to topicPage is not possible, because the arrow between topicPage and profile has not been traversed in its normal direction first.

5.3.2. Backend. The application is initially connected to firebase via the firebase console and is then able to communicate with the backend as long as it has the necessary permissions and an internet connection. Therefore, internet is required for the application to work.

In the case of SmartSparks, the backend is handling authentication and the database.

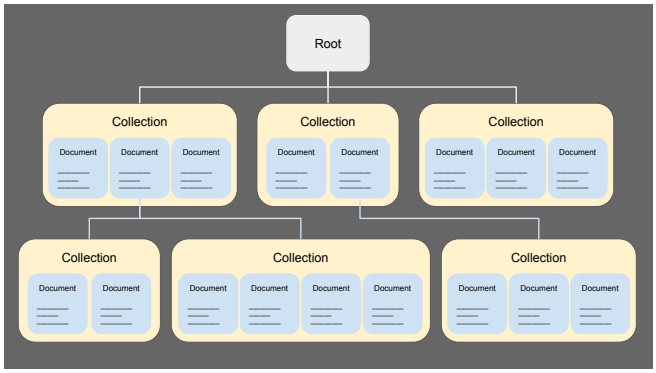


Figure 3: Cloud firestore data architecture.

The architecture of the database was constrained by the choice of backend for this project. Firebase provides two database services: Realtime Database and Cloud Firestore. We chose to use Cloud Firestore, because it is the most recent of the two and therefore better maintained. Furthermore, the data structure used to store the data is very convenient for our project.

Cloud Firestore imposes a particular type of NoSQL storage structure, where the data is stored in collections and documents, as illustrated in figure 3. A collection contains any number of documents. A document contains a set of key-value pairs, the values being the data stored and the unique keys a way to reference them. In addition to this data, documents can also contain any number of collections, which in turn may contain documents and so on. Collections and documents are references by unique IDs which can either be manually defined, or randomly generated.

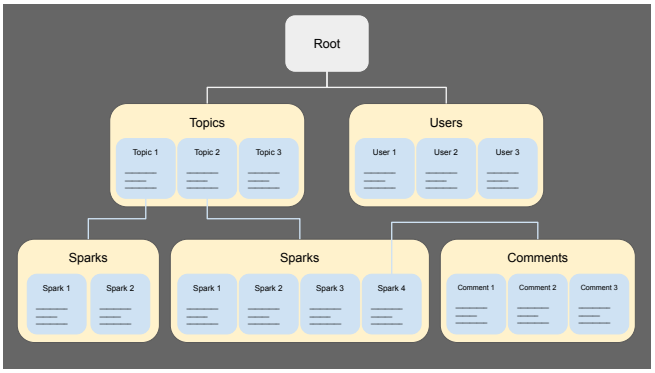


Figure 4: SmartSparks data architecture.

For our application, we used this nested structure to build an intuitive data storage structure, as depicted on figure 4: A collection at the root level contained all topics, each of them being a document. A topic document contained all data about the topic, such as the description and creator, as key-value pairs, as well as a collection of sparks documents. Similarly to the topics documents, each spark document contained the data of the sparks as key-value pairs and a collection of comments. The comments documents only contained key-value pairs, as they have no additional collection of objects associated.

At the root level, a second collection was also created to store the data of each user in a document. These documents have no nested collections, only key-value pairs.

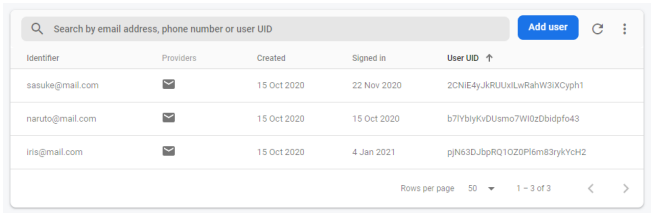


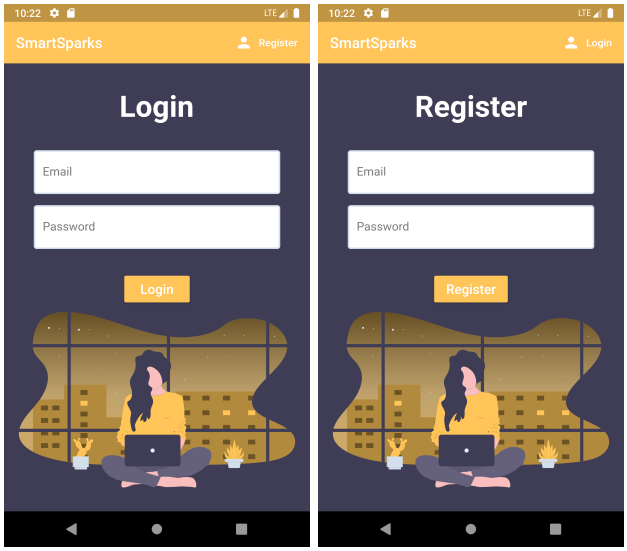
Figure 5: Firebase Auth list of users on the firebase console.

The authentication of users however is happening through Firebase Auth, another service provided by Firebase. The sign-in method enabled for SmartSparks is email and password, because it enables to identify each user (contrasting with anonymous authentication), but does not require any external account on any platform (the email must have a valid email format, but does not need to exist). Another valid choice would have been sign-in with phone number, but we were not sure how to use a phone number with the emulator, while we already knew how to use email and password, so we chose this method. Figure 5 shows the list of users, visible on the firebase console.

5.3.3. Prototype presentation. In this sections, we are presenting our prototype of the application SmartSparks. We will go through each different screen and explain the available features on each of them.

The first screen we see when opening the application is the login screen, displayed on figure 7a. It enables an unauthenticated user to log into the application with an existing account by entering their email and password and tapping the login button. In the top-right corner, a button “register” enables to transition to the register screen, displayed in figure 7b, where the user can create a new account with an unused email address and password.

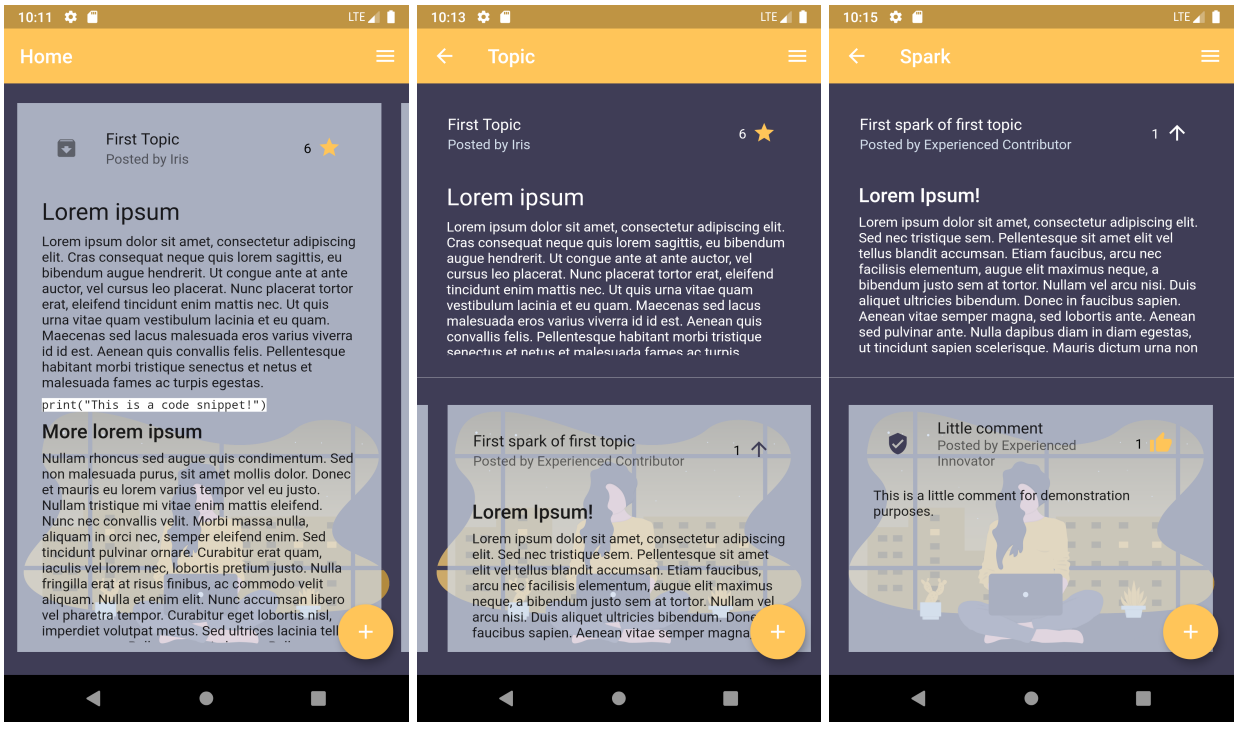
Once the user has logged in or registered, it is brought to the home screen of the application, shown in figure 6a.



(a) Login screen. (b) Register screen.

Figure 7: Unauthenticated user view.

This page shows a list of topics that the user can scroll through horizontally. A floating button in the bottom-right corner will lead the user to a page where they can create a new topic, which is shown in figure 9a. The archive button in the top-left corner of each topic tile leads to the close topic page, displayed in figure 8a. The list icon in the top-



(a) Home screen. (b) Topic screen. (c) Spark screen.

Figure 6: Authenticated user view, main screens.

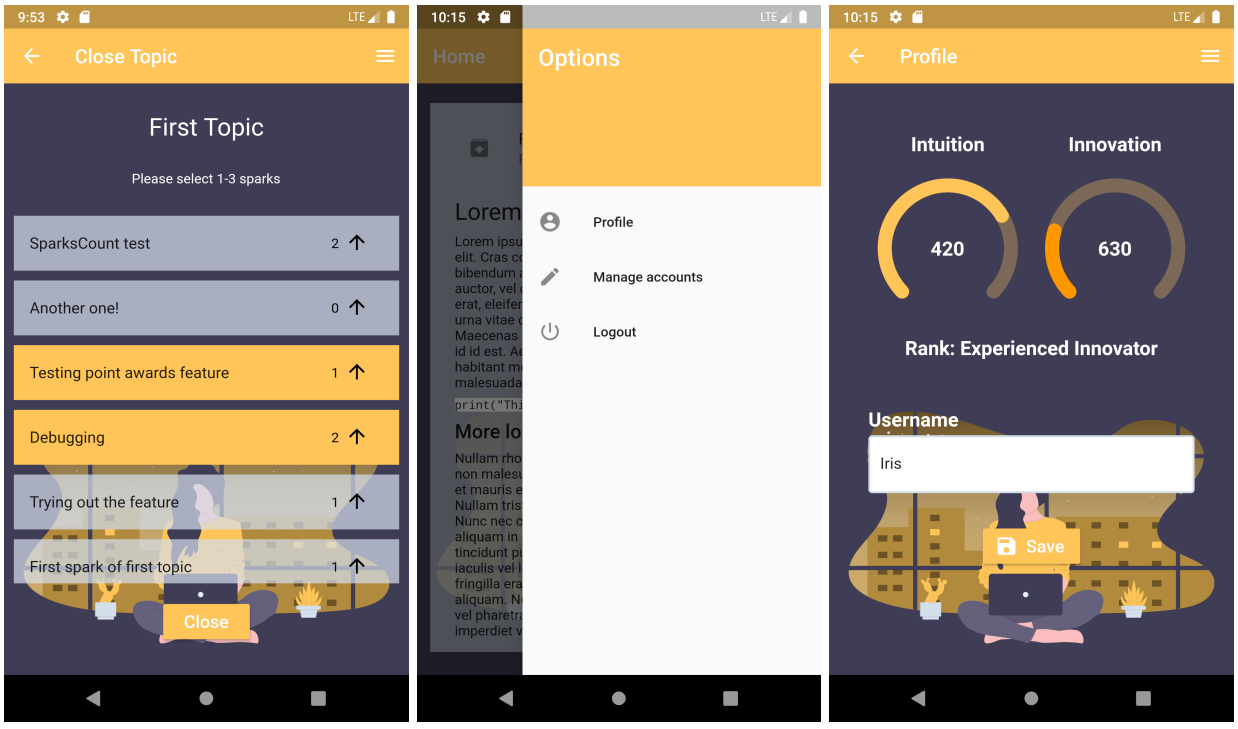
right corner is present on every screen of the authenticated view and leans to the option tab, in figure 8b. These three screens will all be presented later in this section.

The user can tap on a topic in the list to open the corresponding topic page. This screen is shown in figure 6b and shows the same topic in the upper half of the screen, while the sparks posted for this topic appear in a list on the second half of the screen. Similarly to the home screen, this page also has a floating button, leading to a page to create a new spark on this topic. It also has a back arrow, enabling the user to return to the previous screen, in this case the home screen. This arrow is automatically added by the Navigation widget of flutter, so it is guaranteed to always bring the user to the correct previous screen. Finally, each spark tile has an arrow pointing up in its top-right corner. If the arrow is yellow, the user has voted for this spark, otherwise it is gray. The user can vote for a spark by tapping the gray arrow and remove their vote by tapping on the yellow arrow.

On the topic page, the user can tap any spark tile to open the corresponding spark page, as in figure 6c. It shows the content of the spark in the upper half and the list of comments in the lower part of the screen. The back arrow leads to the previous page and the floating button enables to create a new comment on this spark. Just like on the topic page, the user can also vote for a spark directly from the spark page by tapping on the arrow pointing upwards. Similarly, the user can like a comment by tapping the thumb in the top right corner of the comment tile and remove their like with a second tap. The comment can also be marked as useful by tapping on the shield on the top-left of the comment tile. This action is final and cannot be undone. When a comment is marked as useful, points are automatically attributed to the concerned users, i.e. the creator and the people who liked it.

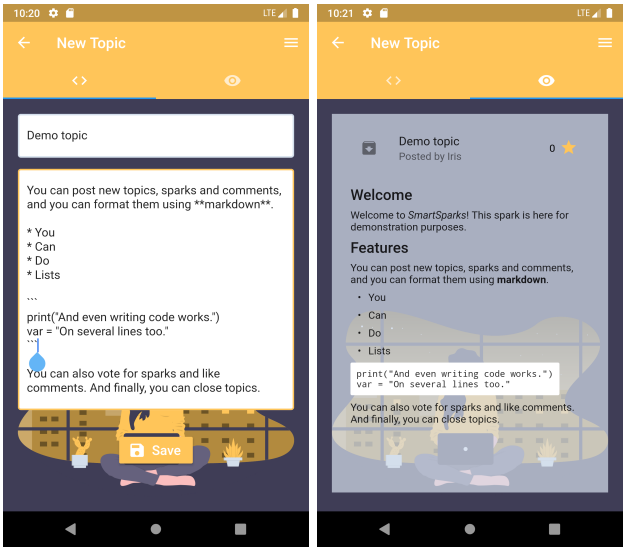
As already said, the floating button on the bottom right of each screen in figure 6 will lead the user to a screen where they can create respectively a new topic, spark or comment. Figure 9 shows the two different tabs on the new topic page, which are identical for the new spark and new comment pages. The user initially arrives on the screen of figure 9a, where they can enter a title and a body text. The application supports markdown to format the description text. If the user swipes to the right or taps the second tab, they will then see a preview of their new content, as shown in figure 9b. This way, they can see how their markdown formatting will appear in the end. Swiping to the left or tapping the left tab button will lead the user back to the edit screen. The arrow back will lead the user back to the previous page without saving the new information, while the save button will save the new topic, spark or comment, then lead the user back to the previous page, where they will also be able to see their newly created content. The user will also automatically be awarded corresponding points if they post a spark or comment.

From the home screen, the user can also click on the



(a) Close topic screen. (b) Options tab. (c) Profile screen.

Figure 8: Authenticated user view, last screens.



(a) New Topic screen - edit. (b) New Topic screen - preview.

Figure 9: Authenticated user view, new contributions screens.

archive icon of a topic to close it. This will lead them to the screen in figure 8a, where the list of sparks of this topic is displayed in a shortened version, with just the title and the number of votes. The user can select a few sparks which are the chosen solutions for this topic by tapping them in the list. The selected sparks will turn yellow to show that they are selected, and a second tap will deselect them. The user can return to the home page without saving their actions using the back arrow, or tap on the close button at the bottom of the screen to register their actions and return on home. If they tap close, the topic will not appear in the list of topics on the home screen anymore, and points will be awarded to the concerned users.

On every screen, there is a list icon in the top right corner, which will open the options tab, shown in figure 8b. The options available are “profile”, “manage accounts” and “logout”. The manage accounts item is only there for demonstration purposes, it will not lead anywhere in the implementation of the prototype. In a real application, this feature would be available to admin users and lead to a page where they can create and manage other admin accounts, as well as validate new normal user accounts. The logout item will obviously log out the user from the system on tap and lead them back to the login page, shown in picture 7a. Finally, the profile item leads the user on the profile screen, displayed in figure 8c.

The profile page shows the user’s current progress. Two point gauges indicate the number of smart and spark points respectively of the user, and the current rank of the user is also displayed. The gauge fills itself up, until the user reaches the next level (i.e. a new rank particle), then empties completely and starts filling up again for the next level. For each level, the color of the gauge filling changes: blue ($[0, 10[$), green ($[10, 200[$), yellow ($[200, 500[$), orange ($[500, 1000[$) and red ($[1000, +\infty[$). The user can also change their username on the profile page, by entering a new username in the corresponding field and tap on save, and in a final version of the application, changing the password would also happen here. And finally, of course, the user can return to the previous page without saving anything using the back button.

5.4. Assessment

5.4.1. Fulfillment of the Requirements. The requirements for the technical deliverable are presented in section 5.1, more specifically in section 5.1.2, as the purpose of section 5.1.1 was to establish the basis on which the requirements for the prototype would be defined. Therefore, the first part of the assessment is to determine whether these requirements have been fulfilled

The first requirement was fulfilled, as the gamification methods were implemented in form of a ranking system with points. Votes were also used in addition to them, and this does not contradict any requirement. Requirement 2 and all features encompassed in requirement 3 were also implemented and showcased in detail in section 5.3.3. Requirement 4 was fulfilled by implementing markdown support for the creation of topics, sparks and comments. We even implemented a preview feature to be sure that the users would have no problem formatting their content. Requirement 5 is difficult to reliably assess without testing, but we consider the interface to be rather intuitive and neat. As for requirement 6, the promotion of unpopular sparks was not implemented into the application due to lack of time, but we did not necessarily need to implement it, so this poses no problem.

Therefore, all requirements for the prototype were fulfilled.

5.4.2. Quality of the Application. In the development process of an application, a prototype is a very useful tool to assess the quality of the envisioned application and improve some flaws prior to the creation of the final product. Therefore, we used our prototype to detect potential issues and improvements to make in the final application. Please note that the prototype has not been tested at all, which makes it difficult to assess the quality of crowdsourcing and gamification features, so the critics are mainly conjectures.

Concerning the crowdsourcing features, we are convinced that the current features fulfill the main purpose of the application, but there is still room for improvement. A main critic to raise is the lack of interaction between the users of the application. The very limited comment feature is the only way different users of the application can communicate with each other. Furthermore, in the current state of the application, it is impossible to reply to

comments and have real discussions that could potentially lead to improvements of the sparks. So, implementing replies to comments has the potential to improve the crowdsourcing process a lot by introducing more user interaction.

As for the gamification features, we think that the points, ranks and voting system will indeed raise the motivation of people to participate in such an application. The potential for recognition by managers or other important people within the organisation will most likely also increase the investment of the users. However, improvements on this aspect can also be done. As already mentioned in section 5.2.1, the amount of points given for the different actions and the ranges for the levels would require testing to be optimized, as the currently used amounts were chosen purely arbitrarily. A leaderboard would be another possible improvement, because it would enable users to situate themselves with respect to the other users and motivate them to improve their score.

Acknowledgment

I, Iris Kremer, would like to thank my project academic tutor Prof. Dr. Steffen Rothkugel for his help and support throughout this semester project. I also want to thank my family for their love and support, and my friends for inspiring some of my ideas in this project.

Finally, I want to thank the BiCS management team for establishing the bachelor semester projects.

6. Conclusion

This paper presented the fifth bachelor semester project made by Iris Kremer, under the direction of her project academic tutor Prof. Dr. Steffen Rothkugel. This project produced one scientific and one technical deliverable. The scientific deliverable of this project focussed on crowdsourcing, gamification and how to use gamification in crowdsourcing. The technical deliverable was the prototype of the application SmartSparks, a gamified crowdsourcing application for organisations. Both these deliverable fulfilled their requirements, so the project is a success, but these deliverables could of course be improved further in the future.

In the scientific deliverable, we arrived at the state where we can affirm that gamification benefits crowdsourcing, but do not know how much exactly gamification impacts crowdsourcing in its process and user experience. A possible improvement would therefore be to go further and analyse by how much gamification in crowdsourcing improves the outcomes of the process and the experience of the crowd with the system used.

The technical deliverable is a prototype, and a prototype is not meant to be kept and turned into a final application. This means that it does not need further improvement in itself. The interesting outcomes to consider here would rather be how the prototype helps to improve our projects for the final application. We described our initial view of the final application in section 5.1.1, and some aspects to improve that we identified with the prototype in section 5.4.2. In the future, the final application would therefore need to take into account the points made in section 5.4.2 when defining the requirements for this final product.

References

[1] Luis von Ahn. Human computation (phd thesis). Technical report, CMUCS-05-193, December, 2005.

[2] Kai Huotari and Juho Hamari. Defining gamification - a service marketing perspective. 10 2012.

[3] David Geiger, Michael Rosemann, Erwin Fielt, and Martin Schader. Crowdsourcing information systems-definition typology, and design. 2012.

[4] Cambridge dictionary. Definition of crowd. https://dictionary.cambridge.org/dictionary/english/crowd. Online; accessed 12th of December 2020.

[5] Cambridge dictionary. Definition of source. https://dictionary.cambridge.org/dictionary/english/source. Online; accessed 12th of December 2020.

[6] Benedikt Morschheuser, Juho Hamari, Jonna Koivisto, and Alexander Maedche. Gamified crowdsourcing: Conceptualization, literature review, and future agenda. *International Journal of Human-Computer Studies*, 106:26–43, 2017.

[7] Ivo Blohm, Shkodran Zogaj, Ulrich Bretschneider, and Jan Marco Leimeister. How to manage crowdsourcing platforms effectively? *California Management Review*, 60(2):122–149, 2018.

[8] eYeka. The state of crowdsourcing in 2015. https://eyeka.pr.co/99215-eyeka-releases-the-state-of-crowdsourcing-in-2015-trend-report. Online; accessed 13th of December 2020.

[9] Rainer Mühlhoff. Human-aided artificial intelligence: Or, how to run large computations in human brains? toward a media sociology of machine learning. *New Media & Society*, 22(10):1868–1884, 2020.

[10] TEDx Talks. The future of creativity and innovation is gamification: Gabe zichermann at tedxvilnius. https://youtu.be/ZZvRw71Slew. Online; accessed 3rd of January 2021.

[11] Karen Robson, Kirk Plangger, Jan H Kietzmann, Ian McCarthy, and Leyland Pitt. Is it all a game? understanding the principles of gamification. *Business horizons*, 58(4):411–420, 2015.

[12] Michael Sailer, Jan Hense, J Mandl, and Markus Klevers. Psychological perspectives on motivation through gamification. *Interaction Design and Architecture Journal*, (19):28–37, 2014.

[13] L. von Ahn. Games with a purpose. *Computer*, 39(6):92–94, 2006.

[14] Alfonso González Briones, Pablo Chamoso, Alberto Rivas, Sara Rodríguez, Fernando De La Prieta, Javier Prieto, and Juan M Corchado. Use of gamification techniques to encourage garbage recycling. a smart city approach. In *International Conference on Knowledge Management in Organizations*, pages 674–685. Springer, 2018.

[15] Gabriela Kiryakova, Nadezhda Angelova, and Lina Yordanova. Gamification in education. Proceedings of 9th International Balkan Education and Science Conference, 2014.

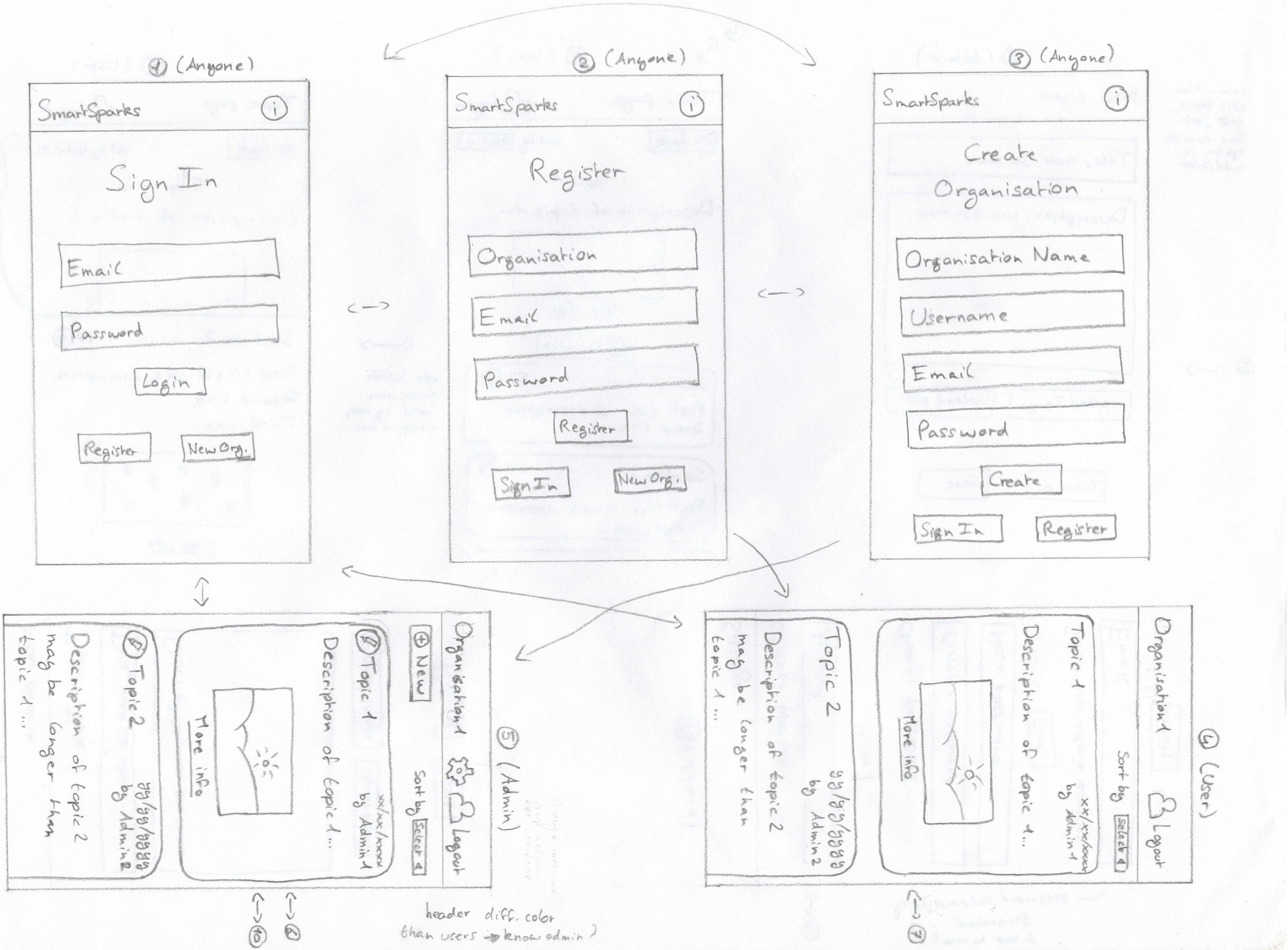
[16] Maciej Laskowski, Marcin Badurowicz, et al. Gamification in higher education: a case study. In *Make Learn International Conference*, volume 25, pages 971–975, 2014.

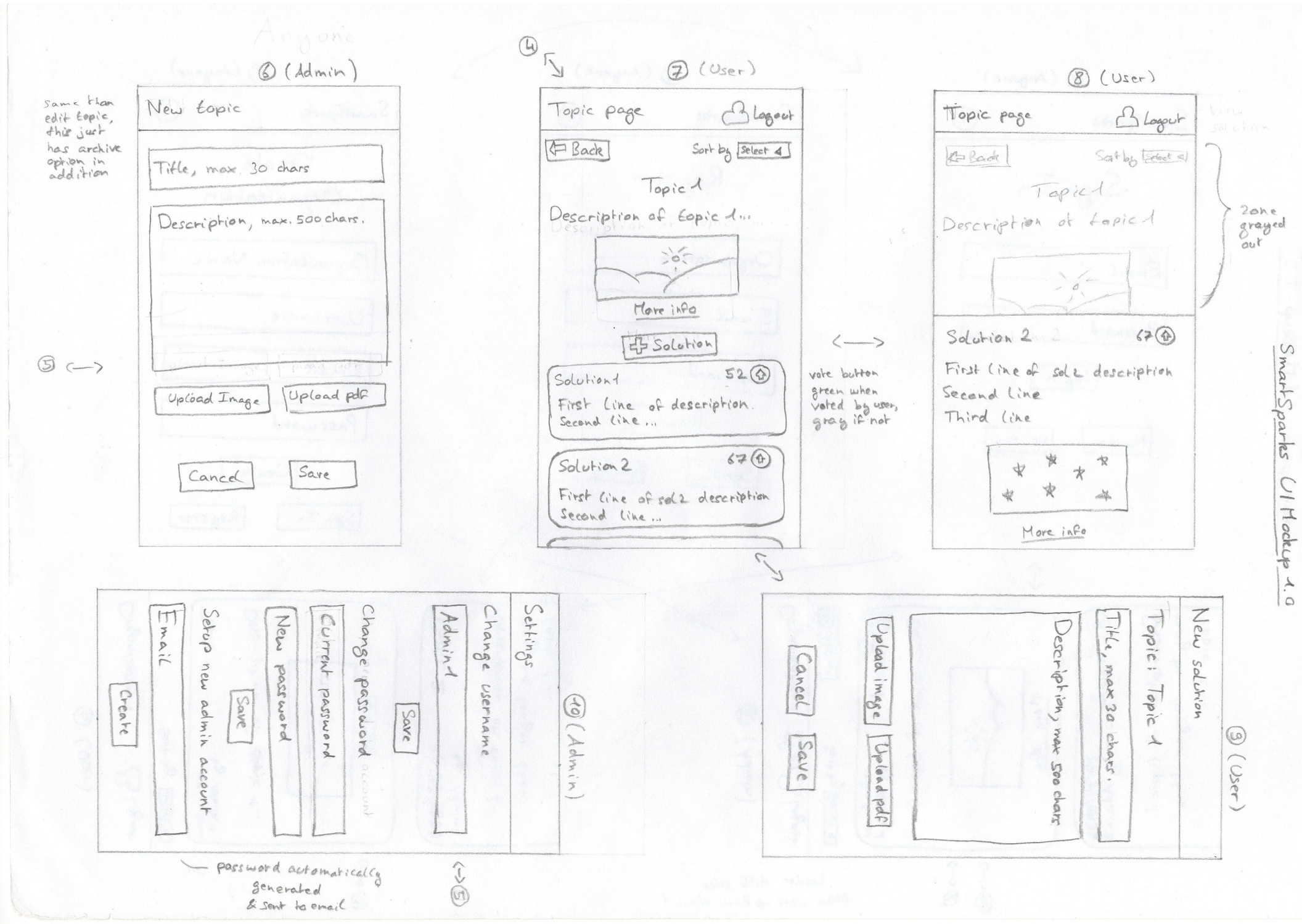
[17] Seth Cooper, Firas Khatib, Adrien Treuille, Janos Barbero, Jeehyung Lee, Michael Beenen, Andrew Leaver-Fay, David Baker, Zoran Popović, et al. Predicting protein structures with a multiplayer online game. *Nature*, 466(7307):756–760, 2010.

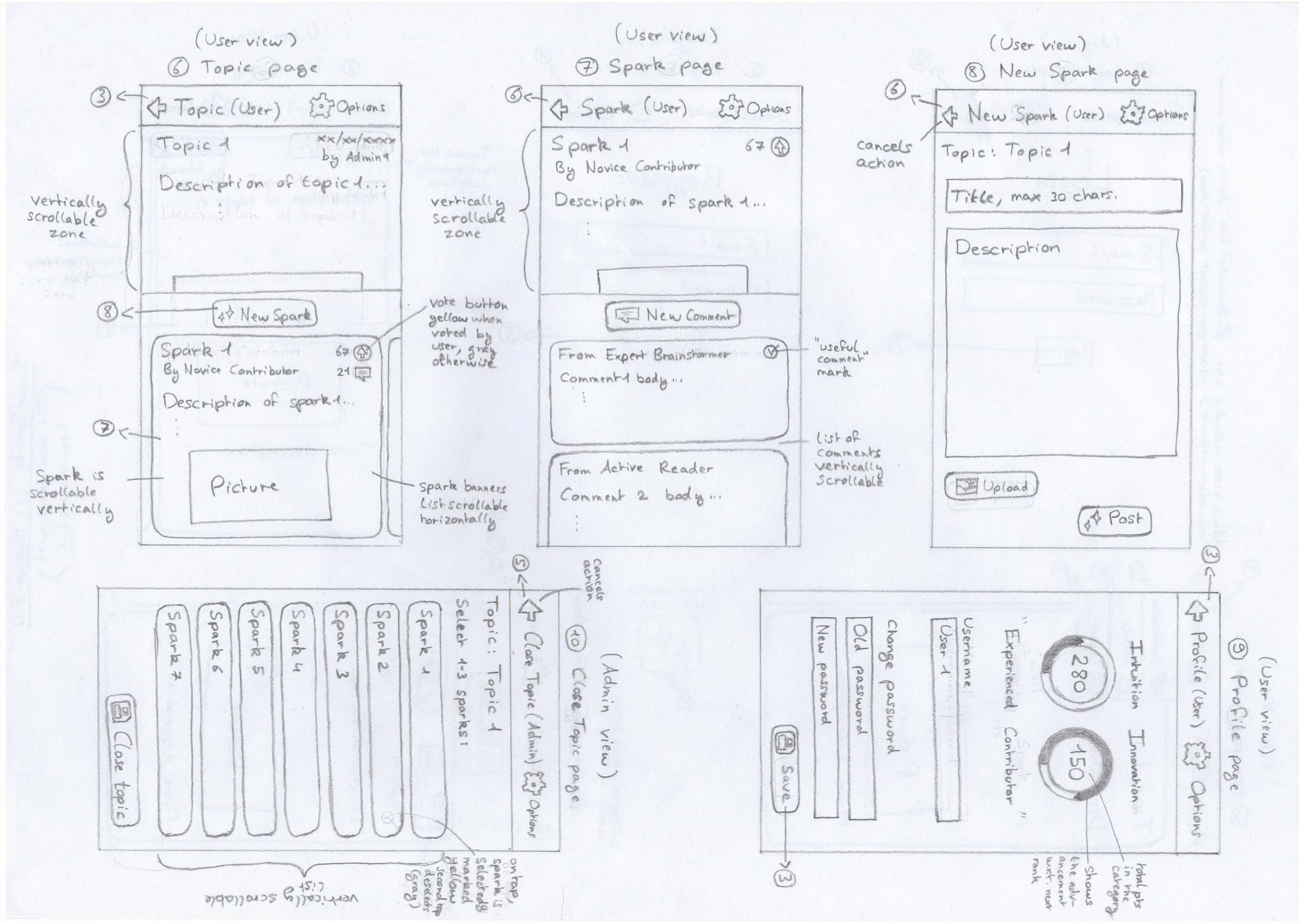
[18] Firas Khatib, Frank DiMaio, Seth Cooper, Maciej Kazmierczyk, Miroslaw Gilski, Szymon Krzywda, Helena Zabranska, Iva Pichova, James Thompson, Zoran Popović, et al. Crystal structure of a monomeric retroviral protease solved by protein folding game players. *Nature structural & molecular biology*, 18(10):1175–1177, 2011.

7. Appendix

SmartSparks UI Mockup 1.0




SmartSparks UI Mockup 1.0
⑥ (Admin)
same than edit topic, this just has archive option in addition
New topic
Title, max. 30 chars
Description, max. 500 chars.
Upload Image
Upload pdf
Cancel
Save
⑤
④
⑦ (User)
Topic page
Logout
Back
Sort by
Select
Topic 1
Description of topic 1...
More info
Solution
Solution 1
52
First line of description.
Second line...
Solution 2
67
First line of sol2 description
Second line...
vote button green when voted by user, gray if not
⑧ (User)
Topic page
Logout
Back
Sort by
Select
Topic 1
Description of topic 1
2one grayed out
Solution 2
67
First line of sol2 description
Second line
Third line
More info
⑨ (User)
New solution
Topic: Topic 1
Title, max 30 chars.
Description, max 500 chars
Upload image
Upload pdf
Cancel
Save
⑩ (Admin)
Settings
Change username
Admin1
Save
Change password
Current password
New password
Save
Setup new admin account
Email
Create
password automatically generated & sent to email
⑤


(User view)
⑥ Topic page
③
Topic (User)
Options
Topic 1
xx/xx/xxxx
by Admin1
Description of topic 1...
vertically scrollable zone
⑧
New Spark
Spark 1
67
By Novice Contributor
21
Description of spark 1...
⑦
Picture
Spark is scrollable vertically
vote button yellow when voted by user, gray otherwise
spark banners list scrollable horizontally
(User view)
⑦ Spark page
⑥
Spark (User)
Options
Spark 1
67
By Novice Contributor
Description of spark 1...
vertically scrollable zone
New Comment
From Expert Brainstormer
Comment 1 body...
"useful" comment mark
List of comments vertically scrollable
From Active Reader
Comment 2 body...
(User view)
⑧ New Spark page
⑥
New Spark (User)
Options
cancels action
Topic: Topic 1
Title, max 30 chars.
Description
Upload
Post
⑤
Close Topic (Admin)
Options
cancels action
(Admin view)
⑩ Close Topic page
Topic: Topic 1
Select 1-3 sparks!
Spark 1
Spark 2
Spark 3
Spark 4
Spark 5
Spark 6
Spark 7
Close topic
vertically scrollable list
on tap, spark is selected & marked yellow
second tap deselects (gray)
③
Profile (User)
Options
(User view)
③ Profile page
Intuition
Innovation
280
150
"Experienced Contributor"
shows the advancement wrt. next rank
total pts in the category
Username
User 1
Change password
Old password
New password
Save
③

(User View)

④ Homepage - options

Home (User) | Options

Switch view ⑤

Profile ⑤

Logout ①

Topic 1

Description of

Picture

More description...

③

Admin view includes also "Accounts" for "delete admin account", "new admin account" & "activate user account" options (demo)

(Anyone)

① Login screen

SmartSparks | ⓘ | Register ②

Login

Email

Password

Login ③

② Register screen

SmartSparks | ⓘ | Login ①

Register

Email

Password

Register ③

SmartSparks app icon ①

indicates which view user is currently in

⑤ Homepage (Admin View)

Home (Admin) | Options ③ (⑤, ①)

New Topic

Close ⑩ | Edit

Topic 1

xx/xx/xxxx by Admin1

Description of topic 1...

Picture

More description...

(User View)

③ Homepage

Home (User) | Options ④

Topics list horizontally scrollable

Topic 1

Description of topic 1...

xx/xx/xxxx by Admin1

Picture

More description...

⑥

individual topics vertically scrollable

SmartSparks UI Hackup 2.0

(In accordance with Requirements 2.1)