Teaching critical thinking through argument mapping

TI3806

Felix Akkermans
Robin Borst
Boudewijn van Groos
Jens Langerak

TU Delft

Teaching critical thinking through argument mapping

TI3806

by

Felix Akkermans
Robin Borst
Boudewijn van Groos
Jens Langerak

to obtain the degree of Bachelor of Science
at the Delft University of Technology,

Project duration: April 24, 2017 – July 4. 2017

TU Delft

Preface

This report concludes the Computer Science bachelor project of Delft University of Technology. This report aims to provide the reader information of the research, design, implementation and testing work done over the span of 10 weeks at the company FeedbackFruits. The project client, FeedbackFruits, is an education technology developer aimed at teachers and students from high school to university. By exploring various products it hopes to find ways to improve the learning experience of students and promote the cultivation of key cognitive traits like critical thinking. During our project we have introduced a solid basis for a new product that aims to promote critical thinking through argument-mapping-based debates. This report also contains recommendations on future work to develop this basis to a mature product.

We want to express our gratitude towards our coach Catholijn Jonker for her guidance, Joost Verdoorn as the company supervisor and all the employees supporting us during the process with feedback, testing and technical support.

Felix Akkermans
Robin Borst
Boudewijn van Groos
Jens Langerak
Delft, June 2017

Summary

During this project we have created a product for the company FeedbackFruits to promote critical thinking in education. Research has shown that argument mapping is a promising method to promote critical thinking. Several existing related platforms were analyzed to draw lessons from their experience. Literature research revealed that product interaction design is very important for teacher adoption and student engagement, which were are also identified as key factors for product success by the client company. We took steps to realize this requirement by researching student engagement and adopting company guidelines used to ensure quality design (e.g. use of company UI framework, internal design review, user testing). User tests indicated enthusiasm and high engagement by the participants, but also showed clear areas of improvement. Though comparable existing solutions are promising for teaching critical thinking, further development and testing would be needed to validate the didactic potential of this product. Considering the limited time available for the project, we have included this in the recommendations on future development and research.

Contents

1 Introduction

Recently, the 2016 US presidential elections and Brexit referendum left many in a state of shock. We witnessed a rising trend of misinformation and group polarization, visibly effecting the belief forming and political decision making of citizens and politicians alike. While our technological power is increasing rapidly, evidently our collective ability to self-correct and coordinate ourselves in time is not following equally. Consider climate change as an example of the increasingly dramatic consequences of our technological nature that calls on our ability to adapt in time.

Science advocates are recently often raising alarm on policymakers rejecting overwhelming scientific evidence on the grounds of mischaracterizing science as a partisan issue and hence "just another opinion". If we neglect this trend we might find ourselves in a situation of epistemic relativism, unable to agree upon basic questions of reality, even in the face of pressing global challenges to the safety of society. Valuable ideas and reasons are lost amidst the rage of group polarization. Simple scientific facts can be pressured out of common knowledge by misinformation. With these phenomena being increasingly amplified through the internet and social media, we consider them becoming an immediate threat to the functioning of our democracy and with that to society.

There has been a lot of attention in the media and public discourse on highly political phenomena like *fake news*, *hoaxes*, *alternative facts*, *filter bubbles* and *ideological echochambers*. Perhaps one of the most pivotal terms is *post-truth politics*, selected as word of the year 2016 by the Oxford dictionary [1] and described as:

> "Relating to or denoting circumstances in which objective facts are less influential in shaping public opinion than appeals to emotion and personal belief."

In the pursuit of understanding this problem, we are obliged to study the human condition, to discover the underlying mechanisms that drive this phenomena. Social psychology, cognitive psychology and behavioral economics provide ample of descriptive theory here. We have a baggage of cognitive dispositions like confirmation bias [2], motivated reasoning [3, 4], tribalism [5, 6] (e.g. in-group bias), self-interest [5] (e.g. tragedy of the commons) and short-term thinking [7] (e.g. hyperbolic discounting).

If we wish to be reasonable and morally consistent, then these leave us vulnerable in our decision making and belief forming. The resulting gullibility to divisive and manipulative rhetoric is but one of the dangers that can be exploited to mislead us.

Hence, we argue that developing several cognitive traits - of which critical thinking is integral - might be key to our ability as a society to converge upon accurate beliefs, find and enact the needed solutions to deal with the pressing and highly complex problems we're facing today. As it were to cognitively "vaccinate" us to the dangers that lurk in our human condition.

In this bachelor project we explore a method of teaching critical thinking. We do this in the environment of the educational tech company FeedbackFruits, dedicated to improving education. Founded in 2012, FeedbackFruits has been exploring various ed-tech solutions. These aim to promote the easy adoption of didactic solutions by teachers, increase student engagement and cultivate key cognitive traits like critical thinking. Because teacher adoption and student engagement have already been identified as the dominant challenges in product success by FeedbackFruits, we dedicate significant effort to ensuring our method also succeeds

here. The difference with typical bachelor projects is that this challenge implies excellence in user experience and interaction design during our process. Hence, we spend careful attention on research, design and validation of these aspects. More can be read about this in our research and design phase (see section 2, 3 and 4.6).

2
Research

The first two weeks of the project are dedicated to research. In section 2.1, we look at the problem we face and how we analyzed it. We identify the key challenges and investigate what literature says about them. In section 2.2, we discuss the frameworks we considered for this project. We end with listing the design goals in section 2.3 and the conclusions in section 2.4.

2.1. Problem definition and analysis

In this section, we investigate the problem for which we have to create a solution. The first two sections, 2.1.1 and 2.1.2, analyze what FeedbackFruits expects from this project. In section 2.1.3, we discuss what can be learned from the literature about teaching critical thinking. Furthermore, in section 2.1.4, we investigate how we can improve the student engagement. To get a better idea of the challenges for this project, we study existing efforts related to argument mapping in section 2.1.5.

2.1.1. Problem definition

FeedbackFruits sees the need for more focus on critical thinking in education. It hopes that providing a pedagogical solution will stimulate that focus and increase its effectiveness.

2.1.2. Problem analysis

FeedbackFruits wishes to have a product fulfilling the need for more focus on critical thinking in education. We hope to design, implement and deploy this product during the length of this project. The success of this product will depend on two factors: teacher adoption and student engagement. Teacher adoption in turn depends on the didactic value and ease of use. Only if teachers can be convinced of both the didactic value and if they are willing to invest the time to apply it, only then the application will be used during courses. Using the application will not automatically improve the critical thinking capabilities of the student. To achieve that we have to motivate the students to use our application in a serious way. This is why we also need to stimulate student engagement.

2.1.3. Teaching critical thinking

In this section we evaluate some of the literature on teaching critical thinking, consider various methods used to achieve this and formulate a didactic solution based on these findings. Before we get to these methods, we should start first with a definition of critical thinking.

About critical thinking

A necessary component of the ability to think critically about a claim, policy or decision is the ability to assess the evidence or reasons which might count for or against it. For that reason, the ability (and disposition) to analyze, construct and evaluate arguments is often considered to be a core component of critical thinking [8]. For a detailed analysis of the skills and dispositions commonly supposed to be involved in critical thinking, see the paper of Ennis [9].

Despite critical thinking having long been a central tenet of educational policy [10], the literature shows a confronting reality that in the end, results are far from what was hoped for [11].

Quality practice hypothesis
One view on improving the teaching of critical thinking is represented by the quality practice hypothesis [12–14]. According to this theory, acquiring expertise in critical thinking, as in other areas, requires large amounts of deliberate practice. The concept of deliberate practice is based on research in cognitive science on how expertise is acquired in a variety of cognitive domains [15]. According to this hypothesis, deliberate practice has the following properties.

Motivated
Motivation should be intrinsic. The students should be deliberately practicing in order to improve their skills, instead of engaging without a sense of purpose.

Guided
The students should have access to help about what to do next.

Scaffolded
In the early stages, it should be impossible for the students to make certain kinds of mistake.

Graduated
Exercises gradually increase in difficulty and complexity.

Sufficient feedback
Students should have some way of knowing whether they are doing the right thing or not.

The quality practice hypothesis states that critical thinking practice must be deliberate to have any chance of substantially improving students' performance [13].

Argument mapping
One of the most important aspects of critical thinking is analyzing and synthesizing arguments. Arguments, especially good arguments, form a hierarchical structure, which can be organized in diagrams. By drawing maps, the structure of arguments can be visualized. Cognitive science has shown that the critical thinking skills of students improve faster when argument mapping is used. According to Van Gelder [14], argument maps have several advantages over traditional teaching methods. Firstly, they make reasoning easier to understand. Secondly, students can easily identify important issues, such as when an objection was made. Thirdly, it is easier to "follow extended critical-thinking procedures" [14]. Finally, it allows teachers to easily identify arguments and give focused feedback.

Debate
It has been shown that debate is a good option for educating students [16, 17]. It also transfers well to an online environment [18]. While it is often the case in a traditional debate that the amount of participants is very limited, this can be solved by assigning roles to the various players [19].

The benefits of an online debate are substantial, as it does not require simultaneous presence and it allows for better research, due to its asynchronous nature and does not break up any normal lecture schedule. We hypothesize that a debate can also be held in the form of an argument mapping, thereby increasing its clarity and making participation easier. The resulting argument mapping will provide the benefit of a clear end result of the debate.

One key factor in a debate is the format used. Assigning roles tends to give more balanced debates and extending those roles beyond supporting or rebutting a certain topic increases the participation [16]. It also allows for a more compromise oriented debate, thus preventing it reinforcing dualism, instead of combating it.

Peer instruction
Another noteworthy method of teaching critical thinking is peer instruction. Peer instruction is a modification of the traditional lecture format to make the lectures more interactive, more engaging and more effective.

At various points during the lecture, the instructor stops and asks a multiple choice question that all students must respond to using flash cards or an audience response system. If most students have the right answer, the lecturer confirms it and moves on. If there is a mixture of right and wrong answers, the lecturer asks the students to spend 1-2 minutes trying to convince a neighbor that their answer is correct. Students

then 're-vote' on the answer. Typically, more students have the right answer after these peer discussions; students who have just mastered the material are able to explain it effectively to their peers.

In Butchart's study of argument mapping, a comparison was made with other methods of teaching critical thinking [8]. Peer instruction was the only other teaching method next to argument mapping to have a statistically significant effect, where the gain was 0.4 standard deviations compared to 0.45 for Butchart's argument mapping method (significant at the 0.05 level).

Peer feedback

The availability of feedback is an integral aspect of teaching critical thinking [11, 13]. With large classrooms, this becomes an obstacle for the teacher in terms of workload. Peer feedback is a commonly used format where students review work from each other. This significantly helps to reduce the workload for teachers, but more importantly, it is hypothesized to impact students' learning at a higher cognitive level, such as their critical thinking skills [20].

In this study, students gave peer feedback in the form of a numerical score (from 0 - 2) based on Bloom's taxonomy and descriptive comments supporting the assigned score. Postings demonstrating analysis, synthesis, or evaluation received 2 points; postings at the knowledge, comprehension and application levels received 1 point. Non-substantive comments received 0 points.

Conclusion

Because argument mapping by its structure closely reflects some of the aspects of critical thinking, we consider this method to have potential to teaching critical thinking. For this reason, we employ it as the leading solution hypothesis to test.

We also consider leveraging a debate format to assign roles to students to increase student engagement and encourage focus on argumentation. In addition, leveraging peer feedback is a promising way of reducing teacher workload in providing quality feedback and potentially aids in engaging higher cognitive functions like critical thinking in the feedback writing phase. We consider these as extensions of argument mapping to increase its potential value as a didactic solution.

2.1.4. Student engagement

A challenging task in developing teaching methods is how to keep the students engaged in the course. Before we explain the different strategies used to improve the engagement, we define engagement. According to Fredricks *et al.* [21], there are three dimensions of engagement: behavioral, emotional and cognitive engagement. Behavioral engagement means that engaged students behave engaged, for example by attending lectures and asking questions. The emotional aspect of engagement is about the interest and enjoyment for courses or assignments. Cognitively engaged students are usually eager to learn or do more than is required. These three dimensions should all be considered in the development of the argument mapping tool.

Engagement factors

Concrete engagement factors are discussed by Kuh [22] and Coates [23], these factors are benchmarked in more detail by Robinson and Hullinger [24]. The first factor is *level of academic challenge*, which measures the amount of effort students put into studying, reading or writing. For trying to measure this challenge, five mental activities were mentioned, namely 1) memorizing facts or ideas, 2) analyzing ideas, 3) organizing ideas, 4) judging the value of information or arguments 5) applying certain theories [24]. Stimulating these mental activities will help students to develop better skills to think critically.

The next factor of engagement is *student-faculty interaction*, which is the ability for student to stay in contact with the instructor. Student are more engaged if they receive prompt feedback on the assignments they make [24]. Discussing grades or assignments was also shown to be important for students in order to be more involved.

Active and collaborative learning refers to the involvement of students in group discussions or other class activities [22]. Critical thinking is stimulated if students work together in groups, whether the groups are online or not. A good example is peer tutoring or peer instruction, which was already explained before.

Finally, *enriching educational experience* is learning to collaborate with students from different backgrounds by using technology [22]. Nowadays, this engagement factor has become less important, since students are used to using information technology in courses.

Existing examples
Now we have seen the theoretical factors of student engagements, let us have a look at some existing examples. Recently, Hew [25] investigated how three top-rated online courses dealt with engaging the students. This resulted in five factors that explain why these courses work well. For each of these factors, we highlight the examples that could be useful for our product.

First, we look at problem-oriented and active learning, which is about analyzing the theory and learning to apply it in practice. For analyzing, it is useful to have clear examples of good and wrong argument constructions. In order to learn to formulate correct arguments, a walk-through or template could be used. This way, the student is actively involved in the learning process. Another way of active learning is the use of (machine-graded) quizzes, so that students immediately know if they encounter improvements in their performance. Giving feedback on work of fellow students is also helpful in learning to think critically. Such a peer review can spark new ideas and perspectives that a student can use for his own work. After having done evaluations for other students, one can evaluate his own work, which is called self-assessment.

Only learning from assignments is not enough to keep students engaged. Social interaction with both the instructor and the other students is one of the key elements to think critically. One example is the possibility to create a link to your work, so that you can contact the instructor if you have a question. Instead of such individual questions and feedback, interactive discussions with the instructor and the students can also be useful. To communicate with fellow students more directly, sub-forums can be used to discuss feedback given in peer reviews for example.

Finally, using helpful course resources could add to addressing the learning needs of students. Providing access to further readings could be useful for students who are eager to learn more. However, we doubt that this will provide better student engagement. Instead, the use of a competition or Hall of Fame can be really effective to keep students more engaged.

To conclude, Hew [25] showed that the previously described engagement factors contribute to the behavioral, emotional and cognitive engagement of students in online courses. With this insight and examples, we can derive requirements for the design and implementation of our tool. Important to keep in mind: the type of assignment is not the most important in order to keep students engaged, but "It's all about the connections" [26].

Deriving requirements for engagement
For reaching a sufficient level of academic challenge, we can conclude that organizing and judging arguments should be possible with our application. As already stated in section 2.1.3, organizing the arguments can be done by using an argument mapping structure. Judging the arguments can be made possible by allowing comments or discussions on arguments. We also saw that interaction with students and instructors should be possible. Communicating with the instructor is already possible with the chat function built into FeedbackFruits. By using comments on arguments, communicating with other students is possible, as well as with the instructor. However, this might not result in students actively collaborating, because students can still be isolated. Assigning roles to students, as concluded in section 2.1.3, should increase this participation rate. However, even with roles assigned, students are still likely to do only as much as is required, so students should be motivated to do more than required. One way of doing this is providing a list with user statistics that can be sorted in some way so that the best students stand out.

2.1.5. Existing solutions
Building on the premise of developing argument mapping, it is worth studying existing efforts and efforts related to argument mapping. Deliberation mapping is very closely related in structure to argument mapping and heavily relies on critical thinking of the participants in order to come to an optimal judgment on proposals. Because of this, challenges and solutions encountered in this domain might have relevance to our design decisions. Here we highlight some conclusions and take-away lessons from (or reported in) the literature of existing solutions and efforts.

Deliberatorium
Deliberatorium was created by MIT with the goal of bringing order to online interaction. Deliberatorium has the goal of discussing an issue in an organized form and settling on good ideas through cooperation.

In such a deliberation map contributions are split in issues, ideas and arguments. An issue is the problem to be solved, an idea is a way to approach that issue and an argument can make a case in favor or against a

certain idea. Each idea can again have more underlying issues, which then again can have ideas and those can have arguments and so on. Deliberatorium presents all this information in the form of a nested vertical list. While this allows for communicating a lot of information with little screen space, it might become a little disorienting for the user in large deliberation maps.

Their research has shown that the platform works well for groups in the size of hundreds with a high participation rate. In one case it generates roughly 3000 items in the deliberation map. This scale would be unfeasible for more traditional forms of collaboration [27]. The high participation rate with large groups makes it show very promising results.

Deliberatorium still has some drawbacks. The authors found the direction of attention and moderation was one to be improved on. While the overview of the discussion was very beneficial to the discussion, attention would often be lost in a small part of the map. Another issue was moderation: most users were not familiar with argument mapping and thus moderators were needed to correctly structure the conversation.

We think Deliberatorium lacks in terms of user experience. We also doubt a new user can quickly become accustomed to the structure used, as it may not allow for more natural communication. While the structure used by Deliberatorium is a logical one from an academic perspective, it may not be the case for users. With an experienced user it can be used very well for deliberation. We think its structure and user experience make it less suitable for a debate.

MOOD

The massive open online deliberation (MOOD) [28] platform is developed by the TU Delft. MOOD tries to take both the majority and the minority into account. According to MOOD, this is necessary if you want to be ethically just. They do this by differentiating between the debate outcome and the moral acceptability of the outcome.

MOOD consist of four steps. First, an open debate question is proposed. In the second step, people can add alternatives to the debate. They can add pro and con arguments to the alternatives. Arguments needs to be supported by facts and sources. People can vote for the alternatives and the most popular alternatives enter the third stage. In the third state, people have to rank the alternatives and for each alternative they have to answer a survey about the moral acceptability. In the fourth and final state, the result of the debate is shown. It shows both the social acceptance as the moral acceptability of the alternatives.

A strong point of MOOD is that it acknowledges that not only facts are important in a debate, but values as well. MOOD does this by measuring the moral acceptability. Introducing this distinction helps to differentiate arguments on their emotional appeal and moral merit. Personal preference of a proposal is influenced by emotions, sentiment, perceived benefits and risks, and social pressures and interactions. Allowing users to express these influences can play a vital role in user engagement.

Moral acceptability is stressed as an important and sometimes conflicting aspect to considering proposals. Using principles such as harm, fairness, liberty, loyalty and authority, the moral aspects of arguments can be considered. These moral implications lead us to regulate and suppress selfishness and make social life possible [29, 30].

Looking at the MOOD platform we have to conclude that it leaves a lot to be desired in interaction design and user experience. Furthermore, we also miss the ability to react on arguments. As far as we know, you can only react on the alternatives, but not on the arguments. Although they have made it possible to correct false claims, we think that for a debate platform, editing false claims is not sufficient and it should be possible to debate about such claims.

Argüman

Argüman is an independent argument mapping platform developed by a primarily Turkish team, with the goal to promote critical thinking [31]. It is free, public and hosts open debates around various user submitted theses.

Argüman provides a relatively small, but effective feature set. Users can create a debate based on a thesis. Users can place "because", "however" and "but" arguments on the thesis and other arguments. Users can also support an argument they agree with and flag an argument as a logical fallacy.

While Argüman allows for an easy and user friendly debate and it allows users to learn about critical thinking, we think it lacks the ability to teach critical thinking. It lacks the tools for a teacher to manage the students and only minor user correction.

Though no literature is available on Argüman, it is a notable existing solution on the merit of the vital importance of user adoption and user engagement. Considering the traffic, diversity and passion of users in

its debates, there are strong reasons to believe that Argüman here achieved success far beyond MOOD and Deliberatorium. This makes the interaction design decisions of Argüman worth considering. Its plain use of language, streamlined user experience and accommodation to the uncritical user's mode of thought and behavior might have been dominant factors in its low adoption threshold and success.

ConsiderIt
ConsiderIt [32] is developed for Living Voters Guide (LVG) with the goal of helping Washington state citizens making their decisions for the 2010 election. Their goal was to promote engaging with other political views. Beside allowing citizens to express their opinion and arguments about a subject, citizens could also 'signal their own acknowledgment of others' contributions' [32].

After selecting a subject, you have to select a list of pros and cons for that subject. You are able to see the pros and cons made by other users and you can add them to your own list. It is also possible to create new pros or cons. After selecting the pros and cons, you have to indicate with a slider how supporting or opposing you are. Next, you are sent to a screen where you can view the most important pros and cons for the different levels of support. For example, you can see what the most important arguments are for the people that are 100% against it, but you can also choose to see the arguments of the people that had indicated that they were neutral.

The strength of ConsiderIt is that it allows people to propose their arguments for their opinions without the risk that others will attack those arguments. Instead of striving for 'true' arguments and answers, it strives to answer the question why people have certain opinions and it teaches people how others feel about certain topics. This is important for certain (political) questions, questions where 'feeling' cannot be ignored. For example, a fact that there is less crime than ever, cannot overrule the feeling of insecurity. This promotes engaging with other political views and helps people to understand others. To a certain extent it does improve critical thinking. However, there is no debate about the arguments. Users can decide for themselves if they find an argument valid, but they cannot discuss this with other users. We think ConsiderIt is great to broaden your view, but it does not teach you to be critical about the arguments.

Discussion
Looking at the existing solutions, we noticed that design sometimes seems to be a neglected aspect. In a 2012 review of deliberation solutions [33], Van Gelder hypothesizes that design often imposes an excessive amount of structure on the user. The large required adaptation of the user's behavior fit to the structure of the platform is thought to be one of the underlying causes for the lack of adoption and engagement of deliberation platforms. Van Gelder proposes to "cultivate" deliberation rather than impose it. This means that the platform should be simple for the user during first contact and focus on subsequent growth of the user to cultivate healthy deliberation practices. For our product this means we should allow the user to enter a debate with ease. The more advanced features should be used after the user has gained more experience.

Furthermore, we were also looking for ways to *teach* critical thinking. Important part of teaching is feedback on your work, or in this case, the arguments you have made. Because of the feedback you can improve your arguing skills. Existing solutions are limited in their capabilities to do that.

The existing solutions are great ways to *stimulate* critical thinking. Since stimulating critical thinking is close to the core mission of FeedbackFruits, they want to own such an application and have it integrated in their own software set. Therefore, we have to develop a new application.

2.2. Software architecture

This section covers the various considerations and choices we have made regarding the software architecture. We start by evaluating the factors 2.2.1 by which we judged each of the options we mention. One key aspect is the integration with the existing FeedbackFruits platform discussed in 2.2.2. However, this does not mean we had to adopt the FeedbackFruits stack straight away; we still considered other options searching for alternatives worth deviating from the FeedbackFruits stack. This was done for both front end (section 2.2.4), back end (section 2.2.3) and databases (section 2.2.5). These choices are then summarized in section 2.2.6. Furthermore, we discuss available libraries for the problems we will be tackling and other useful tools in section 2.2.7.

2.2.1. Considerations

We have to operate under the assumption that our product is a success and thus will continue to be maintained and possibly further developed by FeedbackFruits after this project is finished. This means that main-

tainability is key. Therefore, we limit our options to those of mature projects with solid time spent in production environments.

This maturity requirement also eases the choice. Since all these platforms are in use in production worldwide, we can safely assume reliability and performance should not be a major issue.

2.2.2. FeedbackFruits platform

The FeedbackFruits platform, the 2.0 to be precise, consists of a separate front and back end. The back end is written in Ruby and uses the Ruby on Rails framework. Its API follows the JSON API standard. It uses a PostgreSQL database. The front end uses the Ember framework and thus uses JavaScript and HTMLBars for templating. Besides those, FeedbackFruits has microservices for its accounts, providing authentication and such, and for media, providing file conversions and previews. In all, the FeedbackFruits platform can be leveraged to very cheaply provide chat, commenting, upvoting and many more. It also provides a stable platform to build upon.

2.2.3. Back-end frameworks

Here we discuss various different back-end frameworks. We decided to create a REST API to serve as our back end, a proven industry standard used to provide APIs for both GitHub[1], Twitter[2] and many more. This made our back end pattern remain consistent with the rest of the FeedbackFruits. This means that the amount of work done by the back end will be relatively minor and the majority of the logic will be done by the front end. All the options considered have ORM support and libraries for other databases; they all provide structures for shifting work to background jobs.

Ruby on Rails

Rails is a framework which greatly depends on the dynamic character of the Ruby language. It gives great power with little code. This can be confusing at times, but once the programmer is more experienced, it can lead to very readable code. Rails does occasionally get some bad press because of its performance which can be lacking, but this can usually be solved by optimizing performance critical code. Several team members have extensive experience in Ruby on Rails.

Node.js

Node.js is a run-time JavaScript environment, allowing one to run JavaScript server-side. It has gained massive popularity in recent years, mainly due to its low entry level. In terms of features, there are libraries for nearly everything, but its ecosystem has only recently started to mature. Only some of us have limited experience with it.

Spring

Spring is a framework originally written for Java, but it is also compatible with other JVM languages. Being written in Java, it inherits some of the verbosity of the language. However, it also gains its solid performance and reliable static typing. One of us has done some minor work in Spring.

Django

Django is a framework written in Python. It has less of the magic of Ruby, but still provides a flexible API with Python concise syntax and dynamic typing. Django is also known to suffer performance, but can also be optimized. None of us have any experience with Django.

Conclusion

All these frameworks, with the exception of Node.js, come with support for all major conceivable requirements: ORM, JSON API support, background jobs and many more. A choice for any of them is a fine choice. Node.js falls short in this regard and choosing it will mean many more choices for various libraries. From a technical point of view there is no framework that stands out. Considering we're operating under the assumption of adoption by the company, maintainability is a key requirement. This means we think Ruby on Rails is the best choice, as it is adopted by FeedbackFruits.

[1] https://developer.github.com/v3
[2] https://dev.twitter.com/rest/public

2.2.4. Front-end frameworks

The front end will be the biggest part of our product. The back end will be a REST framework, meaning all the logic will be left to the front end. The industry is saturated with JavaScript frameworks, so a choice is difficult.

Ember

Ember is the most mature web framework on the market. However, not all its history is without trouble; documentation for its now obsolete 1.0 API is widely available on the web and it is known for including breaking changes in minor releases. On the upside it provides a lot of scaffolding and conventions out of the box. With all its conventions and internal code it does have quite a steep learning curve early on. When used the way it is intended, it allows for fast development. Some of us have quite some experience with it.

React

While not actually a web framework, React is commonly mentioned in the same breath. React moves the templating to its JavaScript and embraces a stateless approach. React is not actually a full-fledged framework, but more a library. It limits itself to the templating and leaves the rest to the user to pick his own preferred libraries for.

Angular

While the migration of Ember from 1.X to 2.0 was not without troubles, Angular ups the ante. Angular 1.0 and 2.0 are in no way compatible and migration can be a bother. While the community is still recovering, Angular appears to have stabilized and now provides a neat package with support for TypeScript, Dart or JavaScript and its own template language.

Vue.js

A relatively recent and popular framework is Vue.js, which is kind of a library consisting of easy-to-use tools that can work together. The main advantages of Vue.js are its reactivity by using plain JavaScript objects and its reusable components. Because it is not an opinionated framework, you need to make architectural decisions yourself.

Conclusion

All these frameworks have their pros and cons. React requires more research into various other libraries for supporting a JSON API and such. Ember requires strict adherence to the guides for continued support. Angular has a large API and Vue.js is easy to use. Overall, all of these are okay. Thus we see no reason to diverge from the one used by FeedbackFruits: Ember.

2.2.5. Databases

While sometimes overlooked, databases are an important choice to consider. They determine much of both the performance, reliability and other design. The differentiating factor is the choice between a traditional relational SQL database or NoSQL database.

Relational and NoSQL

In recent times NoSQL databases have appeared and gained popularity. Document stores give developers the option to shove an entire object at once in the database without considering its structure. This allows for fast prototyping. However that lack of consideration has proven itself to lead to performance issues in the longer term. So much even that FeedbackFruits has migrate away from a NoSQL database at the end of 2016. We opted for the more battle tested relational SQL databases, which we all have experience with.

PostgreSQL

PostgreSQL is an open source relational database management system. It has one of the biggest feature sets of the industry, including optional object storage. It still has a lot of active development with regular releases.

MySQL

MySQL is split between a community supported open source version and an Oracle backed commercial version. It is very popular, yet not very feature rich, implementing a subset of the SQL standard..

Microsoft SQL Server

Microsoft SQL server is mostly targeted at larger enterprise software. Yet still provides a very stable and feature rich option.

Conclusion
All considering, all mentioned databases have performance suitable for our application. All of them have proven to be fast enough for a far larger scale than we expect our application to be. Again, here we will go with the ease of following the conventions by FeedbackFruits.

2.2.6. Summary
In the end we found that a choice for a framework is, in general, more a choice of personal preference than crucial for the success of the product. While some frameworks may work better in certain cases, all of them can be adapted for practically all applications. Ultimately we settled on following the FeedbackFruits stack, which allows for easing integration, better maintenance and a lot of code reuse. While this can be considered a foregone conclusion, we think looking at alternatives was a prudent approach.

2.2.7. Tools and Libraries
While the choice for Ruby on Rails and Ember make the amount of choices for libraries and tools a lot smaller, we still have to make some decisions for our more specific use cases.

Graph
The biggest feature of our product will be the argument mapping. We require at the very least configurable line styles and boxes. We couldn't find a JavaScript library meeting even these requirements. So we'll have to roll our own.

Tree manipulation
We found some libraries for managing tree data structures in JavaScript. Most notably `arboreal`. Since Ember requires us to write our models in a DSL and the traversal is mostly done through components, we suspect we won't have a strong need for such a library.

RuboCop
RuboCop is the code style checker used within FeedbackFruits for Ruby. Adopting it is prudent to ensure the same code style.

RSpec
FeedbackFruits uses the RSpec behavior driven development testing framework. The behavior driven approach allows for an almost natural approach to testing. We quite like it and hence will use it.

Ember Suave
Ember Suave is the code style checker user within FeedbackFruits for both HTMLBars and JavaScript. We will be adopting it to match the existing code.

Version control
FeedbackFruits requires the use of Git and GitHub for version control. These are tools we are familiar with and, in general, we like how they work.

2.3. Design goals
As stated in the problem analysis 2.1.2, the main requirement for the success of the product is teacher adoption. The design goals: didactic value 2.3.1 and good user experience 2.3.2 are logical consequences of the former. Both are hard requirements for a teacher to adopt a new tool.

2.3.1. Didactic value
Although perhaps rather obvious the importance of the didactic value of the product cannot be overstated. It has to be clear and evident for the FeedbackFruits sales department to be able to sell the product to universities and other educational institutions. The didactic value is tri-fold. Primarily the product should teach critical thinking to students. Secondly it should help the students learn about the matter being discussed. Then it should allow the teacher to better gain insight into his students' learning process.

2.3.2. User experience

For all of FeedbackFruits products teacher adoption is key. So it places great value in ensuring high standards of user experience. Our product is no exception and should follow the same standard: Material Design[3]. The design should be simple, providing no more information than needed and entice engagement.

2.4. Conclusions

We have now looked at various aspects of argument mapping, debate, student engagement, peer instruction and evaluation. We found lots of different approaches we can take in order to solve the problem posed. The hard part is limiting us to a range of options we can actually implement and test in the time given. As a result, this project will leave a lot of room for future research. We believe for a study activity having a debate in the form of an argument map is a good approach. Adding roles to the participants is of added value. Furthermore, we wish to enhance this with encouraged student engagement in the form of a peer evaluation after the debate.

[3] https://material.io

3

Design requirements

In this chapter the requirements are listed. We start with explaining the process of deriving the requirements. Next, we explain how the requirements were derived from the literature. In section 3.3 we specify the functional requirements using the MoSCoW method. With the non-functional requirements, section 3.4, we specify the other requirements we place on the product.

3.1. Deriving requirements process

FeedbackFruits has given us a lot of freedom with the assignment. Their only requirement for us was to create an application that improves critical thinking. We had to come up with an idea and pitch it to them. They gave us feedback and we used the feedback to improve the requirements. This process repeated itself until FeedbackFruits was convinced that our idea would improve critical thinking.

3.2. Requirements from the literature

Our literature research provided us with several features which can help us to achieve our goal to improve critical thinking and engagement. Here we list the requirements and lessons that we learned.

Argument map structure
: We found that an argument map is possibly the best way to increase critical thinking. Therefore, we take that as starting point for our application.

Debate roles
: We found that roles in a debate could increase the participation. In section 3.2.1 we describe the roles we are going to use.

Moderator role
: One role we highlighted is the moderator, which we found to be of added value to a debate.

Commenting on arguments
: We found that critical thinking benefits from peer feedback. One way of accommodating this is by providing the ability to place comments on arguments.

Upvoting strong arguments
: This way anyone can quickly distinguish the better arguments in a debate map.

Values
: We found providing users with the ability to specify the values they associate with increases their ability to communicate their opinion.

Groups
: We hypothesize debating in small groups (around 6 people) increases participation.

Furthermore, we found that social interaction is important for engagement. This is mainly achieved through the comment system. Besides the comment system, we also want to notify users when something has changed. This will give the users the idea that they are not working alone on the assignment and will also improve the engagement.

3.2.1. Roles

In our literature research we found that using debate roles could increase the participation. The requirements list several different roles, but they are not equally important for the success of this project. Therefore, some roles are listen under *should*, while others are listed under *could*. Here, we provide an overview of all the different roles that are listed in the requirements.

Proponent and Opponent
These are the two most important roles. Their job is to defend or attack the thesis. They can also attack or defend the arguments that have been made. It is also possible to flag logical fallacies.

Evaluator
The evaluators evaluate the arguments that are made. They can indicate how strong the arguments are and if a flagged fallacy is reasonable. This role is used when the debate has been locked. Hence, it is possible that users first participate in a debate and afterwards evaluate another debate.

Moderator
The role of the moderators is to improve the arguments that are made, but they cannot make arguments themselves. They have the following tasks:

- Decompose arguments. This means that if a node contains multiple arguments, a moderator can split the node into multiple nodes.
- Remove duplicate arguments.
- Remove or edit incorrect logical fallacy flags.
- Propose a consensus as a summary of multiple arguments.
- Participate in the discussion in the comments of a node.
- Propose an edit for an argument, for example when an argument is poorly formulated.

Questioner
The goal of the questioner is to improve critical thinking by asking questions. This can be about an arguments, but also about the subject. The goal of the question is to discover related or hidden issues, that do not fit in a supporting or opposing arguments. Questioners can also give feedback to the user.

3.3. Functional requirements

The functional requirements are classified using the MoSCoW [34] method. The *must haves* requirements are the most important requirements. Without those requirements the project will be considered a failure. Under *should haves* we list the requirement that are necessary to qualify the product as good. The *could haves* requirements are the requirements that will only be done, when there is enough time left. In the final section we list the *won't have* requirements. These are the requirements we are not going to implement, but did consider. All these requirements are also turned into user stories and included in our product backlog, which can be found in appendix C.

3.3.1. Must haves

The *must* requirements are essential for the success of the project. In addition to the features that are essential for creating an argument map, it also includes features that we think are essential for the acceptability.

Creating an argument map
The teacher must be able to create an argument map for a group of students.

Tree structure
To give a clear overview of the argument hierarchy, the arguments are displayed in a tree structure. In contrast to a graph, in a basic tree it is not possible to create cycles or use one argument to support multiple arguments. We think that this will reduce the complexity of the map, which makes it easier to understand.

Adding supporting and rebuttal arguments
A core part of the argument map is adding the arguments. We decided to distinguish two type of arguments: supporting and rebuttal. Argüman has also a third type: "however". This argument is often used as rebuttal, therefore, we do not think that more types would add any value to the application.

Access
Only the teacher and students of the course can view the map. The teacher can control who has access to the map.

Overview for teachers
With a large classroom and many argument mapping assignments running, maintaining an overview of the situation is crucial to the teacher. Providing the teacher with a simple overview of relative activity over the assignments is a first step to ensure this. This could for example be done by visualizing or summarizing the amount of arguments created. If development time allows it, this could be extended to provide further insight by showing how this activity evolves over time and how this activity is distributed over the various debate roles (e.g. arguers, evaluators). This is a first step to enabling guided/informed teacher intervention based on analytics.

Source
Arguments about facts rely on sources. Therefore, users must be able to support their arguments with sources. Sources should be clearly visible and users must be stimulated to support their arguments with sources.

Edit arguments
Users must be able to edit the arguments they have made. This allows them to fix any errors in their arguments. A potential danger in this is that they change their arguments completely. Since the argument map will be used in smaller groups, we do not expect this to be a great risk. A solution for this risk is the use of revision history, which is proposed in the *could* section.

3.3.2. Should haves
The should haves are the requirements we think necessary to ship a good product. We believe that we can realistically achieve these.

Automatic group creation on size preference
We hypothesize that a group size in a debate of around six people will allow for high engagement and good material processing. Of course this is a variable for which a teacher may have his own preference. Still, the creation of groups is something that can be taken out of the teacher's hands and thus make the product more attractive.

Voting on arguments
A simple, but useful, feature widely used in social media, is the ability to like or upvote someone's contribution. This easily identifies the better arguments in the debate.

Comments on arguments
To help students engage in meaningful discussions we want to allow them to comment on an argument. This will give allow on topic discussions where it is relevant.

Activity notifications
When an objection or argument has been made on a user's argument, this deserves a notification of the author. Likewise, when a flat discussion has started on an author's argument, an edit is proposed, or an upstream argument edit has been accepted, this deserves the notification of the involved authors. The method/medium could vary from primitive email notifications to well integrated mobile push notifications.

Debate deadline
Communicating a strict deadline serves to ensure viable arguments have been expressed before the debate is locked and apply a degree of pressure to the student engagement in the assignment.

Live updates of the map
In order to immerse users in the argument mapping we think that live updates of the activity of other users are a should have. This makes the collaboration a lot more powerful.

Debate roles
In literature we found assigning roles in a debate helped both with participation and knowledge processing. We therefore think adding these to our product vastly improves its value. We see the following roles, with some examples to illustrate their focus:

- Proponents
  - Defending a thesis
- Opponents
  - Attacking a thesis
- Evaluators
  - Observe the debate (authors are anonymized for a-partisanship)
  - Evaluate the validity/strength of arguments
  - Evaluate the validity of logical fallacy flags

Role assignment (automatically)
Teacher adoption of argument mapping assignment heavily depends on the workload required. By proposing assignment of students to debate roles this organizational effort can be greatly reduced for the teacher. Initially this can occur by the simple heuristic of automatically random assignment.

Argument decomposing
Existing solutions suffer heavily from composed arguments, making directed rebuttals, support or feedback impossible. To maintain the value of argument mapping, users should be able to decompose a composite-node into separate nodes.

3.3.3. Could haves

Propose edit
We want to give others the option to propose a change to another's argument. Part of critical thinking is not only being able to benefit from the flaws of another's argument, but also being able see it from his point of view. By giving people the option to make a request for an edit, we encourage people to do as such. Writing a good edit request requires proper insight in the other's argument and encourages good debate. On the other hand, receiving a positive and considerate proposal to, for example, moderate one's language can be a non-offensive way of confronting one to adapt one's arguing behavior.

User mentions in comments
Nowadays, users have become accustomed to @naming their contacts in a message to gain their attention. In discussions we expect users to occasionally call in their teacher or someone else.

Argument revision history
By implication of having the ability to propose edits, arguments can change significantly with many revisions. Because such changes might invalidate sub-arguments, it is worth being able to consult a history of the arguments.

Reference another argument node
In complex arguments some arguments can be applicable to support more than just its direct parent. Allowing to reference other arguments can serve to reduce the much observed duplication in online debates.

Additional debate roles
In extension of earlier mentioned roles, we would like to add the following roles.

- Moderator
  - Decompose composite-arguments
  - Remove duplicates
  - Remove/edit incorrect logical fallacies
  - Propose consensus when arising
  - Enter discussions to settle disputes
  - Propose argument edits to moderate language
  - Authors are anonymized to promote a-partisanship
- Questioner
  - Question an argument in discussion
  - Give feedback on reasoning of user

Compromise nodes
Apart from regular argument or objection nodes, a merger between two conflicting branches of argument can be proposed. Pro arguments can be made on the benefit of the thesis and objections on the costs of the thesis. Compromise nodes can propose to consider these together in a cost-benefit analysis with certain weights. Mapping the space of such cost-benefit analysis based alternatives could lead to one being accepted and achieve consensus.

Consensus nodes
When two or more branches of arguments actually agree on mutual values or facts, but this is unapparent through different language, or lack of awareness on the participants, consensus nodes can help to converge.

Due to time constraints and limited developer capacity this requirement later got simplified in the form of 'Consensus proposals'. This feature allows a consensus proposal to be added to a node and all argumentation branches below.

Role assignment (manually)
The automatic assignment of roles can be accommodated with the teacher having the ability to make manual changes to support intervention when needed.

Advanced voting on arguments
Based on literature study insights there is a strong value in accommodating for an intuitive and emotion driven user behavior to promote passionate engagement and user adoption. From a didactic point of view there is a strong value in encouraging critical thinking based user behavior based on moral and rational considerations. We introduce the following dimensions of voting to support these:

- Personal preference (judgment based on e.g. sentiment, emotion, social acceptability)
- Moral acceptability (judgment based on e.g. values, virtues, moral-consistency)

Values underpinning arguments
Often a discussion may lead to a difference in core values. In order to distinguish those, we want to include an option for users to denote their values and specify their opinion of the relevance of that value for the argument.

User statistics
In an argument map the writing and actions (e.g. support votes, edit proposals) are incredibly interwoven. This makes it near impossible to assess the activity of a single user. Having a page where statistics about such user activity are summarized is an important first step in bringing insight on these matters to both peers (as a student) and the teacher.

Flag logical fallacies
Consistent logic is a cornerstone of critical thinking. Due to cognitive dispositions like confirmation bias, humans have a tendency to commit formal or informal logical fallacies when reasoning. Allowing users to flag that certain logical fallacy occurs in an argument serves to make this problematic phenomena of our cognition clearly visible and recognizable. This can help raise user awareness and promote them to engage in consistent reasoning without fallacies.

Ad-hoc fallacy explanation
The uncritical user is likely unaware of aspects of critical thinking like logical fallacies. Providing clear and informative explanations at the point where the user meets these features can aid in developing understanding and appreciation.

Peer evaluation
Peer evaluation leverages social control to stimulate student performance. After finishing a study activity, students are asked to evaluate a number of their peers. If this is communicated clearly, it will motivate the students to impress their more closely monitoring peers instead of the more distant teacher. We believe including peer evaluation will improve student engagement and provide teachers with better feedback on their students' performance.

Multiple theses
When our application is used in bigger groups, it is possible to split the group in smaller groups. By allowing the teacher to propose multiple theses, each group can receive a different theses. If there are more groups than theses, only then some of the theses should be reused.

3.3.4. Won't haves
Since we only have limited time for the project, we cannot implement every desired feature. We have decided to focus on features that contribute to teaching critical thinking. The features that are listed here are interesting features, which will improve the user experience, but they do not improve the teaching critical thinking features. Therefore, we will not implement them during this bachelor project.

Role assignment (advanced)
In extension of the requirement above, preference by the student or the teacher can be gathered before the exercise begins and roles are assigned. Student might want a certain role to align with their position on a thesis which could enable more passionate engagement in the exercise. Teachers might want to see student exercise an under-appreciated aspect of critical thinking and moderate this with their preference. By processing these preferences algorithmically, these can be used to inform a more sophisticated role assignment.

Keyboard shortcuts
An often sought after way of navigating a user interface by power-users.

Auto-collapse maps
When the argument map has grown to a considerable size and complexity (branch degree, branch depth), it should still stay "welcoming to a new user". Hence - to ensure user engagement - argument maps should always retain a sense of overview to a new user and never violate the user's confidence to enter it. To mitigate the intimidating or overwhelming factor of large argument maps, a new user should be presented with a "reduced version" on the first visit.

For example, only a few levels (e.g. 3) of a branch' depth should be shown, with the rest hidden (collapsed, which can be expanded on command by the user). Of all branches, only a few (e.g. 5) of the most elaborated (in terms of argumentation) and most popular/controversial (in terms of support/dispute votes) branches should be shown, with the rest hidden. The selection of the argument should reflect their significance in the argument. This also serves to focus a late user that is unfamiliar with the argument on the most significant arguments of the debate first, hence doubling as an on-boarding mechanism.

Retaining some memory of what is expanded by the user would then be needed. When the user has become familiar with an argument and has started to explore beyond the initially provided selection, the "boundaries of what is visible are pushed" by the user. Branches are expanded and hidden branches are revealed. These changes should be remembered, so that they track the user's mental model of the argument map. As the user uncovers more of the argument, these should stay with the user as the user returns to the argument map later.

Automatic arrangement of map
To provide new users a focus on the most relevant argument first and the fringe argument second, the map could be automatically rearranged to put the most contentious arguments in the most discoverable location (e.g. first in an ordered list).

Public debates
Allow unregistered users to create debates and participate in free debates (without the assignment of roles). This would have to be separate from the assignments used in courses.

Argument duplication mitigation
The larger a debate becomes, the higher the probability that the user is unaware of arguments on the fringes of the debate. Unsurprisingly, online debates suffer from an escalation of redundancy and duplication, crippling them with a lack of focus and complexity.

Our literature study informed us to initially consider small groups of participants in debates to promote student engagement. Since our debate sizes will be small, we expect the amount of duplicated or similar arguments will not be significant crippling factor.

To support larger debates, developing a method of recognizing duplicates can aid at various points to tip users of already existing and similar arguments. Encouraging the merging, supporting or revision of an existing argument is then a large productivity boost. This could also encourage listening skills in the uncritical student to carefully consider the input of others, instead of dismissing them on the grounds of bias.

Export to file
For example to PDF or CSV. For administrative purposes and to support primitive ways of integration with other tools.

Generate minimap
For the very large and complex argument maps, a zoomed out minimap can aid in navigation for the expert user.

Rearranging arguments manually
Where automatic rearrangement turns out to be inadequate, moderators can intervene by manually adjusting the arrangement of the argument map (e.g. drag and dropping).

Mobile support
Mobile use is growing rapidly. However, the large two-dimensional size of argument maps is incompatible with the small screen size of mobile devices. Finding some method of navigation or alternate presentation would be required so solve this. Currently the user base consists primarily of desktop users, hence this is not a high priority.

Anonymous posting budget
Some arguments - like controversial political ones - might be socially unacceptable because of taboo, or morally unacceptable but socially very popular. Because posting an argument goes on record, many users will suppress the expression of these arguments out of a fear of backlash and punishment.

Because we aim for a learning experience where we promote accountability intellectual growth, also in one's moral career, we will need to provide an environment where expression is encouraged rather than suppressed. Expression makes these arguments then become subject to discussion and debate. When good reasons are provided against it, in a civil fashion, owners are much more likely to revise their position and way of thinking which gave rise to this position rather then when these are never expressed.

Allowing users to post arguments anonymously can be encouraging. To prevent abuse and general unaccountability we could limit this anonymous posting to around small amount (e.g. 3). Subsequent subarguments, discussion and upvoting is anonymous for free, but in possible gamification does not yield rewards for the author.

3.4. Non-functional requirements

In this section we discuss the non-functional requirements we place on our product.

3.4.1. Maintainability

Maintainability is the measure of effort required to add features and fix bugs in a codebase. As the product maintenance will be taken over by FeedbackFruits, it is very important to keep it maintainable. We hope to achieve this by adhering to the standards set by FeedbackFruits. This means using code style checkers, 100% backend test coverage and code reviews by at least 2 peers.

3.4.2. Usability

Usability is measure of the ease of use and ease of adoption of a product. As discussed in user experience 2.3.2, usability is of great importance of teacher adoption and the ultimate success or failure of this product.

3.4.3. Performance

Performance is the measure of time needed for a piece of software to perform an action given a set of resources. It has a major influence on the usability of the product, so we need to take that into account in the implementation phase. The performance is mostly determined by the requests between back end and front end. Although we cannot expect FeedbackFruits to scale up their servers because of us, we can achieve good performance by critically thinking about programming decisions, as this is under our control. Amongst others, this means limiting the amount of requests to the database in the back end server, preventing unnecessary loops and trying to delay the I/O intensive work from blocking the request if possible.

4

Process

In this chapter we discuss the various aspects of how we went during the development phase. We go in depth about our software development methodology in section 4.1. Our plans for quality control are explained in section 4.2. In section 4.3 we explain the way we distributed our work. In section 4.4 we discuss the work location and its effects. We had some issues with getting started, this is discussed in section 4.5. Finally, we end with how we handled both internal and external communication.

4.1. Software development methodology

This section explains the methodology we used for developing our product.

4.1.1. Setup

For the development we settled on a stripped version of SCRUM. We employed daily stand-ups, to keep everyone aware of what is being worked on. We had sprints of one week, with a sprint reflection and planning on Monday morning. In this meeting we also made time for some personal interaction, important discussions and updates on upcoming meetings. We mainly worked together at the same desk in the FeedbackFruits office. This allowed us to quickly help each other should one become stuck.

4.1.2. Roles

At the beginning we allocated the roles of president/project leader to Boudewijn and made Felix the general secretary. We quickly shifted this to having Felix be the secretary for the external contact and rotating the role of internal secretary every week to spread the load generated by that task. The role of president of the meetings was constant except for the few occasions the president had the role of secretary. The role of product owner fell on the team as a whole. This allowed us to more thoroughly prioritize and plan features. We think this worked fairly well overall, although this did require everyone to keep track of the project progress.

4.2. Plans for quality control

FeedbackFruits has a person employed for quality control. We allowed this person to check our product to make sure it is up to FeedbackFruits standard and fixed any issues this person reported. Further quality is ensured by the very positive SIG score. Unfortunately, we were unable to schedule a user test in a time schedule in which feedback could be used to improve the product, so there we have some room for further improvement (see also chapter 6).

4.3. Work distribution

Some of us had previous experience working at FeedbackFruits, while others were new to both FeedbackFruits and the languages we used. Hence, one of the challenges we faced was distributing the work in an honest, yet efficient manner. We settled on the following distribution: Boudewijn on front and back end, Jens also on front and back end, Robin on front end, and Felix on front end and styling. All in all we believe this led to a good distribution of work, with us being able to quickly finish features. Although work wasn't spread

entirely evenly, which was caused by the different levels of experience, we are more than satisfied with all our contributions.

4.4. Location

FeedbackFruits has created the space for all four of us to work at the same table. In the beginning we experimented somewhat with working from home one day of the week. However, we found that we were less focused in that situation. Also communication and coordination were less strong than when you can physically speak to one another. So we settled of simply gathering in the office. This helped us greatly.

4.5. Getting started

The FeedbackFruits stack is not a simple one. The front end has a deep and wide dependency tree, a common issue with node.js based projects. This causes installation to take some time. Also our backend requires a Redis, Elasticsearch and PostgreSQL server running. In better cases setup can take one day. One of us had issues trying this on a VM and his hard drive, which meant he needed several attempts before getting everything working. This cost him a few days. The team tried to help where possible, this eventually led to the issue being resolved.

4.6. Design

As previously stated, literature review and company experience taught us that design is very important for product adoption by both teachers and students. We all have a background in computer science, so we found it necessary to take several steps to maximize our chance of success on this challenge, despite our lack of design expertise, which we detail below.

4.6.1. Literature research

As detailed during our research phase, we spent a notable part of it on researching the fundamentals of student engagement. This helped those already with experience in the company on reviewing and reweighing our assumptions on the subject. For the other team members this helped to develop an awareness of this priority and its primary requirements.

4.6.2. Dedicated team member

Fortunately, we had one team member with a reasonable amount of design experience. We found it necessary to dedicate him to primarily focus on this challenge. This would go on to ensure that this potential is optimally exploited.

4.6.3. Adherence to company practices

Another fortunate advantage is that two group members already had experience in the company. They were well aware of lessons learned earlier by the company, as well as the practices and standards adopted in response to those. From the start we had the intention to stick to these for our benefit and we believe doing so has paid of. The most important goals of these company practices are explained below.

Adherence to design language and UI framework

Previously, the company already decided to adopt Google's Material Design as the leading design language for its products. This was done to save the design research and implementation effort of developing a comparable design language in-house.

Material Design is scoped to specify everything between interaction of the interface to the graphic details of the interface. FeedbackFruits has been implementing and extending this into an Ember.js based UI framework available for internal use.

We decided to adopt the work done on this UI framework as much as possible, so we would benefit from the design research and development iterations already done by Google and FeedbackFruits.

Internal design review

During the design process interface mock ups would be posted for review by both the team members, the resident designer active at the company and several other employees who expressed interest or affinity with argument mapping. This yielded feedback which was processed before implementation.

User testing
A central tenet of FeedbackFruits's policy is early and frequent user testing. These serve to validate the design assumptions made, to discover where these were incorrect through feedback and gauge the overall user experience.

We spent considerable time on preparing tests, but sadly this was a troubled process. We still managed to perform a few small user tests which yielded very valuable feedback. More on the setup and results of these can be read in section 6 dedicated to this.

4.7. Communication

In this section we discuss how we managed the various routes of communication, what went good or bad and how we adjusted. We first discuss the internal communication (section 4.7.1), then the communication with FeedbackFruits (section 4.7.2) and lastly the communication with the TU Delft coach (section 4.7.3).

4.7.1. Internal communication

As previously stated, our internal communication involved daily and weekly updates. This, coupled with our physical colocation and work times with the addition of a Slack channel, sufficed all our communication needs. One of the drawbacks was our tendency to put on headphones and music in order to drown out office noise. This caused one case where a task was done twice by two different people. Overall communication went smooth, with meetings being of reasonable length considering the amount of items being discussed. The reasonable length was for a big part caused by the structured way we held the longer meetings, such as the weekly sprint meeting. The detailed agendas we used, forced us to focus on the current agenda item and kept us from deviating into another discussion.

4.7.2. Communication with FeedbackFruits

Communication with FeedbackFruits went very well. It was smoothed by us working in the same office and some of us being familiar with most of the employees, having worked at FeedbackFruits previously. This allowed us to benefit from changes made by FeedbackFruits and in return we made some contributions to the core systems of FeedbackFruits. We think it is safe to say even apart from our BEP result, we made valuable contributions to the FeedbackFruits code base.

Our communication with our supervisor from FeedbackFruits also went very well. The criteria were clearly stated at the beginning of the project. We were allowed to work autonomously, but were monitored to see if we were progressing properly. Help was always available, when it was needed both for feedback on our product and advice on how to implement a certain feature.

4.7.3. Communication with TU Delft coach

Our communication with our TU Delft coach went okay. Although it was always friendly and no major issues were to be found, we received some constructive criticism. Our coach noted at the midterm meeting that our weekly reports could have proved insufficient to correct our efforts, should that be needed. Thankfully such corrections haven't proved necessary and we adjusted our weekly reports to include a status report of our product backlog. After evaluation this shift proved successful to give our coach more insight in how we are doing.

4.8. Learning goals

At one of the first meetings our coach advised and challenged us to make this BEP not just the delivery of a product, but a conscious learning experience. We tried to stay aware of what we were doing and if there might be a distribution of work where we would learn more. We had to balance this with our productivity and believe we managed well. It required us to be quick to help each other and occasionally be patient while waiting for results. In reflection we think this was very good advise. All of us made significant steps whether in learning a new framework, organizing and guiding a project, presiding, and minuting meetings or team work.

5 Implementation

In this chapter we go in depth about our product implementation and the various choices we made during development.

5.1. Implementation phase

In this section we discuss various important parts of our implementation phase.

5.1.1. Groups and roles

Although the FeedbackFruits platform has support for groups in the form of both courses and single use groups, it lacks the support for groups within study activities, e.g. it is impossible to hand in a document as a group. One employee at FeedbackFruits planned to work on this during the earlier weeks of our BEP and we were hopeful to be able to benefit from his work. Unfortunately, he had been too optimistic in his planning and postponed his work twice. Ultimately, we had to make the decision to implement our own groups and roles, but we spent a couple of weeks waiting on his delivery. During this time we focused on other important features, but productivity could have been higher had we been able to use both groups and roles earlier in the process.

5.1.2. Backlog

During the research phase we first created the requirements using the MoSCoW method. Later we have listed all the requirements in a product backlog (appendix C). In the backlog we prioritize the features even more than in the MoSCoW method. Because of this, it was always clear which feature should be done next. The list is quite detailed and the bigger features, such as roles, are split into smaller features. Not all those sub-features have the same priority. For example, the role of questioner is not as important as the role of opponent. Halfway through the project we realized it would be more valuable to have certain feature sets fully implemented over following our priorities and having less closely related features. We pivoted towards finished edit proposals and consensuses in the hope those would prove valuable.

5.1.3. Overall impressions

In general we are very satisfied with our progress during the implementation phase. We prioritized quite well according to the established product backlog. Exceptions did come up for the mentioned groups and roles, and we decided to drop the evaluator role, since we were not confident on the benefits of that role. There were times where coordination on shipping features could have been better, but many features were delivered in a rather short time, even impressing the people at FeedbackFruits.

5.2. Technical challenges

5.2.1. Performance

The downside of any Ruby on Rails project is that performance is not guaranteed. During testing we noticed significant drops in performance, with response times peaking at seven seconds while running on a Heroku Hobby Dyno. Of course this is unacceptable performance. We investigated this and found that the biggest

cost was the fetching of all the arguments and their subarguments. For each argument a request was sent to fetch all the subarguments. We changed this to including the subarguments in the root element and thus preventing all those requests. This meant we have one longer running request of a couple of seconds, but this will reduce the load on the server for future requests, which should result in more consistent response times. Further examinations led us to believe most performance gains lie inside the core of the FeedbackFruits platform, which we thought to be out of scope.

5.2.2. Modularity

FeedbackFruits has a very modular front and back end. Almost every feature set is contained within its own repository while depending on other repositories. Rails has very good support for this with Rails Engines, which work very nice. Ember only recently gained experimental support for engines, so FeedbackFruits has written its modular front ends through addons and providing modular components instead of routes. This is not the way Ember is best used and it needs workarounds which are bug sensitive.

Because our project is new, we had the possibility to choose how to provide our features to the rest of the platform. We decided to adopt the engine approach, since the features it provides are very useful. This meant we had small challenges to solve in integrating with the rest of the platform. We think this was done sufficiently and in a way that future migrations FeedbackFruits might take can learn from. Creating the back end in a Rails Engine was also a choice which was very simple to make, with no major challenges.

5.2.3. Database tree traversal

Another issue we faced was traversing our tree in the database. Luckily, PostgreSQL provides recursive queries, which allow for such queries to be done, without making concessions to the table structure. This did require a small time investment in learning how to use those. This proved to be relatively simple, while still being efficient and made us very happy with our choice for PostgreSQL.

5.3. Product result

In general, we can be very positive about the end result. We completed all our must haves, most of our should haves and even managed to finish some could haves, see also the product backlog C. In summary, we have made a product which allows a teacher to distribute students in debate groups and the proponent, opponent and moderator roles. The groups will then each go in debate with the students researching the thesis. They will be able to create arguments and see each of those in real time. They can discuss the arguments and propose edits or consensuses. For the details, see appendix E. All together, we found these features to facilitate the environment necessary for both learning critical thinking and studying a subject.

5.3.1. Schematic overview

To better understand the core functionalities of the debate, a UML diagram can be seen in figure 5.1. The basic elements similar to the existing solutions are the thesis, the arguments and the users. The added value for the arguments is the ability to upvote them. For the users, the most notable difference compared to other systems is the role some user has. Other elements are the presence of comments on arguments, proposing a consensus, which is basically an argument that summarizes its sub-arguments with the ability for users to approve of reject it and proposing an edit for someone's argument.

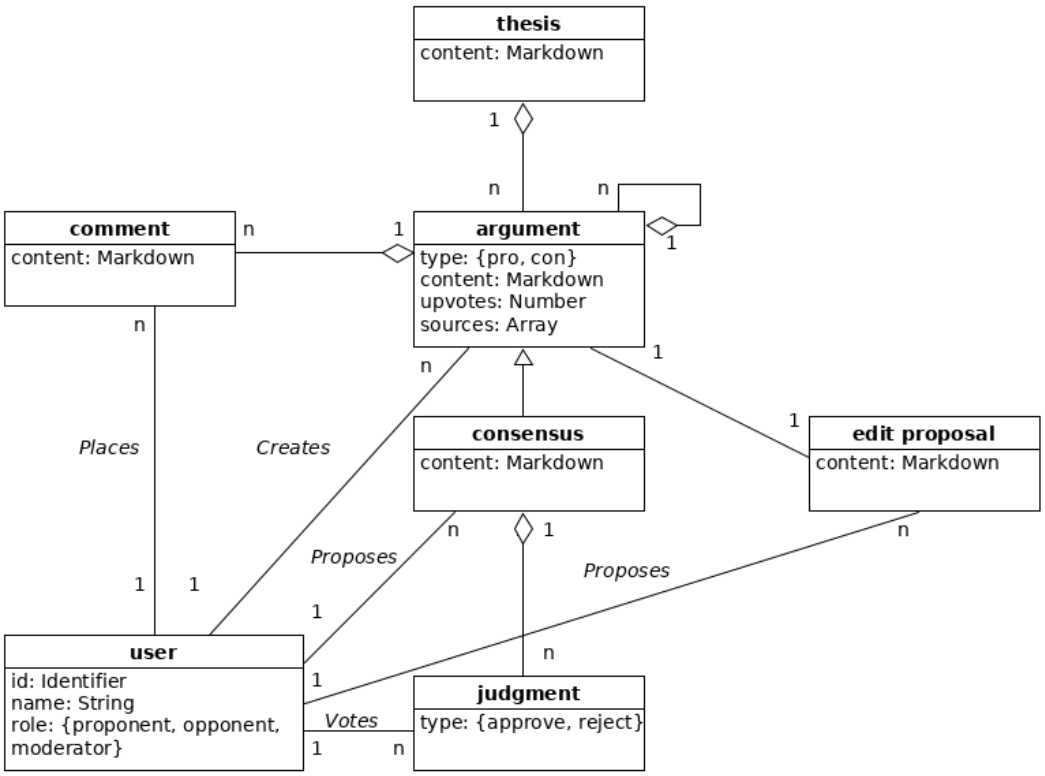

Figure 5.1: Basic UML diagram of the debating tree structure.

The UML diagram does not highlight the user role in particular, as this would make the diagram too complex, hence hard to understand. However, the user role has a major influence on the permissions for the actions that are visualized in the diagram. An overview of those permissions can be seen in table 5.1. Note that there is no difference in permissions for both proponents and opponents. One could argue that proponents should only be allowed to create supporting arguments and opponents rebuttals. However, assigning the permissions for arguments at lower levels in the debate would not be that straightforward. For this reason those permissions are equal. In the case arguers do create arguments not intended for their role, then moderators can detect this and remove these arguments.

Table 5.1: Overview of permissions per role. Teachers do not technically have a role, but they do have the permission to perform any action possible in a debate map.

	Arguer (proponent / opponent)	**Moderator**	**Teacher**
Create argument	Yes	No	Yes
Edit other argument	No	Yes	Yes
Delete other argument	No	Yes	Yes
Propose edit	Yes	Yes	Yes
Propose consensus	Yes	Yes	Yes

5.4. SIG Feedback

Due to a lack of coordination we accidentally submitted our project twice. This meant we also received feedback twice. The one gave us a score of 4.5 out of 5. The main issue being the unit size. The other gave us a score of 5 out of 5. The only complaint was the low amount of front-end tests compared to the back end. We suspect that the first review might have accidentally included our back end specs, which have a certain structure with which the unit size can increase rather fast.

Apart from the lack of front-end tests we were told to maintain the way we were programming. This is what we did, so we expect that the final SIG score will not differ much from the first score. Regrettably, we did not add any front-end tests, which we further explain in section 6.1.

6 Testing

From early on we had a strong ambition to do various forms of testing to validate our work. Of course we had to anticipate there would be false assumptions in our designs and we could expect to find mistakes and unforeseen consequences in our implementation. To counter these aspects of software development reducing the quality of the product, we developed both automated unit tests of the code running and performed several user tests to perform integrated validation on the user experience and interaction design. In this chapter we look at both.

6.1. Automated tests

FeedbackFruits is still in the process of tackling automated front end tests, we saw little benefit in investing time to set this up ourselves with the risk of deviating from any progress FeedbackFruits might make in the future, so on the front end we had to resort to manual testing.

On the back end FeedbackFruits has very strict testing norms: they require 100% line coverage for models and controllers. We tried to follow these as close as possible, resulting in 97,47% line coverage overall, while achieving 100% line coverage for our model and controller logic. This is something we are very happy about and we believe this will prove very valuable in the continued maintenance.

6.2. User tests

User tests serve to provide a user centered perspective on the performance of the integrated product. We strongly desired user feedback as early on as possible, in line with company practice. So, as soon as the product reached a level of functionality where basic argument mapping was possible, we held our first user test with employees of FeedbackFruits.

6.2.1. Ethical concerns

One of the major setbacks we faced was the approval process of our research in the form of user tests. Because we are a TU Delft project, and our user tests involve humans, TU Delft regulation on this research applies. In our case this means we needed to acquire approval of the TU Delft Human Research Ethics Committee. In part this was a failure on our side to become aware this requirement applied. By the time we were tipped by our coach, we were already nearing the project mid-term. We quickly learned that the application process for our scheduled tests with a high school class would take in the order of a month to go through, which meant that we had to cancel that session. This long application process was triggered due to the age of our high school students, who would often still be minors, and legally unable to provide consent. This meant we had to pivot to a smaller user test with participants who were of age with a very short time frame. Unfortunately, the application for this test got stuck in bureaucracy and we were unable to get approval within time and thus had to cancel that session as well. After this we again pivoted to the resulting user test schedule, detailed below.

6.2.2. Product user tests

User test with teachers

Because we still had not obtained the approval from the Human Research Ethics Committee, we would not be able to record any personally identifiable data. To resolve this, we mostly stuck to high-fidelity prototype based user testing.

We had two teachers go through the entire process of creating and preparing an assignment on the platform to administrating it during course runtime. Feedback and observations were noted down and later analyzed into a problem-proposal structure. We tracked the amount of accounts of a certain problem. The severity of the user experience gain/pain expressed by the user was then interpreted together with our design experience and that of the other resident designer to establish confidence values that the suggested interventions would be beneficial. We were by now entering feature freeze, so this analysis would serve as a guideline to prioritizing the future implementation of these.

Subject	Feedback	Suggested intervention	Accounts	Confidence
admin.app-view.setup-task.groups	Descriptive text on group setup is quite confusing. Mental model building of the group structures clearly failed.	Use of infographics!	1	0.8
admin.app-view.setup-task.roles	The purpose of the debate roles isn't very clear to the teacher. This arrises when setting up the groups and choosing a roleplay setup.	Ability to click on group in infographic, with tooltip for extra affordance. Or a info icon/button next to the roles.	1	0.7
admin.app-view	Title field of the assignment was confused as the field for the debate thesis	Recompose the step order of creating an assignment, where adding the title is one of the last steps	1	0.3
admin.app-view	Order of steps feel un natural. Would expect something on a more expanding nature, starting from the course, developing to the elaborate.	Start with the didactical purpose of the assignment. Select the appropriate group and roleplay for it. Then think about appropriate theses for this purpose. Read about how to find one (examples ... Teachers Network?). Pick from suggestion or fill one in. Write an introduction to finish it off. Save it by adding a title and location.	1	0.5
admin.app-view.debate-task	What if I work realtime or on short notice, and I just noticed the group is ready. Critical moment, gotta get it started now that I still have their attention! I want to hit start now, but it's hidden in the edit mode	"Start debate now" button	2	0.8
admin.app-view.debate-task	Actions on the table rows feel inappropriate	Clicking row opens up the details dialog, button at the end opens up map.	1	0.4
admin.app-view.debate-task	Metrics in the table rows might not be the most relevant, and don't clearly connect with the overview metrics above, which starts with percentage started and average time spent per user, and only then continues with arguments made.	Percentage started and time spent per user instead. And if room left, also the others.	1	0.6
admin.roles.moderator	What if the moderator spots something very nasty? What should it do? Just handle it by him/herself? Can't it get in touch with the teacher to discuss/report this?	Moderator-to-teacher hotline in form of chat	1	0.7
admin.roles.moderator	Adversarial arguments against editing modes of moderator: suddenly changing an argument is kind of crude and offensive to the author. Edit proposals are more civil, but what if it's really urgent, or a tiny obviously good thing (spelling)? What if it needs to be handled by the moderator doesn't dare or know how to and the teacher should?	Provide all options to cover these usecases: 1: Direct edit 2: Propose edit 3: Report to teacher	1	0.7
map.consensus	Maybe consensus isn't the right term because it might be interpreted to apply to everyone in the map agrees?	"Summary" instead?	1	0.3

Figure 6.1: An excerpt of the teacher user test analysis.

Early internal user test (product in alpha)

This was a test using our own employees, who albeit familiar with our platform, had close to no mental model present on our product. The interaction with the platform was limited to signing in, browsing to the activity and opening it. This consisted of minutes of interaction, from which all interaction was dictated by our product code. This assured to a decent level that the platform had a negligible influence on the feedback on our product (common platform features like authentication are outside the scope of the project anyway).

As expected, the product was still considerably crippled and bug riddled, and was far away from yielding insight on the value of the tool for critical thinking, because this experience was stranded in the lower level of user interactions, as expected. The test did show large engagement on the users - an important criteria - though this could also be attributed to company culture bias. Above all this test had a strong positive value in aligning our development priorities with the larger user experience pains and potential gains. Issues were logged in GitHub and tackled in the next weeks.

Midterm internal user test (product in beta)

During the midterm week another internal user test was scheduled with the same purpose. By this time the classical debate roles feature was implemented, as well as several other features aimed at empowering emphatic moderation behavior.

Together with a thesis which related more to the company culture and explicit request to only engage in serious debating, this resulted in the first use as designed for. Still only about a third of the activity consisted

of use as desired. It is unclear how much of this could be attributed to factors outside of our control (like tired employees or absence of the pressure to pass in an actual course), but clearly there was space to take further steps to guide to more productive use.

This test yielded an even greater amount of user feedback, which showed a clear shift towards a higher level of user interaction emerging like longer running social interaction. These were most likely still not or rarely engaging the user in any critical thinking.

We hypothesized that further empowering moderators and the teacher to mitigate trolling and other unproductive behavior would be the most significant interventions we could make in the remaining time to guiding the users to more intended use.

This test also again exposed a shortage of power on the side of the teacher to control overall grouping and role allocation situation and make intervention there. Though we already expected this and had planned for these features, it was becoming clear that these would not fit into the development schedule anymore and remain as a top priority for future development.

All feedback was logged into GitHub as issues, though many were considered outside the scope of the project due to time constraints.

Student user test (product release candidate)
We have a user test planned after the writing and submission of this report. We hope to be able to present the results of this test at the BEP presentations, given we receive the required permissions in time.

6.3. Client response

FeedbackFruits's response to the internal tests was and remains very positive. The company has learned to look through the bugs, knowing they can be fixed, and it sees an engaging study activity full of didactic potential both for teaching critical thinking and for application in less rhetoric centered courses. The company would have preferred to see a test with actual users, but it is understanding our mistake. Furthermore, FeedbackFruits has asked us to place extra effort on fixing as many of such bugs and usability issues as possible, knowing that stability is something universities look to before considering adopting tools. In the last weeks we directed our efforts towards increasing the stability and usability and we hope FeedbackFruits will further validate our work.

7

Discussion

In this chapter we discuss our end result. We look at the product from an ethical point of view in section 7.1 and compare our product to existing solutions in section 7.2.

7.1. Ethical implications

In this section we discuss the ethical implications of our product. We consider both the ethical benefits (section 7.1.1) and risks (section 7.1.2), then we draw our conclusions in section 7.1.3.

7.1.1. Benefits

Our tool is ready to become a valuable addition to the collection of tools FeedbackFruits offers. It could offer teachers a very good option to enrich their lessons and encourage critical thinking. It could also add to better equipping students for later in their lives when they are faced with the increasing exposure to contradicting claims and information. This is especially relevant on topics to everyone's interest like health, medicine or nutrition, as well as common responsibilities in society, professionally or politically.

Furthermore, with little effort our tool could be extended and developed to more of a deliberative application. In that form it could be used to analyze issues with several stakeholders. Again, greatly benefiting its users.

Another possible direction FeedbackFruits could take, is to extend the tool to support massive amounts of users and opening the debate for everyone. Allowing anyone to crowdsource a debate to the entire Internet, similar to Argüman. Although care should be taken to keep these discussions properly moderated, it would encourage more insightful debate than other forms of Internet communication.

7.1.2. Risks

As in any debate, there is a risk of debates hosted in our tool becoming polarized. This is decreased by its deliberate design to equally respect expression of supporting and objecting arguments. However, polarization could still be envisioned to occur through social interactions. For example, an opposing group could be shunned from the debate through aggressive, unemphatic and swift decimation of their argumentation. Once ostracizing them by shear inhospitableness, an ideological echo-chamber could develop where critical argumentation could remain absent. Hence, its use should always be supervised should such situations arise. We find this risk acceptable, since it already often arises in common debates and we expect the probability to be lower with our tool in the supervised context of the classroom.

Another risk one could envision is that the tool becomes brutally efficient at aggregating compelling justification for just claims. When such claims are of a sensitive nature (i.e. the reader is highly emotionally invested in them being right or wrong) such as with religion, economic doctrines or other ideologies, it could become a disturbing experience the use. However cultivating the intellectual courage to confront ideas and reasons on tough questions of truth is an essential aspect of critical thinking, so designing against this risk could be self-defeating without probable cause. To ensure use is guided in a fashion that avoid unnecessary suffering we again consider the classroom with an emphatic and perceptive supervisor a suitable application. The tool was intentionally designed to be un-opinionated on the claims and argumentation used for debates to ensure the risk of abuse is not increased by this mechanism.

7.1.3. Conclusions

We find the risks of our product to be minor, not exceeding any risks resulting from day to day social interaction. The responsibility for this will have to be shared between the teacher using our tool and FeedbackFruits. The benefits of it greatly outweigh the rather minor risks. If developed further it could prove to have even more of an impact.

7.2. Comparing to existing solutions

During the research phase we have looked at existing applications that promote critical thinking. In section 2.1.5 we described these applications and what we could learn from them. Now we can add our own product to that list and therefore it would be good to compare it to the others. We only compare it with to most related applications, applications where you can have an in-depth debate about a thesis. This means that it should be possible to debate about arguments. Therefore, we only compare it with Deliberatorium and Argüman. In this section we evaluate if our application is the best option for the use case it was designed for, namely promoting and teaching critical thinking in classrooms.

7.2.1. Deliberatorium

As we have previously stated, Deliberatorium is meant more for discussing an issue and its possible solutions than debating or teaching critical thinking. Discussing an issue requires many nuances which Deliberatorium provides through its issues/ideas/arguments structure. This does come at the cost of steep learning curve, further worsened by a lacking design and bad user experience.

For debating we have a far better user experience and feature set more fitting to the task. Our presentation of the arguments in a graph makes the structure more insightful than the nested list presented by Deliberatorium. While we miss the dedicated option of asking questions provided by Deliberatorium, our comments section on each argument does allow for this.

We also provide features specific for the teacher to create and manage the debate, which Deliberatorium lacks entirely. All in all, our solution is better for teaching critical thinking through debate.

7.2.2. Argüman

Argüman is most similar to our application, especially visually, but there are a few differences. The most notable difference is that in our application users get a certain role (opponent, proponent, moderator), while in Argüman everyone can add supporting and rebuttal arguments. We assume this is mostly due to the different purposes of the applications. Argüman is intended for being used as a debate platform, allowing to create both supporting and rebuttal arguments. In contrast, our tool focuses on the teaching part of critical thinking, to which assigning different roles should contribute. Research is needed to clearly indicate which approach is better.

As stated previously, we consider Argüman to be very good at learning critical thinking, but lacking for the teaching of critical thinking. For that purpose, we find our tool is better suited.

Compared to the other applications we belong to the visually attractive and we provide features that can be used to teach critical thinking. During the design of our product we had a strong focus on user experience. This resulted in special overviews for the teacher, a visual appealing application and intuitive controls. Furthermore, all the applications tried to promote critical thinking, but our application has also a strong emphasis on *teaching* critical thinking. To achieve this, we have tried to find ways to improve the arguing capabilities of the users. For example, the comment system and edit proposals can be used to improve existing arguments. All in all, we think that this application is a valuable contribution to the existing critical thinking applications. Especially in classroom usage we think that we are the best option. However, more research is needed to verify our claims.

8

Conclusion

The goal of this project was to create a product incorporated in the FeedbackFruits platform that promotes critical thinking in education. We achieved this by creating a study activity centered around arguments with support for debate roles.

For the success of this application, teacher adoption and student engagement were identified as dominant design challenges. In the design process we built on lessons learned by FeedbackFruits. This process resulted in a list of design requirements. For example, teachers are provided with analytics to provide an overview of the debates. Students are encouraged to engage in collaborative and constructive social interaction around arguments using features like upvoting, comments, edit proposals and consensus proposals.

Our application is visually best compared to Argüman, but our solution was designed for teaching critical thinking, resulting in a study activity that could be easily adopted by teachers. In addition, we introduce several unique features focused on the classroom environment, like assignment of different debate roles (e.g. moderators) and discussions to improve the arguments itself. Though still preliminary, we found indicators that our investments in the design process resulted in a satisfactory usability and user experience of our product. Didactic validation as well as several other key improvements have been suggested for future work.

We already received very positive feedback from the client. FeedbackFruits found it lies close to the core mission set for itself: improving student engagement and critical thinking. Even though FeedbackFruits is not in a position to immediately continue the development of our tool, they have agreed to maintain it indefinitely and hope to be able to develop it in the long run. From a business standpoint we have agreed it was a success.

9 Recommendations

In this chapter we provide our recommendations learned from having gone through the project. First we discuss the intended applications in section 9.1 and then future work we hope to see in section 9.2.

9.1. Applications

We believe our product is suited for most students starting at high school level through to university. It can be a great priming tool, serve as an evaluation halfway throughout the period or replace discussions which would normally take time out of normal lessons. We hope FeedbackFruits takes the opportunity to test all these use cases.

We hope FeedbackFruits will take this tool and promote its use in the education of critical thinking. We cannot be certain as to the impact it will have, but we think it shows great potential.

9.2. Future

During this project one of the hardest things we had to do is limit ourselves to the things we knew we could realistically implement and test. This means we have a multitude of assumptions we think need testing and have lots of features with great potential.

9.2.1. Assumptions

Group size

Early on we decided to design our product for debates with small amounts of people, hence avoiding the technical challenges that come with the amount of data generated by large amounts of people. Also, we believe that the smaller circle of interaction will help students to be more engaged. This was initially based on the finding that in normal debates the amount of active participants is limited to about 6 people.

We were unable to find support for this hypothesis for online debates. This opens up the possibility for future research in this field. We advice FeedbackFruits to allow the DebateMap tool to be used to study the effect of the amount of participants in a debate on the engagement of the participants.

Roles and debate variations

Another one of our assumptions was that introducing roles would prevent people from getting stuck in their own opinion. We implemented the roles of moderator, proponent and opponent. We also considered the role of evaluators, with the task of simply observing the debate and at the end rating everyone's performance. The role of questioner, someone who simply poses questions to the arguers, could also have been a valuable addition.

Even more variation can be found when considering a preparation step before the date, to outline the stakes held by the stakeholders or rotating the roles halfway throughout the debate. The tool could also be used to map a real life debate and then continue online. We hope further study will help gain insight into this.

Effect on critical thinking

Literature showed that argument mapping is a promising method for teaching critical thinking. Products using this method already exist for the classroom, some with an impressive list of educational institutions as

customers. The free-for-all argument mapping platform Argüman enjoys a high traffic of users and extensive debates. However, our platform changes many variables and has not seen any real world use. So, it would be premature to draw conclusions on its efficacy to cultivate critical thinking.

Rigorous research on this matter was precluded from the scope of the project due to the effort to set up controlled tests with meaningful sample sizes, as well as obtaining respected tests like the California Critical Thinking Skills Test. Of course this question is still of strong interest to us, also because it so closely aligns with the company mission. We hope that future use of the tool in real classrooms will already start answering this question by yielding data that can be used to start building inferences on its effect on user behavior.

9.2.2. Potential features

Logical fallacies

We have always liked Argüman's ability to flag an argument as a logical fallacy and even included it in our designs. Due to time constraints we decided to focus on edit proposals instead. One of the challenges of flagging arguments as logical fallacies is the requirement it places on the user's knowledge of rhetoric. Therefore, we also considered the ability to simply flag something as logically lacking to alert others. Although we liked this idea, we were not sure it would be a good compromise. We believe experimentation with this could uncover a very good balance between logical correctness and usability.

Emotions and values

Another one of our better ideas was to include the ability to indicate an emotion or value strongly associated with an argument. For example, the argument "People should not interfere with my business" would include anger and self-determination. We found various deliberation tools included this in various forms. Since we focused mainly on debate, we thought this might not be the best fit for the tool. If the tool were to be extended to support deliberation, such a feature could be very valuable to encourage mutual understanding.

Evaluation task

One of the strengths of FeedbackFruits is the ability to receive feedback from both teachers and fellow students, and to give feedback yourself. Once the deadline of a debate has expired, it would be useful if students must evaluate the arguments and moderations made in some debate map other than the one they have been working in. This way the students are sparkled to actively participate in the debate, since their work will be evaluated afterwards.

Bibliography

[1] Oxford dictionary definition of "post truth". `https://en.oxforddictionaries.com/definition/post-truth`.

[2] Scott Plous. *The psychology of judgment and decision making*. Mcgraw-Hill Book Company, 1993.

[3] Milton Lodge and Charles S Taber. *The rationalizing voter*. Cambridge University Press, 2013.

[4] Ziva Kunda. The case for motivated reasoning. *Psychological bulletin*, 108(3):480, 1990.

[5] David G Myers and Steven Michael Smith. Exploring social psychology. 2000.

[6] Donald M Taylor and Janet R Doria. Self-serving and group-serving bias in attribution. *The Journal of Social Psychology*, 113(2):201–211, 1981.

[7] G Ainslie. The cardinal anomalies that led to behavioral economics: Cognitive or motivational? *Managerial and Decision Economics*, 2015.

[8] Sam Butchart, Daniella Forster, Ian Gold, John Bigelow, Kevin Korb, Graham Oppy, and Alexandra Serrenti. Improving critical thinking using web based argument mapping exercises with automated feedback. *Australasian Journal of Educational Technology*, 25(2), 2009.

[9] Robert H Ennis. A taxonomy of critical thinking dispositions and abilities. 1987.

[10] Richard Rothstein, Tamara Wilder, and Rebecca Jacobsen. Balance in the balance. *Educational Leadership*, 64(8):8–14, 5 2007.

[11] Daniel T Willingham. Critical thinking. *American Educator*, 31(3):8–19, 2007.

[12] Tim Van Gelder. How to improve critical thinking using educational technology. In *Meeting at the crossroads: Proceedings of the 18th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education*, pages 539–548. Citeseer, 2001.

[13] Tim van Gelder, Melanie Bissett, and Geoff Cumming. Cultivating expertise in informal reasoning. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, 58(2):142, 2004.

[14] Tim van Gelder. Teaching critical thinking: Some lessons from cognitive science. *College teaching*, 53(1):41–48, 2005.

[15] K Anders Ericsson and Neil Charness. Expert performance: Its structure and acquisition. *American psychologist*, 49(8):725, 1994.

[16] Ruth Kennedy. In-class debates: fertile ground for active learning and the cultivation of critical thinking and oral communication skills. *International Journal of Teaching and Learning in Higher Education*, 19(2):183–190, 2007.

[17] Laurence Simonneaux. Role-play or debate to promote students' argumentation and justification on an issue in animal transgenesis. *International Journal of Science Education*, 23(9):903–927, 2001.

[18] Caroline Park, Cheryl Kier, and Kam Jugdev. Debate as a teaching strategy in online education: A case study. *Canadian Journal of Learning and Technology / La revue canadienne de l'apprentissage et de la technologie*, 37(3), oct 2011.

[19] Elizabeth Green Musselman. Using structured debate to achieve autonomous student discussion. *The History Teacher*, 37(3):335, may 2004.

[20] Jennifer C Richardson, PA Ertmer, JD Lehman, and TJ Newby. Using peer feedback in online discussions to improve critical thinking. In *Proceedings of The Annual Meeting of the Association for Educational Communications and Technology, Anaheim, CA*, 2007.

[21] Jennifer A Fredricks, Phyllis C Blumenfeld, and Alison H Paris. School engagement: Potential of the concept, state of the evidence. *Review of educational research*, 74(1):59–109, 2004.

[22] George D Kuh. What we're learning about student engagement from NSSE: Benchmarks for effective educational practices. *Change: The Magazine of Higher Learning*, 35(2):24–32, 2003.

[23] Hamish Coates. A model of online and general campus-based student engagement. *Assessment & Evaluation in Higher Education*, 32(2):121–141, 2007.

[24] Chin Choo Robinson and Hallett Hullinger. New benchmarks in higher education: Student engagement in online learning. *Journal of Education for Business*, 84(2):101–109, 2008.

[25] Khe Foon Hew. Promoting engagement in online courses: What strategies can we learn from three highly rated moocs. *British Journal of Educational Technology*, 47(2):320–341, 2016.

[26] Marcia D Dixson. Creating effective student engagement in online courses: What do students find engaging?. *Journal of the Scholarship of Teaching and Learning*, 10(2):1–13, 2010.

[27] Mark Klein. The MIT deliberatorium: Enabling large-scale deliberation about complex systemic problems. In *2011 International Conference on Collaboration Technologies and Systems (CTS)*. IEEE, may 2011.

[28] Ilse Verdiesen, Martijn Cligge, Jan Timmermans, Lennard Segers, Virginia Dignum, and Jeroen van den Hoven. Mood: Massive open online deliberation platform a practical application. In *1st Workshop on Ethics in the Design of Intelligent Agents*, pages 6–11, 2016.

[29] Jesse Graham, Brian A. Nosek, Jonathan Haidt, Ravi Iyer, Spassena Koleva, and Peter H. Ditto. Mapping the moral domain. *Journal of Personality and Social Psychology*, 101(2):366–385, 2011.

[30] Jeroen van den Hoven and Virginia Dignum. Moods: Massive open online deliberation.

[31] Argüman. `http://en.arguman.org/about`.

[32] Deen G Freelon, Travis Kriplean, Jonathan Morgan, W Lance Bennett, and Alan Borning. Facilitating diverse political engagement with the living voters guide. *Journal of Information Technology & Politics*, 9(3):279–297, 2012.

[33] Tim van Gelder. Cultivating deliberation for democracy. *Journal of Public Deliberation*, 8(1), 2012.

[34] Dai Clegg and Richard Barker. *Case method fast-track: a RAD approach*. Addison-Wesley Longman Publishing Co., Inc., 1994.

A

Project description

The provided project description differs from the original description at BepSys. Before the start of the project, the description is reformulated. We also provide a description of the company we worked for.

A.1. Reformulated project description

Recently, the rampant spread of biased/mis-information, political filter bubbles, and group polarization has become an urgent threat to the functioning of democracy, and with that to society. Take the Brexit referendum, the US presidential elections. In a trend where facts don't change beliefs anymore, also the scientific community is raising alarm on policymakers rejecting overwhelming evidence on the grounds of mischaracterizing science as a partisan issue, and hence "just another opinion".

Cultivating our ability to think critically can have a major impact to fight this trend. Becoming aware of our political bubble, biases, fallibility in our intuitions, detecting fallacies and misleading rhetoric, learning how to skeptically inquire into the validity of claims, deciding what are trustworthy sources. These are all valuable cognitive traits we need as a society to tackle pressing and highly complex problems such as climate change.

This project proposes the development of an extension of the FeedbackFruits platform to promote critical thinking. FeedbackFruits is a startup with the mission to improve education, and they have already developed an extensible platform to improve the learning experience of students. Here teachers can easily adopt new methods of teaching. , reaching thousands of students and teachers.

Through the use of argument mapping (see `en.arguman.org` as an example) we hope to develop an application that can be deployed in courses to start teaching some of the important aspects of critical thinking. To increase student engagement this can be done in a playful format of teams that can score rewards.

A.2. Company description

FeedbackFruits is a company with the main purpose of improving student engagement through technical innovation. This is done through a consortium, called the "Education do tank" where educational institutions are actively asked for didactic situations to be supported by FeedbackFruits. Some of FeedbackFruits most popular features are the peer feedback module, the interactive presentations and the emphasis on social interaction between students.

B

Project plan

This appendix gives an overview of the project. First, we describe the goal and motivation for the project. Then, we list our stakeholders. In section B.4 we give an overview of the schedule for this project. Next, we have a section that describes the team structure and process. In that section we assign some key roles, we describe how we are going to communicate and how we keep track of the planning. In the final section, we discuss how we are going to test and evaluate the developed application.

B.1. Motivation and goal

The spread of mis-information is getting an increasing problem. Due to group polarization and filter bubbles we are increasingly receiving one sided information. Additionally the scientific literature shows humans have a cognitive baggage of cognitive dispositions. They have biases to fit the evidence to already held belief, and dismiss conflicting beliefs (e.g. as 'fake news'). For the sake of dealing with misinformation in ones personal life (e.g. bogus health and nutrition claims), being a responsible professional (e.g. moral management decisions) and a functional democracy (e.g. moral and evidence based policy design decisions) education has a strong responsibility to cultivate *critical thinking*.

The goal of this project is to develop an extension of the FeedbackFruits platform to promote critical thinking. We approach this from the perspective of argument mapping to develop awareness of abductive reasoning and our natural vulnerability in domain.

B.2. Company and supervisors

The client of this project is the educational tech startup FeedbackFruits. The company is dedicated to improving education. It has been exploring various ed-tech solutions. These solutions aim to promote the easy adoption of modern didactic solutions by teachers, increase student engagement, and the cultivation of key cognitive traits like critical thinking. Because teacher adoption and student engagement have already been identified as the dominant challenges in product success by FeedbackFruits, significant emphasis will be dedicated to ensuring our method also succeeds here. The difference with typical bachelor projects is that this challenge puts a significant role of ensuring user experience and interaction design on our process.
Joost Verdoorn is our coach at FeedbackFruits. From the TU Delft we have Catholijn Jonker as a coach.

B.3. Constraints

This project must create an application that stimulates critical thinking. The application is designed for usage in courses, therefore the group size is limited. It must be easy for a docent to adopt the tool. To achieve this, it must only take a few steps to setup the application. Furthermore since student engagement influence the success of this application, we have to pay special attention to it. Finally when the project is finished it must be easy for FeedbackFruits to maintain the project.

B.4. Schedule

This section describes the schedule for the bachelor project. An overview of the schedule can be found in table B.1. A detailed planning for the production phase will be made in week two.

Week 1 [April 24 - 28]
The focus of the first week will be the literature research. First, we will research what factors influence critical thinking. We will examine existing products, with a similar goal as this project. Furthermore, we will identify what the challenges and problems are, for the type of application we want to make. Finally, we will investigate what the best approaches are for the challenges and problems.

Week 2 [May 1 - 5]
The goal of the second week is to create a detailed overview of the requirements. This is done with the MoSCoW method. Furthermore, we will investigate in tools and algorithms that will help us to implement the requirements. We will also make a planning for the production phase.

Week 3 - 6 [May 8 - Jun 2]
During this period, we want to create a usable version of the application. To achieve this all the 'must' and some 'should' requirements must be implemented. The first of June, the code is send to SIG for a code review.

Week 7 - 8 [Jun 5 - 16]
The goal of this period is increasing the user experiences. By doing usability testing we want to identify missing features. These features should be implemented in week 7 and 8. Week 8 is the final week we will work on the code.

Week 9 [Jun 19 - 23]
The focus of this week will be the report. During this week we will finish the report.

Week 10 [Jun 26 - 30]
On June 26th, the report and the code must be handed in. The rest of this week is used to create the presentation and to prepare a demo.

Week 11 [Jul 4]
On 3 or 4 July we will give the presentation, this will include a demo of our product.

Table B.1: Overview of the schedule. The rows in bold are dates where we have to hand in a deliverable.

When	What
Week 1 [April 24 - 28]	Literature research and competition analysis.
Week 2 [May 1 - 5]	Requirements and preparations for the production phase.
Week 3 - 6 [May 8 - Jun 2]	Production phase.
01-Jun	**Send code to SIG.**
Week 7 - 8 [Jun 5 - 16]	Usability testing and improve the application based on the results.
Week 9 [Jun 19 - 23]	Write report.
26-Jun	**Due code and report.**
Week 10 [Jun 26 - 30]	Presentation and demo preparation.
Jul 4	**Presentation and demo.**

B.5. Team process and structure
To make sure that everyone knows what is going on, we start each day with a short meeting. In that meeting we will discuss what has been done and what we are going to do. At the start of each week we will have a meeting in which we will discuss the goals of the coming week and in which we will evaluate if we are still on track. Approximately every two weeks we will have a meeting with our supervisors. Felix makes sure that for every meeting minutes are made. He does this by appointing someone as notary for that meeting. Boudewijn is responsible for keeping track of the planning. He should keep track over the overall process and check if everyone delivers his work on time.

For communication, we use Slack and WhatsApp. Project documents are stored on a shared Google Drive and the code is shared using GitHub. Documents that should be handed in, will be written on Overleaf.

B.6. Testing and evaluation

To ensure the quality of our product we will have to test it. First we will test it by writing unit tests. The back end should have a test coverage of 100%. Front end does not need unit tests. In addition to the test, we will also review each other's code. Changes that alter/add less than 50 lines should be reviewed by one person. Changes that alter more lines should be reviewed by two persons. Starting in week seven, we will do several usability tests. We will ask a group of students to perform a discussion with our application on a specific subject. Furthermore, we will also try to find a teacher that is willing to use our product during a lecture.

C

Product backlog

In this chapter, an overview of the functional requirements is given. Each requirement is is turned into a user story in order to differentiate in the different users and their permissions. To prioritize, the requirements have been ordered by a rank value, with a lower rank meaning more important. Most of the requirements are fulfilled; all the must haves have been done and almost all the should haves as well. Two should haves have been dropped. Assigning the role Evaluator to a student was dropped because we thought that this role would not be very useful during a debate. Decomposing an argument was not done because we decided that this feature had lower priority. Besides, decomposing could also be possible if someone adds a comment on an argument and the author splits argument himself. Some of the could haves are also done.

Table C.1: Product backlog as a list of user stories. A dropped user story means that requirement was decided not be a requirement anymore. User stories with a blank entry in the 'Done' column means that we have decided not to implement it during the project. These features can be implemented in the future, though, that is up to FeedbackFruits to decide. For our project it means that those features have become won't haves.

As a...	**I want to...**	**Importance (1 = very important)**	**Done**
	MUST HAVES		
teacher	create an argument map.	1	Yes
user	view the arguments in a tree structure.	1	Yes
user	add a supporting argument to a node.	1	Yes
user	add a rebuttal argument to a node.	1	Yes
teacher	give students access to an argument map.	2	Yes
teacher	have an overview of all maps. This overview should include information on how actively the map is used.	2	Yes
user	add sources/references to an argument.	2	Yes
user	edit an argument that I have made.	3	Yes
	SHOULD HAVES		
teacher	create random groups (with a given size) automatically.	10	Yes
user	be able to vote on arguments.	10	Yes
user	place comments on arguments.	11	Yes
user	get e-mail or browser notifications when an update occurs in a map .	12	Yes
teacher	set deadlines on maps	12	Yes
user	not be able to edit the map after the deadline.	12	Yes
user	get live updates of the map.	13	Yes
teacher	assign the role Arguer to a student. This can be a pro or con arguer.	14	Yes
teacher	assign roles automatically.	14	Yes

teacher	assign the role Evaluator to a student.	15	Dropped
user	be able to decompose an argument.	16	Dropped
	COULD HAVES		
user	propose an edit to any argument.	23	Yes
user	mention other users inside a comment.	23	
user	view the argument revision history.	24	Yes
user	refer to another argument node.	25	
teacher	assign the role Moderator to a student.	26	Yes
teacher	assign the role Questioner to a student.	26	Dropped
user	propose compromise nodes when multiple nodes disagree on some fact or value.	27	Dropped
user	propose consensus nodes when multiple nodes agree on some fact or value.	28	Yes
teacher	assign roles manually.	29	
user	I want to vote distinctly for personal preference and for moral acceptability.	30	
user	denote my values and specify my opinion of the relevance of that value for the argument.	31	
user	see the statistics of other users.	32	
user	flag logical fallacies.	33	
user	have access to an ad-hoc feature explanation on logical fallacies.	34	
user	evaluate the work of other students once the debate map deadline has expired.	35	
teacher	propose multiple theses so that each automatically created group can get a unique thesis.	35	
	WON'T HAVES		
student	provide my preferences before the roles are assigned.	40	
user	use keyboard shortcuts to use the tool.	41	
user	see the map auto-collapsed initially.	42	
user	see the arguments rearranged automatically.	42	
non-participating user	view the map.	43	
user	be informed about possible duplicate arguments.	44	
user	export the map to a file.	45	
user	generate a minimap of the entire map.	47	
user	rearrange arguments manually.	48	
mobile user	use the tool.	49	
user	post a small amount of arguments anonymously.	50	

D

Transfer

In this chapter we discuss the transfer of our project into the hands of FeedbackFruits. FeedbackFruits has made it clear from the beginning that continued development of our product is not feasible in the short term. However, they have expressed their willingness to maintain it indefinitely. This is split between the back-end engine (section D.1) and the front-end engine (section D.2).

D.1. Back end

The back end strictly follows the FeedbackFruits standards. We adhered to all the same requirements for the other features, even exceeding the test coverage of other very mission critical engines. Tests have been written to fully cover all important functions.

The area which may be most subject to change is the distribution of the groups. FeedbackFruits had no implementation for this and we were forced to write our own. We believe our implementation can serve as a good example how groups could be implemented and, should the necessary lessons be learned, help FeedbackFruits generalize our structure. Yet, this might require FeedbackFruits to make some changes to it.

D.2. Front end

Our front end is a slightly different story. FeedbackFruits uses Ember components in separate addons to ensure separation of concerns. We created the debate map project in a more powerful Ember engine. FeedbackFruits has expressed desires to eventually migrate all its addons to Ember engines, but cannot allocate the resources in the near future. Our decision to use an Ember engine allowed us to work more quickly, while depending less on the FeedbackFruits platform. We believe no major changes will be necessary to maintain DebateMap as an Ember engine, in fact, it might serve as a good example of the necessary changes for migrating the addons to engines.

The independence and flexibility we gained by using an engine meant we could deviate from the strict framework set by the FeedbackFruits platform where convenient. We did this in a manner more similar to how Ember is supposed to be used. This meant we are rather confident that DebateMap should be more maintainable and less sensitive to bugs than the rest of the FeedbackFruits platform, save for some of the integrations we might have overlooked.

D.3. Unresolved issues

All of the bugs found during testing have been listed on the GitHub repository. These issues have been labeled by priority and type, differentiating between bugs and interaction issues (some of which are feature requests). We advice looking at these to get an idea of the most prudent issues to be resolved.

E

Product

In this chapter we give an overview of the most important features of our product. First, the core argument mapping features are highlighted. Next, we will show the features that can be used to improve the arguments. Finally, we show the assignment overview for both the teacher and student.

E.1. Argument map

The core of our application is the argument map (figure E.1). This map is presented as a tree of arguments. Users can add an objection or a supporting argument to the thesis or an other argument (figure E.2). Arguments can be supported by sources (figure E.3). Besides objections and supporting arguments, we also have consensus nodes. These nodes summarize parts of the debate and can contain conclusions (figures E.4, E.5).

E.2. Interaction

We have created several ways for users to interact with each other. The goal of this interaction is to improve the arguments that are made. This can be done by upvoting arguments, placing comments on arguments (figure E.6), as well as by submitting an edit proposal (figure E.7). In such a proposal the argument itself can be reformulated or sources can be added.

E.3. Overview

Teacher

For the teacher we created a page that gives an overview of all the debate groups in the current assignment (figure E.11). We list some overall statistics and the statistics per map. In the overall statistics we show how many students have started and on average how many arguments are made, how much time has been spent and how many moderations were done. For the different argument maps, we show the total number of arguments, the total number of moderations and the number of participants. This allows the teacher to see which groups have started and which groups might need some extra attention. It is also possible to view the details of the participants of a specific map (figure E.12). The detailed overview contains for each user the role that was assigned, as well as the amount of time spent in the debate and the amount of arguments, comments and moderations.

Student

In the student overview, all the participants of the specific argument map are listed (figures E.8, E.10). The participants are grouped by role. Furthermore, the student can find information about the different roles. For each role there is a tool-tip that explains what a user with that role is supposed to do and what his privileges are.

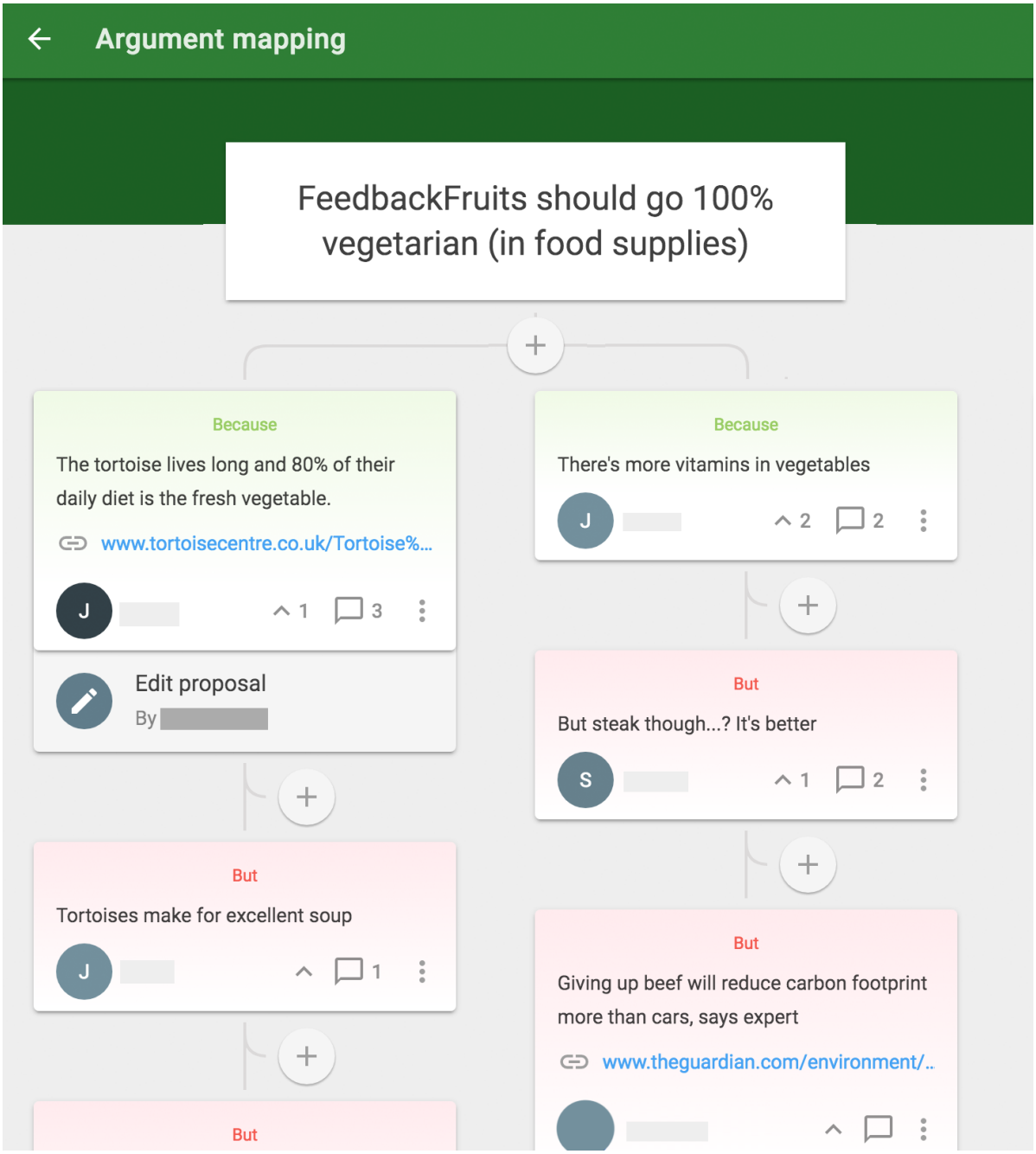

Figure E.1: An example argument map

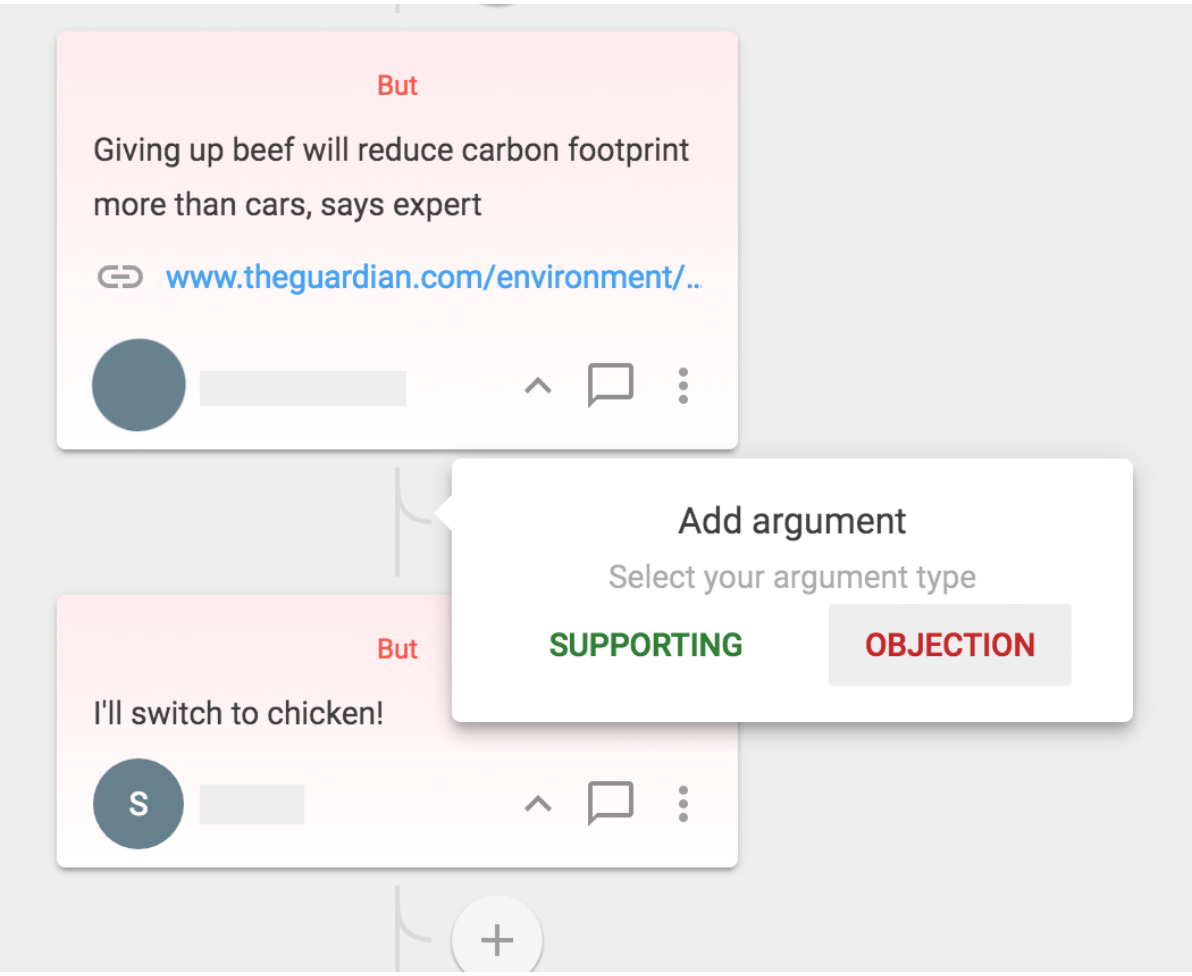

Figure E.2: New arguments can be inserted in response to any argument

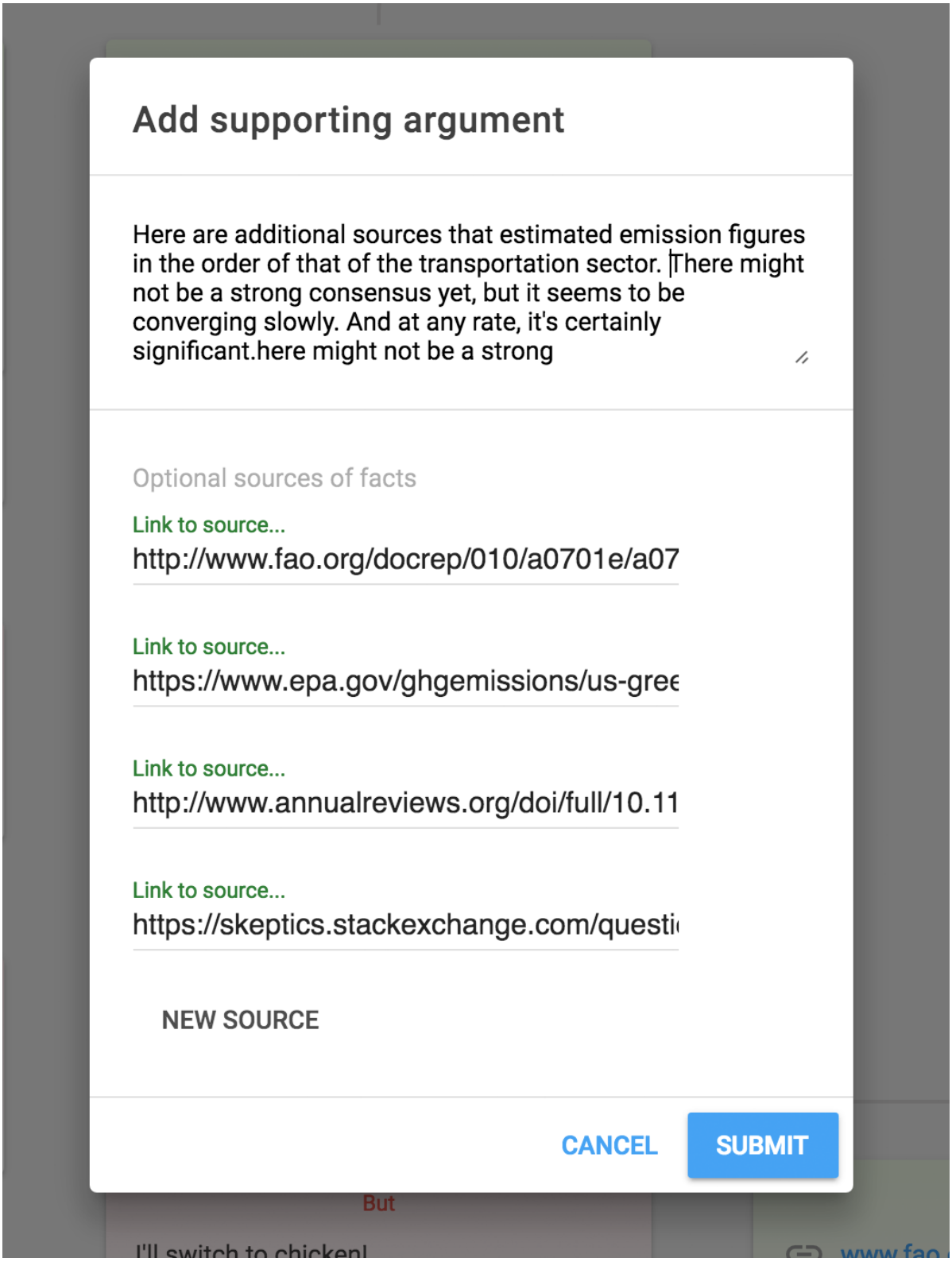

Figure E.3: New arguments can also cite sources

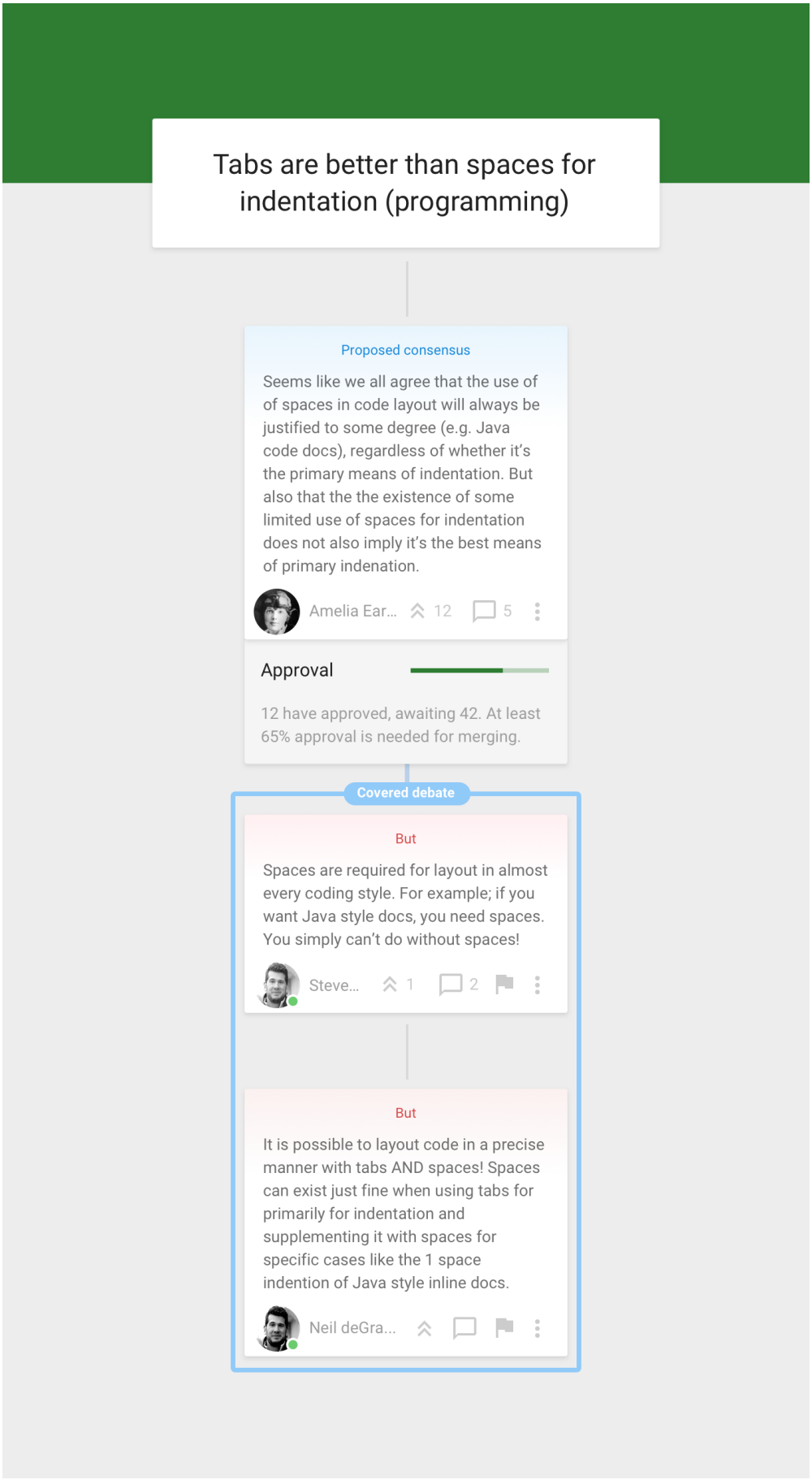

Figure E.4: Example of a proposed consensus on an argumentation branch

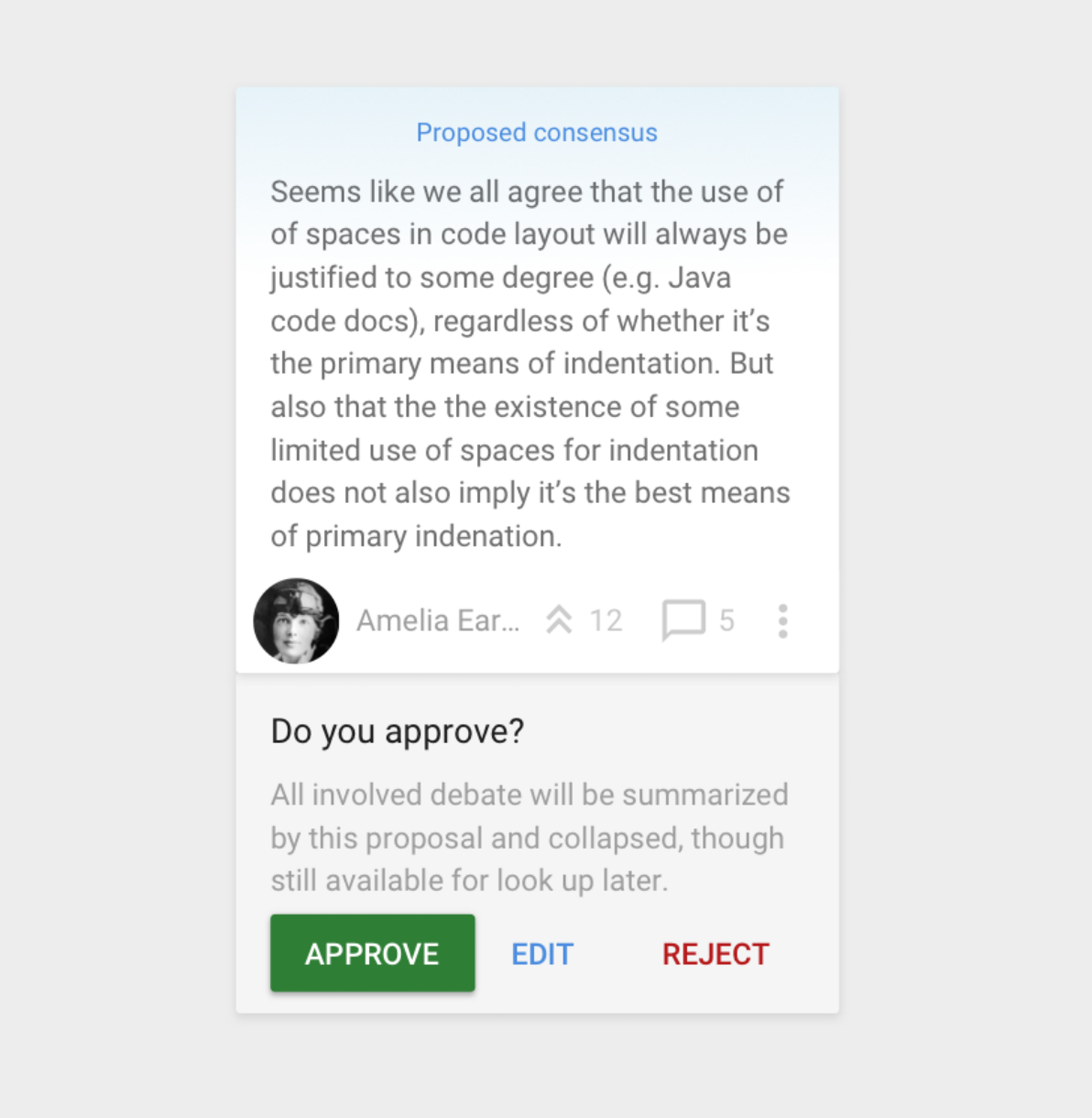

Figure E.5: A consensus proposal for a participant perspective. To ensure that no expression gets suppressed, all participants involved in the argumentation branch with a consensus proposal are asked for approval. When a configurable threshold of agreement is reached, the argumentation branch gets collapsed, available for lookup later.

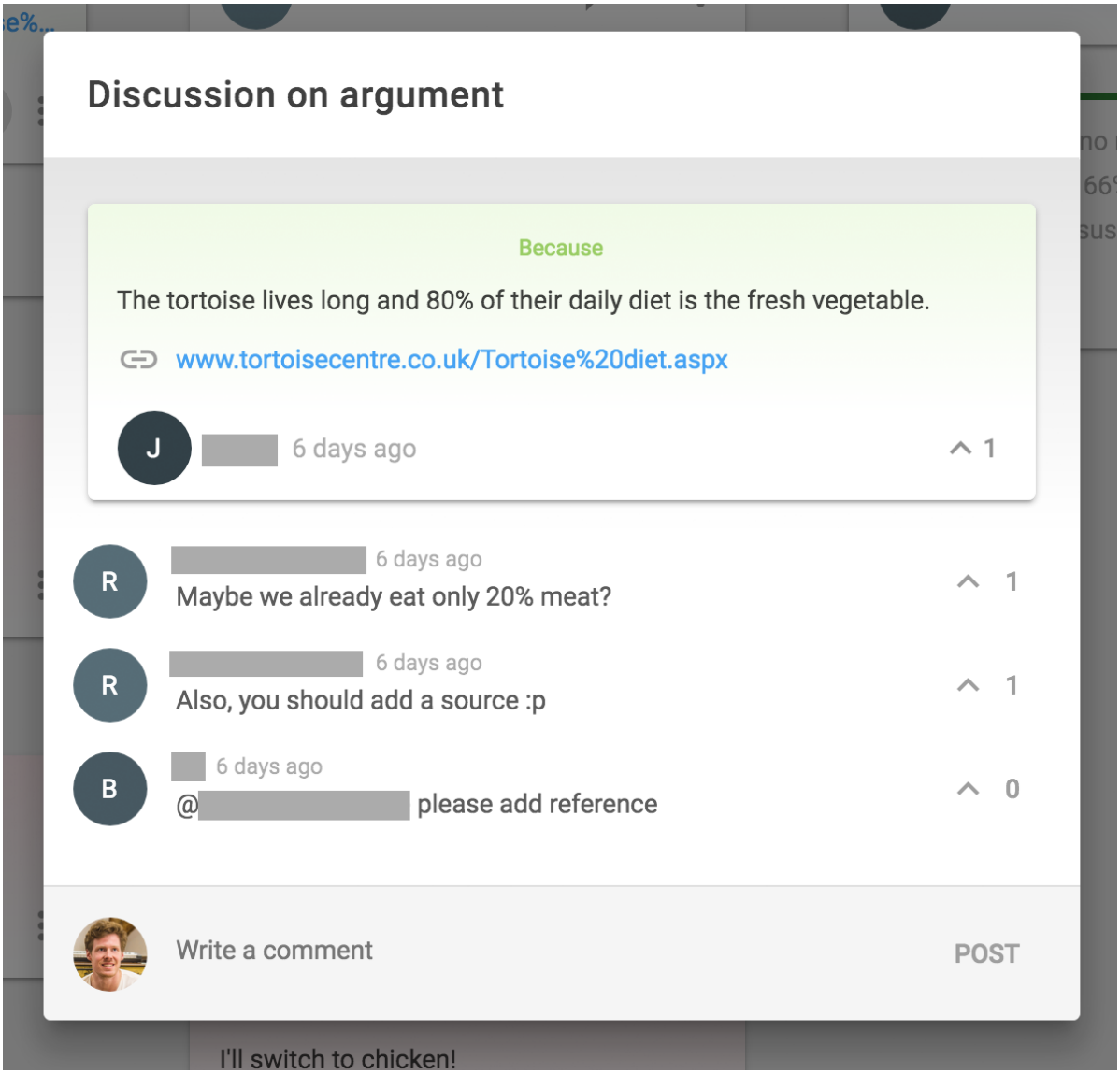

Figure E.6: Example of discussion on an argument. This dialog is opened when clicking on the argument text or the comment button.

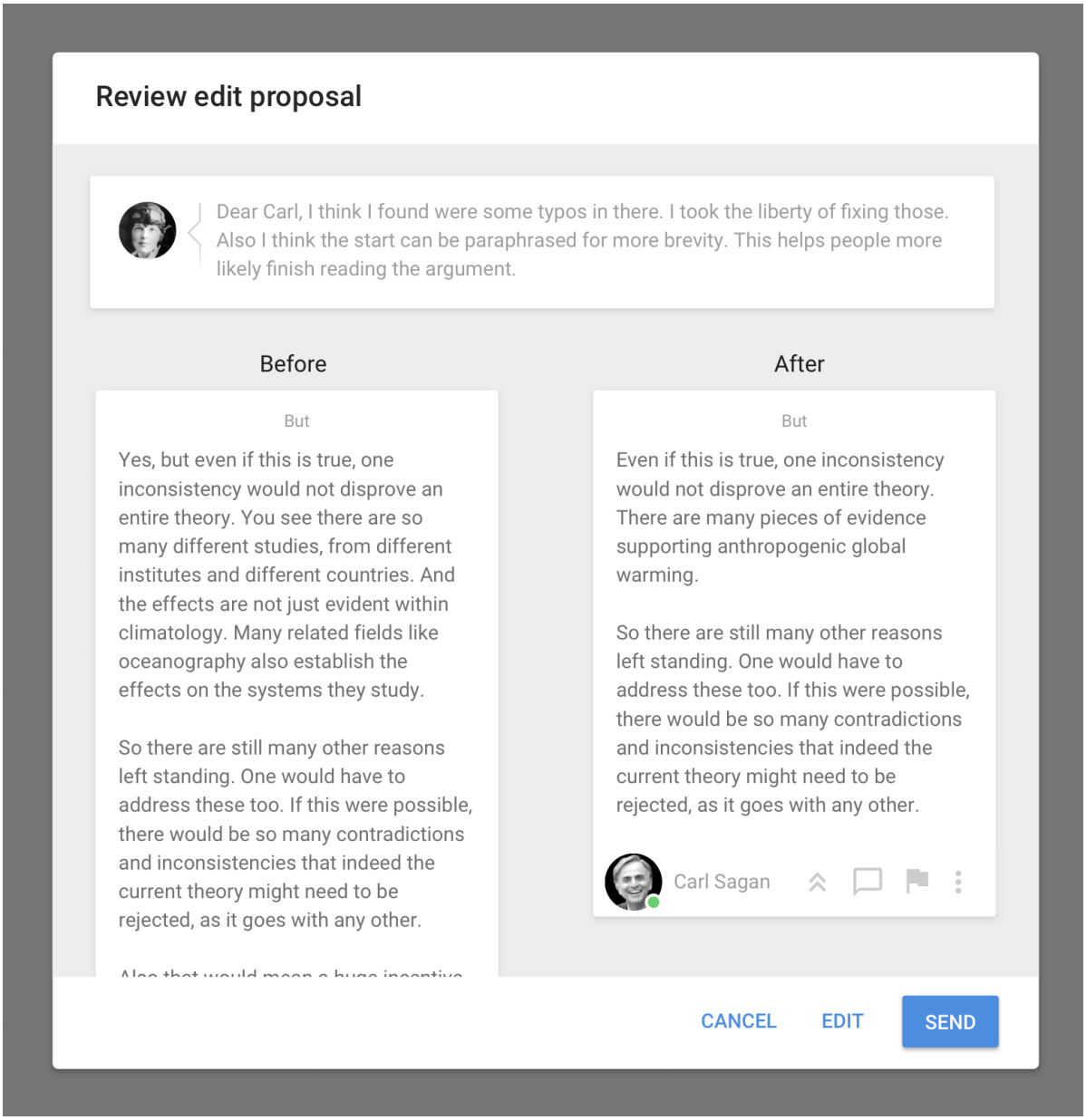

Figure E.7: Example of someone proposing an edit on an existing argument.

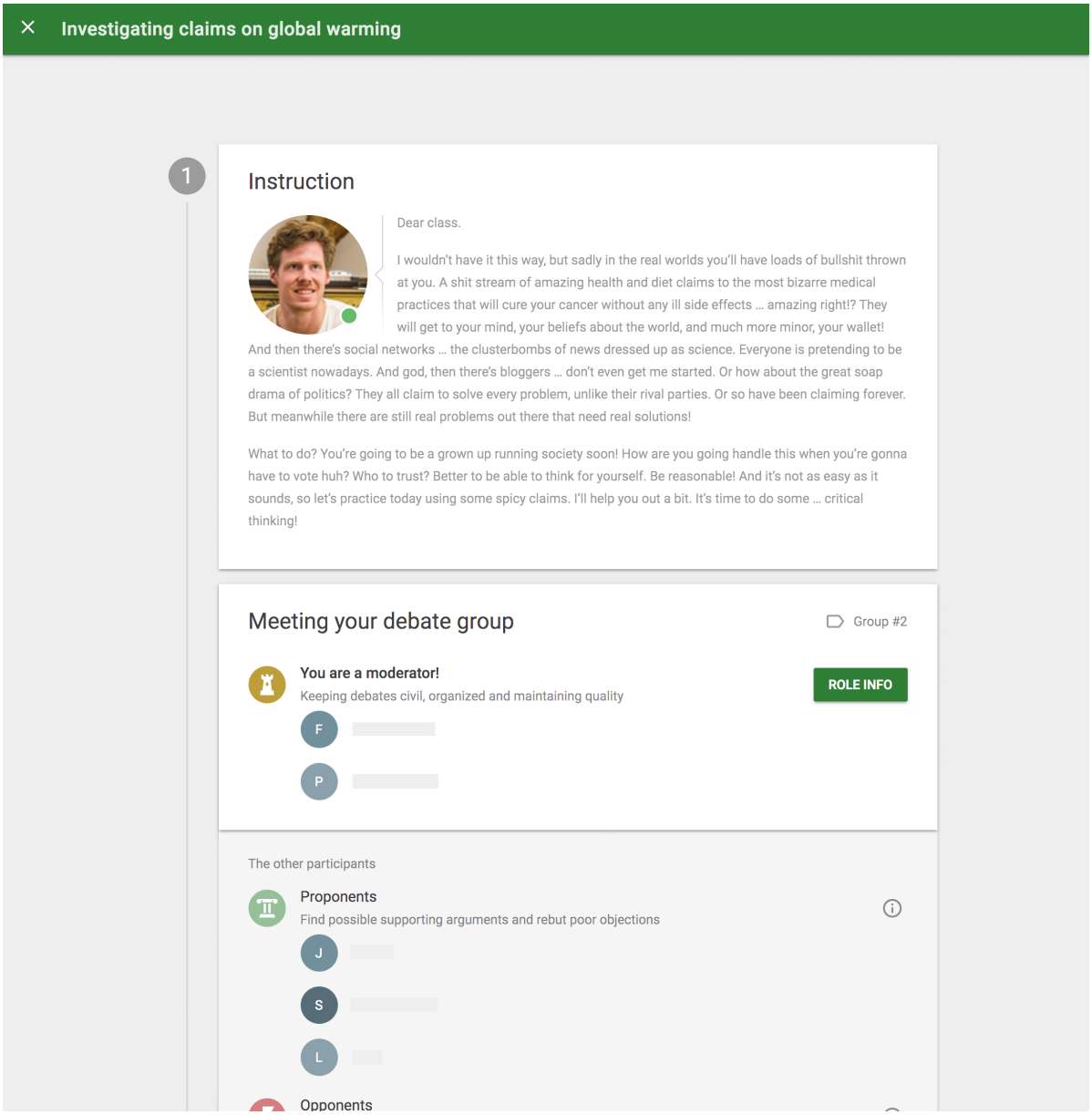

Figure E.8: Example of student viewing his assigned role, as well as those of his classmates.

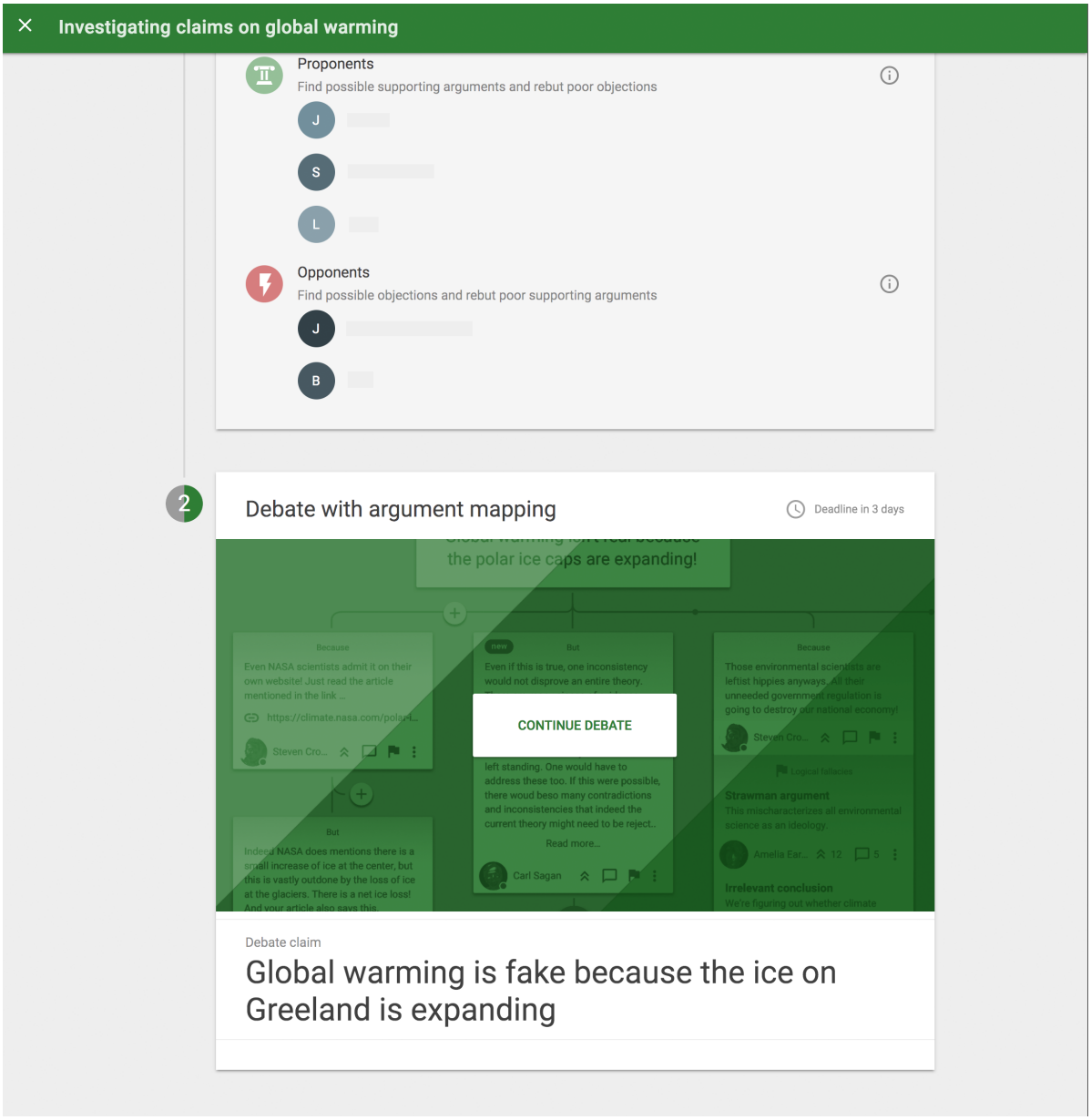

Figure E.9: Example of student viewing the rest of the assigned roles and the entry portal to the argument map.

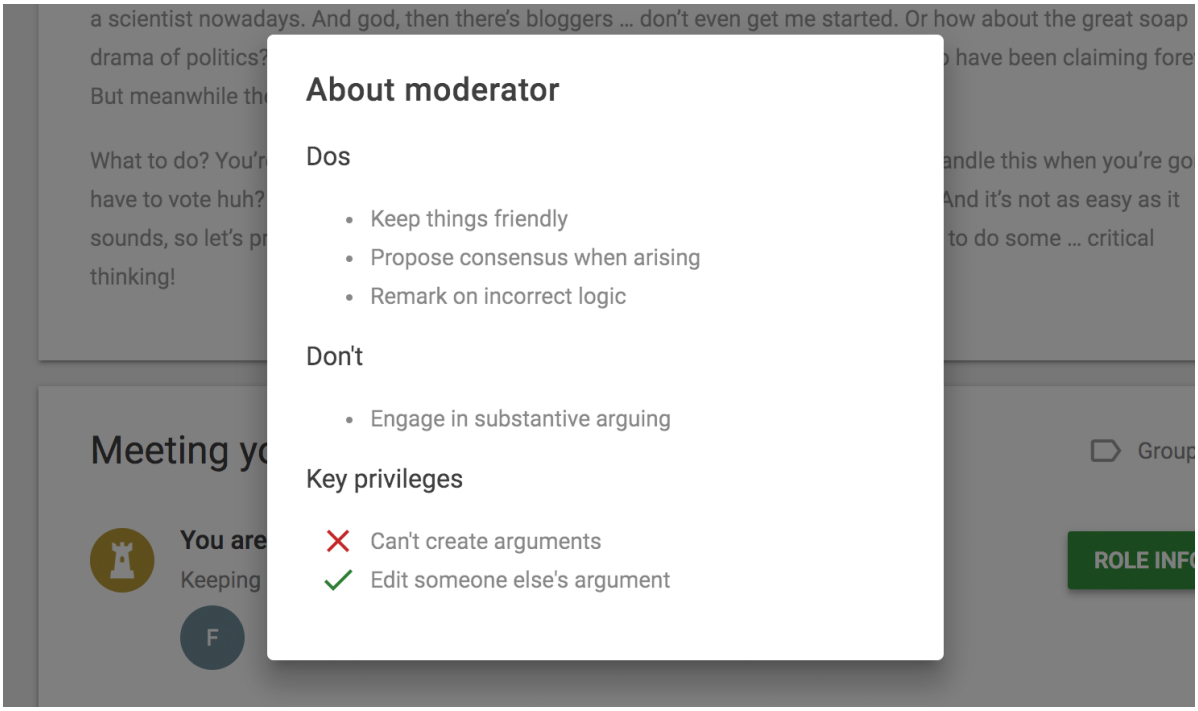

Figure E.10: Example of student viewing additional info about his/her role.

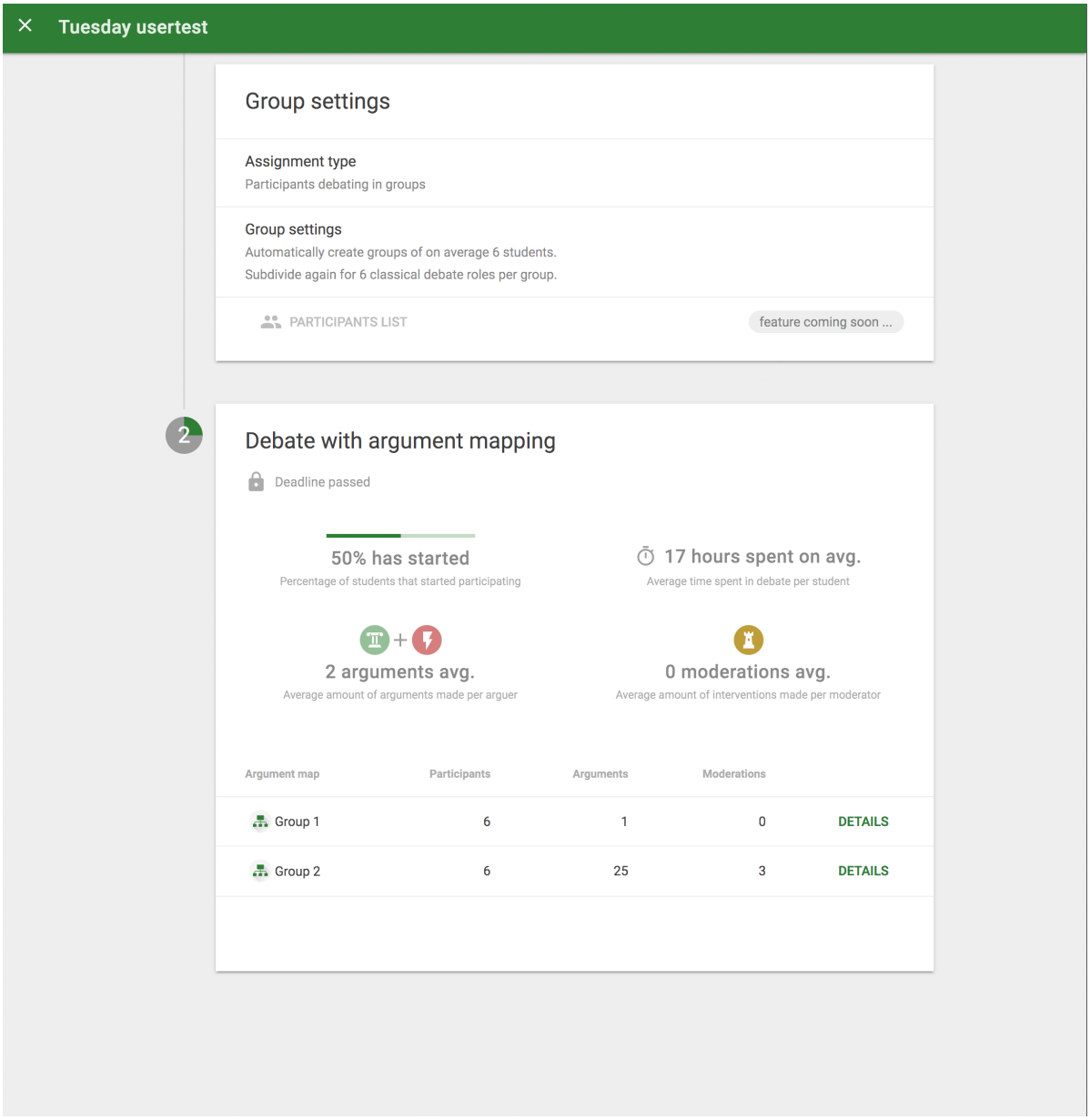

Figure E.11: Example of the teacher perspective on the assignment, showing the debate groups and some basic analytics. Additional details on a specific group can be queried using the 'details' button.

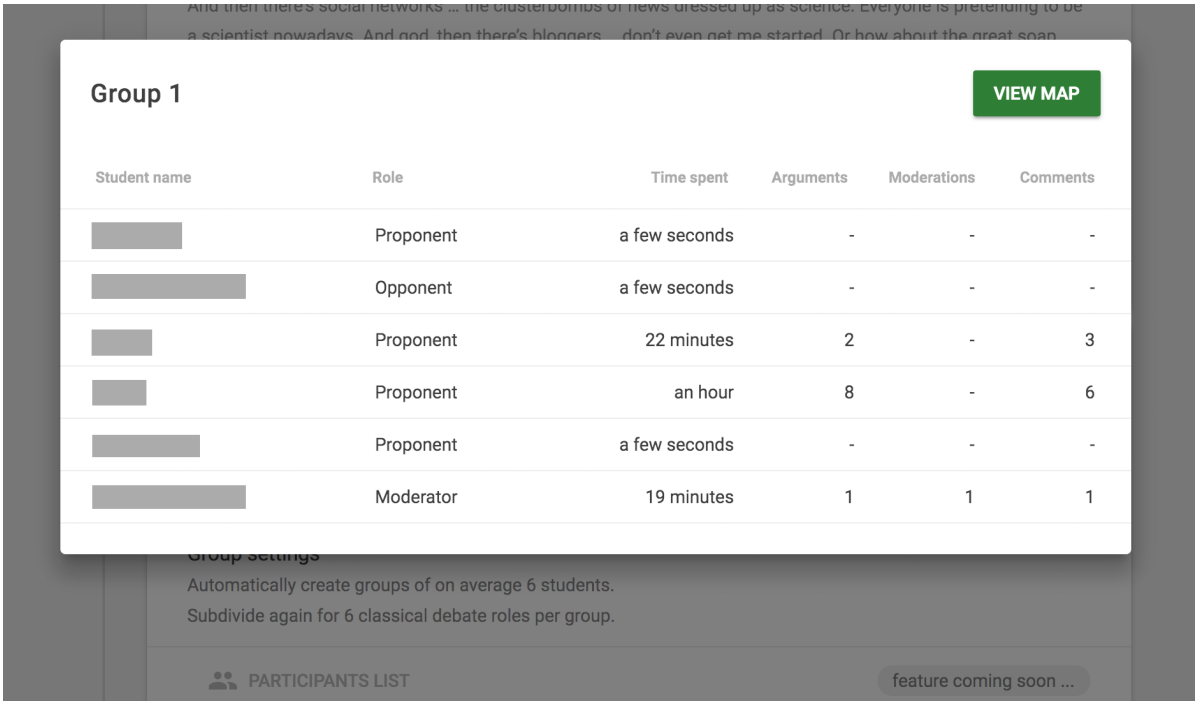

Figure E.12: Example of teaching viewing detail about a specific debate group.

Infosheet

Title of the project: Teaching critical thinking through argument mapping
Name of the client organization: FeedbackFruits
Date of the final presentation: July 4, 2017

Description

FeedbackFruits is a company with the main purpose of improving student engagement through technical innovation. Our project goal was to extend the FeedbackFruits platform with an argument mapping tool to improve a student's critical thinking skills.
The core challenge for the success of this project was thinking of a way to get students enthusiastic about critical thinking, hence willing to participate. By creating a simple and intuitive debating environment and enforcing the use of roles during a debate, this should be accomplished. User tests with members of FeedbackFruits already showed promising results.
The most important lesson we learned during the research phase was that the use of moderations can really help improving the quality of online debates. This finding made us focus on thinking extensively how to implement the moderator and its tasks properly, as this is one of the main factors for the success of the project.
During the project the Scrum approach was used in order to have control on the tasks each group member was working on. Each week a sprint meeting was performed in which was reflected on the previous sprint and was discussed what had to be done the next sprint. Each morning there was a stand-up to keep track of what everyone was doing and what their issues were.
The final product is a argument mapping tool incorporated in the FeedbackFruits platform which can be used to help students to think critically in a educational manner. Features like commenting on arguments and proposing edits or consensuses add to improving students' skills to more think about arguments they make. More research and testing is needed to actually evaluate the effectiveness of our tool in improving education.
To improve the tool we recommended to investigate the size of the debate groups. Other recommended improvements are adding the ability to flag logical fallacies, evaluating and grading students' contributions, and allowing the students to more express themselves by involving values and emotions.

Members of the project team

Name	Interests	Contributions and role
Felix Akkermans	Critical thinking, product design and strategy, assuring user experience quality	External appointment secretary, front end designs, front end developer
Robin Borst	Building nice features, testing and improving user experience	Front end developer
Boudewijn van Groos	Organizing teams, thinking on the business side of things and solving technical challenges	Appointment president, back end developer, front end developer
Jens Langerak	Web development, user interaction	Back end developer, front end developer

All team members contributed to making notes at appointments and preparing the report and the final project presentation.

Client and Coach

Name client: Joost Verdoorn, employee at FeedbackFruits.
Name and affiliation coach: Catholijn Jonker, member of Interactive Intelligence Group at TU Delft.

The final report for this project can be found at: `http://repository.tudelft.nl`. Any questions can be sent to boudewijn@vangroos.nl