SOFTWARE REQUIREMENTS SPECIFICATION

for

StackOverflow4UC

Version 1.0

Authors : Alexy de Almeida (2019192123)
Edgar Duarte (2019216077)
Rodrigo Ferreira (2019220060)
Sofia Neves (2019220082)
Tatiana Almeida (2019219581)

Customer : Duarte Meneses
NEI informatics department coordinator

Professor: Mário Zenha-Rela

Team Contact: tsalmeida@student.dei.uc.pt

January 23, 2023

Revision History

Revision	Date	Author(s)	Description
0.1	05.10.22	All authors	Creation and initial structure of document
0.2	11.10.22	All authors	D1 - Writing Problem Statement and State of Art
0.3	20.10.22	All authors	Correction of the document structure
0.4	28.10.22	All authors	D2 - Writing of Contextual Design and corrections based on feedback
0.5	13.11.22	All authors	D3 - Writing of Use Cases
0.6	25.11.22	All authors	D4 - Goal Modelling
0.7	14.11.22	All authors	D5 - User Interaction
0.8	27.12.22	All authors	D6 - Non-functional Issues
1.0	23.01.23	All authors	D7 - Final Delivery

Contents

1 Problem Statement

1.1 Software Requirements Specification project

AAC (Academic Association of Coimbra) is an association that represents University of Coimbra's students. Their main goal is to integrate students into the academy, encouraging them not only to create lasting relationships with other students but also with the faculty members. They also count with a wide variety of projects, concerning different areas like sports and politics. AAC counts with multiple student nucleus that represent their respective departments. NEI (Núcleo de Engenharia Informática) is the AAC's representative in this project, an online platform where members of the UC community can ask/answer questions and share materials through a common repository.

1.2 StackOverflow4UC

StackOverflow4UC is a platform aimed to create a network of shared knowledge that can be accessed by anyone in the UC environment when desired.

The problem to be solved comes from the University of Coimbra's community, namely students, professors and alumni, who need aiding in:

- having more information regarding university projects, assignments or any other important information;
- having a single source of truth;
- getting their specific questions answered, which are not found easily.

There are four different types of stakeholders that play an important role to this problem:

- the university's students;
- the university's professors;
- alumni - former university students;

- the dean of the university and its intermediaries.

This is a problem that needs to be solved because it is an ongoing issue among all of the UC community. There is a lack of resources, such as: papers, articles, exams, questions and so on. All of that makes studying at the University of Coimbra harder than if those resources were up and available for public use of the UC community.

The client's intention is to have a place where the UC community can have their questions answered and check previously asked questions. A software system platform would allow these students to ask their questions and get a response in return. Moreover, it would gather knowledge, in a single place, from all of the university's courses. This would also be useful for future reading of incoming new students.

For the next three months we'll explore and propose requirements with a possible solution. However, the client expects a solution as soon as possible, but there isn't yet a specific deadline. The main risk resides in not having access to/knowing the necessary information for making the product work. Even though the information on our platform would belong to AAC, the platform might face legal constraints imposed by the University of Coimbra, specially regarding plagiarism. Furthermore, UC has procedures and rules to detect plagiarism in works submitted by the community. This way, students might get apprehensive in using the product, as most of them don't fully know the rules and might not want to get involved with the risks associated with it. This is an obstacle that might prevent us from solving the main problem.

2 State of art

This state of art will make an overview of already existing successful and failed solutions related to the problem. In addition, it will address the way our project harbors some of the stated features of other successful ideas, while also adding pioneering features to this field of study.

2.1 Background

In this section we will provide an overview of existing solutions. These solutions will help us understand what has been previously done, as well as to learn from previous mistakes.

In order to find these solutions, as a search criteria, we looked for platforms with a community that contained forums and/or Q&A elements. The most important thing with this search is the social aspect and exchange of ideas, considering it is the heart of the software.

2.2 Successful solutions

2.2.1 Stack Overflow

This website consists of a place where people can ask and answer questions[1]. Its target audience is programmers.

What distinguishes this from other platforms is that:

- Through membership and active participation you can use a system of upvoting and downvoting for added credibility;
- The ability to use tags;
- It has an outstanding amount of users (over 14 Million registered users);

- It has over 21 Million questions and over 31 Million answers.

2.2.2 Uniarea

This is a website that has a forum very similar to stack overflow - people ask and answer questions[2]. However, the main difference is that it helps students from all kinds of different degrees of education - middle school, high school and university, although its main focus is the second one.

When comparing this platform to others, the main differences are:

- The possibility to upload files in response to the question;
- The possibility to download files;
- The fact that it only focuses on the portuguese students makes it more accessible to them.

2.2.3 Bloomfire

Bloomfire[3] is a software application that allows users to create communities. Within the communities, people can post questions and answers, create new content or browse for an existing one. The software's goal is to increase accessibility to information inside a company so that employees can have the knowledge they need to work efficiently.

Bloomfire differs from the other platforms because:

- It's mainly designed to be used by the companies' employees inside their workspace;
- Supports 53 file types;
- Content can be uploaded in the form of videos, audio files, images, slide decks or text documents;
- Has automatic video and audio transcription capabilities and makes the text of its transcripts searchable.

2.2.4 Reddit

Reddit[4] is a social news website where people come together to share their questions and thoughts. Users are allowed to create *subreddits*, niche communities where people talk

about specific topics. Posts inside of each *subreddit* can have increased visibility through voting by members. This software is highly popular, mainly, due to its free speech policy and diverse communities.

Compared to other platforms, Reddit offers:

- Video, file, image and text sharing;
- The possibility to create new sub communities, which allows the existence of super niche groups;
- Unique post voting mechanism;
- Good moderation, each sub community can assign a team of moderators to ensure an adequate environment for that niche.
- It's a famous platform used by people from all around the world.

2.2.5 Quora

Quora is a question-and-answer website to gain and share knowledge, where people can contribute with their unique insights. This is a particularly important platform because of its reward social aspect and reward system, Quora+, which works similarly to Google Answers: a subscriber pays a fee monthly or yearly to access content that any creator puts behind a paywall instead of paying for a single answer.

2.3 Failed solutions

2.3.1 Yahoo! Answers

Yahoo! Answers[5] was a community-driven question-and-answer website. The users could ask and answer questions about different topics that were subdivided in categories. The answers were ordered by upvotes, allowing the best answers to get more visibility.

The platform was shut down on April 2021. The platform failed due to several reasons, such as, not being a user friendly platform, most questions asked were confusing or irrelevant, as well as unasked for answers, reasons which lead the users away.

2.3.2 Google Answers

Google Answers[6] was a service offered by Google that allowed users to provide some questions and pay for it to come up with an answer. Several reasons lead to its failure, such as having a small community and the competition being, at the time, on the side of Yahoo! Answers. But the main reason for its failure was the service model, which required people to pay for information when they could easily find it online, for free. Not only that, but there wasn't enough data to provide correct and accurate answers, as well as being dependent on a certain number of researchers to provide those answers.

2.4 Objectives

Our intention for this project is to be able to join the best features from the 'successful solutions' section as well as add new pioneering features in this field. Some of the features our project will put in use are:

- Anonymity - Being able to question and answer anonymously;
- File upload and download - Being able to upload and download files to answer/question someone.
- File diversity - Being able to support multiple file types;
- Tags - The ability to categorize different types of content;
- Git, Dropbox and Google drive integration - The ability to select files from repositories/folders on cloud-based storage and upload them to an answer;
- Reply of reply - Being able to reply to an answer's reply (thread system);
- Report of obsoletion - Some answers might become obsolete over time. For this reason, we determined that there are two types: (i) Legacy system (incorrect only in certain contexts e.g. Answer only valid for Java version 2.0.0); (ii) Invalid system (over time the answer is now wrong);
- Report system - It will allow users to report questions and answers that they consider nonsensical or offensive. After a certain number of reports (strikes), the user will lose the ability to stay anonymous and may get their account permanently banned after moderator approval.
- Reward and punishment system - Users are compensated by giving good answers as well punishing users for misbehaviour;

- uc.pt domain names restriction - Limit the use of the platform only to registered users with uc.pt domain names;

All of the features combined would, in theory, make it possible for the end user to be able to have a single source of truth in UC.

Additionally, it will allow the users to have a better experience where they feel at ease to make questions or even give answers. We expect this platform to create a shared knowledge between members of the UC community, allowing them to easily get access to detailed academic information.

3 Contextual Design

In this chapter, we will present the different models of contextual design that we found relevant for our project.

For the construction of the models, we collected information from various members of the UC community from different departments of the university in order to find out how information is shared.

We interviewed 5 students: R. Lemos from Biology, S. Batista from Chemical Engineering, D. Meneses from Informatics Engineering, M. Lopes from Medicine and M. Melo from Management. We also interviewed PhD student L. Duarte (who is also a professor) from Mechanical Engineering and ex-student R. Casaleiro from Informatics Engineering.

3.1 Flow Model

The flow model is a graphical representation of how both information and artifacts flow through the system as it is used. [10]

The responsibilities of each individual are:

- Student: Their main responsibility is to acquire knowledge in the field they are studying for. In order to achieve that, they might need to study for exams/tests, deliver assignments or prepare for practical classes.

- Professor: Their main responsibility is to promote learning. It can be done with, or without, preparing classes, helping students, preparing course assignments, writing exams/tests or grading students assignments/tests/exam.

An interview was held with the grade coordinator of the LEI course in which there was nothing of value to be added to the field. For this reason, both the roles of the grade coordinator and the department head were not added.

We can see some breakdowns in the model below, such as:

1. The information is only shared amongst a few entities, leaving others unaware of the answer to the problem.

2. The information is not stored for future consultation.
3. The entity does not have the answer to the question.
4. The answer to the question is wrong.
5. The answer to the question is outdated.
6. Lack of availability to answer the question.

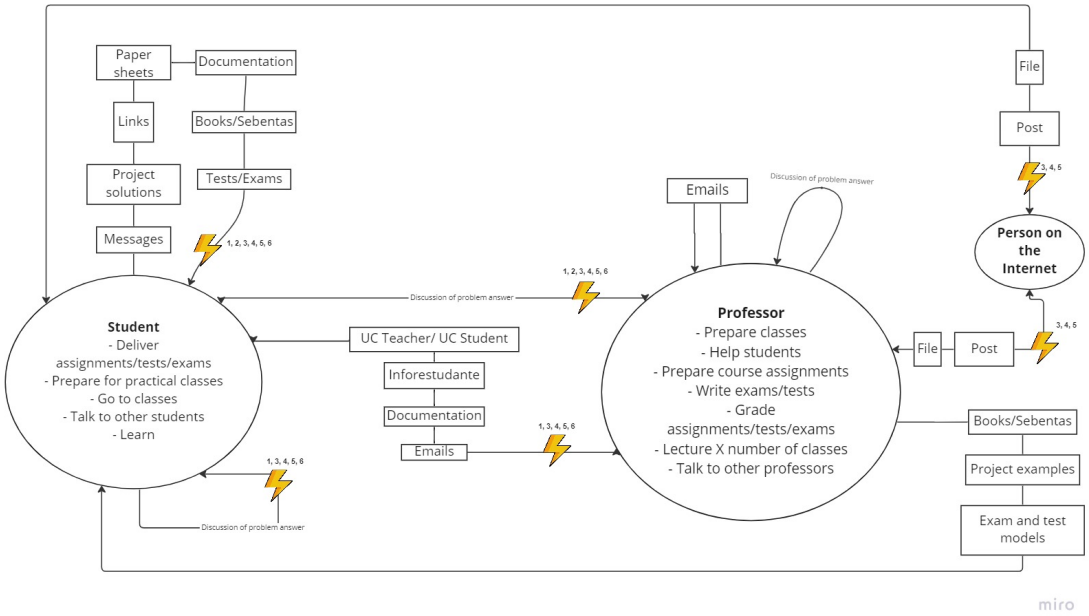

Figure 3.1: Flow model

3.2 Sequence Model

The user usually follows the same steps to look up for information. They go for their most comfortable source of information (a professor, a friend, the internet, etc) and, based on the information received, they either continue their research or simply conclude the process, accepting the information received as enough to form an answer or giving up, thus ending the sequence.

There isn't a specific order by which the user looks up the information. They can either go first to a teacher (1.1) or to a colleague (1.3) and, if the answer provided isn't good enough (3.2), they go back to the same or other sources (1.1, 1.2 or 1.3).

Some breakdowns were identified at the end of the process. When the user accepts all the information provided by the sources they went looking up (3.1), the final answer constituted by that information might be incorrect, misleading and so on. Additionally, if the user doesn't yet accept the answer and continues looking up for more information (3.2), they might risk wasting a lot of time. There is also a possibility where the student gives up on getting answers to a question (3.3).

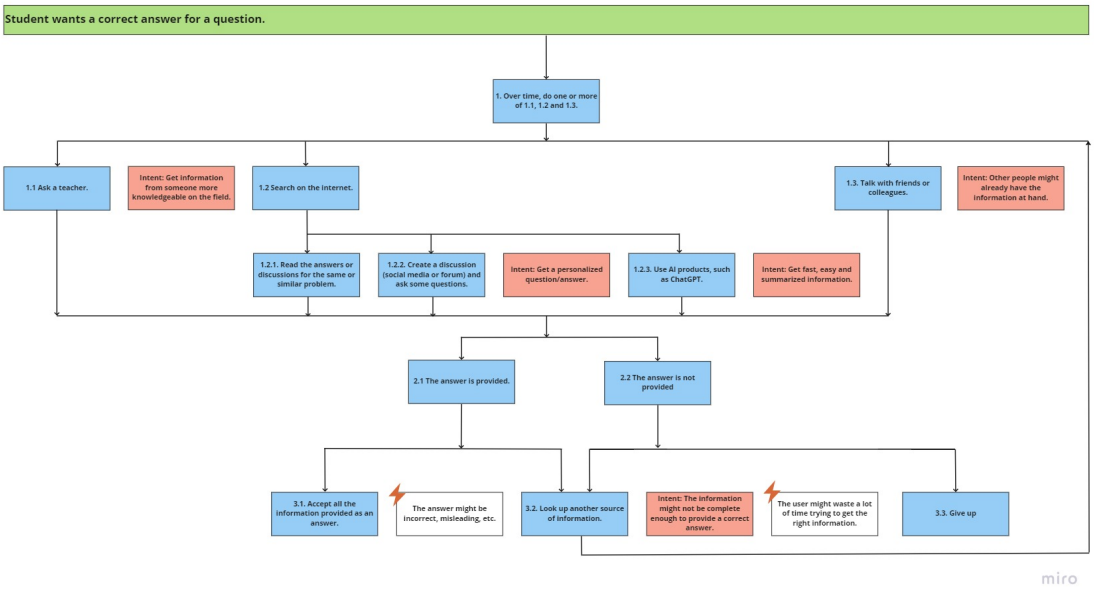

Figure 3.2: Sequence model of a student who wants a correct answer for a question.

3.3 Artifact Model

The artifact model consists of analyzing artifacts used in the flow of information, in order to obtain a greater perception of how things work in reality. As the sharing of information between students at the university of Coimbra is a problem that covers several types of artifacts (forums/emails/*sebentas*), we considered the Stack Overflow (an artifact used by computer engineering and mechanical engineering interviewees) a very relevant artifact to be analyzed in more detail.

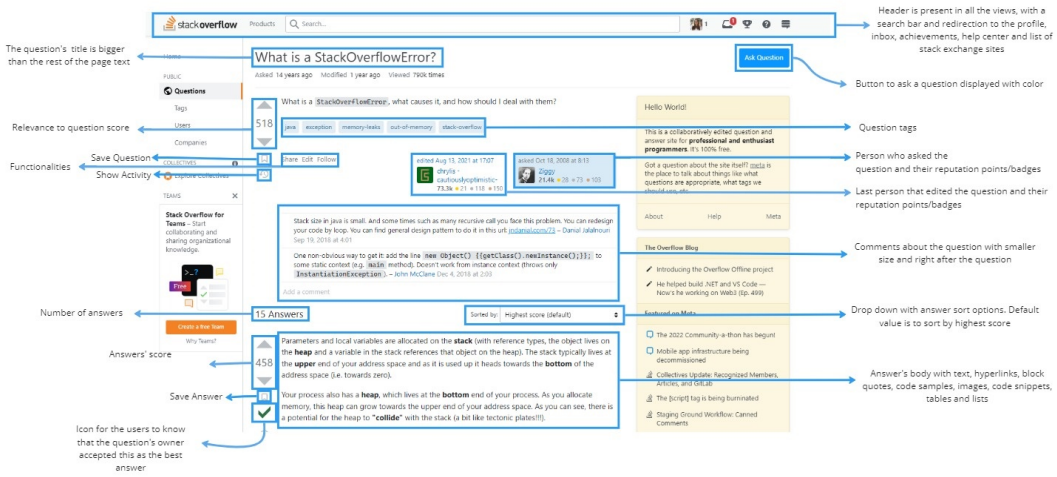

Figure 3.3: Artifact model - Stack Overflow

In the Stack Overflow example, there are several components that allow us to draw conclusions about which aspects are considered most relevant. On all pages of the site there is a Header consisting of a search bar, which allows users to easily search for other types of information at any time. Access to the profile, inbox and achievements are also components found in the header that reveal the importance given to the user and their contribution to the site.

A blue button is placed in the upper right corner so that the user can ask a question. This makes the process much faster if they realize that the question they are currently viewing does not match what they are looking for. The page is basically in black and white, but this button is given a blue color, highlighting its importance.

Focusing on the page of a question, we can see that the title appears at the top with some prominence, followed by the body of the question and its respective tags. We can note that the tags are highlighted with color and that they are clickable. Each of them redirects to a page with questions that have the same tag assigned. This type of behavior reveals the importance of the tag system for the organization and orientation within the site.

The question also has a voting system (score) that is displayed on its left side, which that translates into how much the community found the question relevant. The number representing the relevance appears in a larger font size than the question description and is associated with up and down arrows, making this feature gain some emphasis on the page.

There is also some emphasis on the person who asked the question, showing not only the date the question was asked, but also the photo, nickname, reputation points and

achievements they have. All the information that appears for the person asking the question also appears for the person who last edited it.

Close to the body of the question, we have some buttons that allow functionalities such as share, edit the question, or even follow it. In addition, there is a field, with a letter smaller than the one used for the body of the question, where the comments to the question can be found.

Regarding the answers, there is an indication of the number of answers given to that specific question and a filter to change the order of the answers, where the default is ordered by the highest score.

In the example, we can see that the response body is just text. However, when writing an answer, we can enter text with different formatting, hyperlinks, block quotes, code samples, images, code snippets, tables and lists. This type of functionality was made taking into account that the community using stack overflow are programmers

All answers are accompanied by their score and may or may not contain comments. There is also the possibility that a certain icon will appear below the answer's score, which symbolizes that the question's creator considered it to be the best answer, thus giving, in some way, greater relevance to the question creator's opinion regarding the evaluation of the answers.

3.4 Cultural Model

The cultural model is used to better understand the different cultures at play in the current model of sharing information, as well as the conflicts that exist between cultures. This model demonstrates the impact each role has on the sharing of information.

- Student: The students usually get materials from their professors or from online repositories where materials are shared by other students, but some students might go to the repography, as a last ditch effort, to get resources shared by previous students (even though it isn't free). For any questions the student may have, they either ask their friends, other students or professors. For some students, going to the professor is a last case scenario, seeing that they are afraid of being judged (and maybe penalized) if the question is considered "dumb". As the most accessible group, friends are usually relied on when a student has questions, even though they might be the one of the most unreliable sources of information. The students can also ask questions associated to the institution they are in and questions about their academic life, which might be answered by their department/university.

- Professor: Professors are, in most cases, the most accurate source of information that a student can get. When asked a question, they are very likely to be available

to answer. Even so, their status can frighten many students from ever asking them questions. The quality of the classes and support documents affect the Student directly.

- Other Students: Students share information amongst themselves, which means that other students can influence the amount and quality of information that is available. The reason for sharing information/materials might vary a lot, but, in most cases, these students feel a sense of duty to share knowledge among the entire department Some students might voluntarily create content to help other students.

- Friends: Through discussions and social interactions, they influence the Student and are one of the first sources of information when the Student has questions. They often try to help, even though they are not sure themselves about the answer, leading to misinformation.

- Alumnus: Might be close to the Student, meaning that they are very likely to have knowledge about the subjects/difficulties the Student is facing (seeing that they have gone through the same). Due to this, they play a significant role in spreading information (and sometimes misinformation).

- UC: The UC has no option but to impose the rules approved by the dean on all the departments/universities.

- Department/University: Has an help desk to aid students with any question relating to the institution or their academic life.

- Repography: Is a last case resource that students use in order to have access to materials from previous students. Is not very liked due to not being free.

- Dean: As the entity that rules over the UC, the dean is the one who approves the rules and imposes them on the UC. It is in his best interest that the UC performs well, seeing that this brings him and the university good reputation.

There are several break points present in the model. These occur when a Student asks a question to either a friend, another student or an alumnus and they do not have the answer or have wrong answers. When a Student asks a question to a professor, there also exists a break point because the professor might not know the answer or be unavailable or unapproachable. Lastly, there is a break point between the Student and the Repography, seeing that they might not have materials that cover the needs of the Student or they might be incorrect.

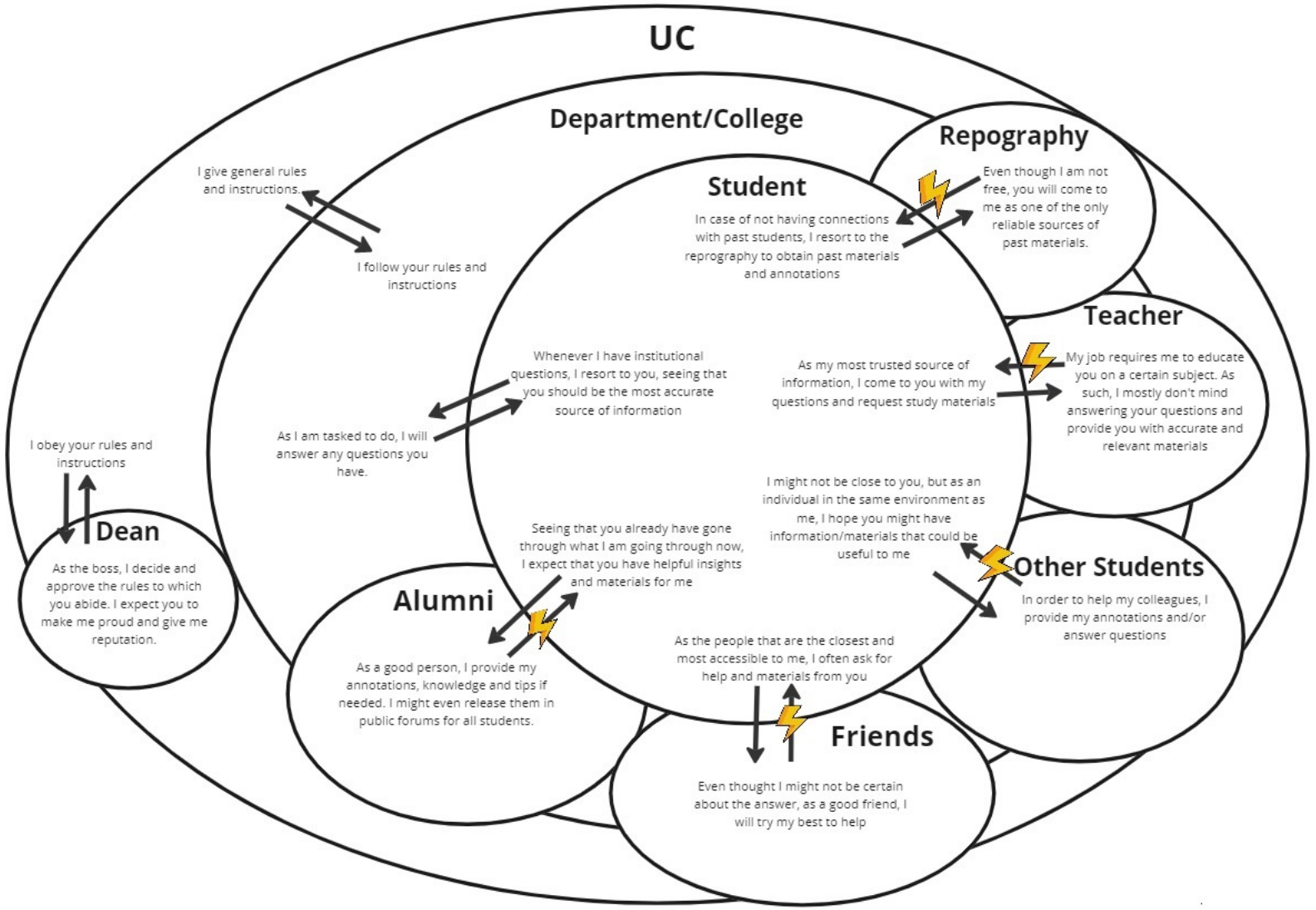

Figure 3.4: Cultural model

3.5 Personas

According to the contact we had during the interviews, we formulated the following personas:

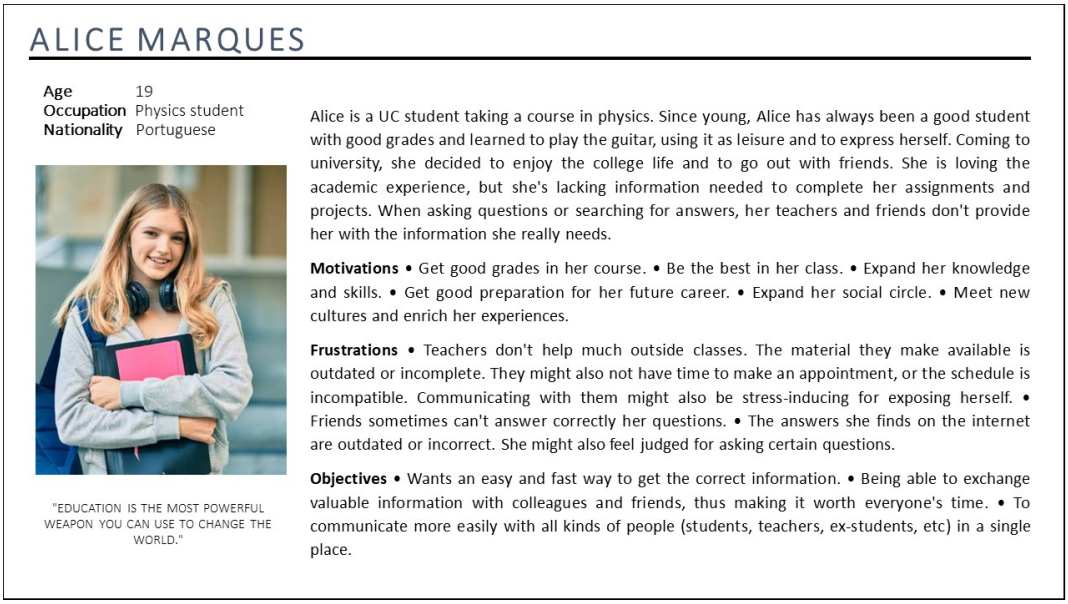

Figure 3.5: Student Persona

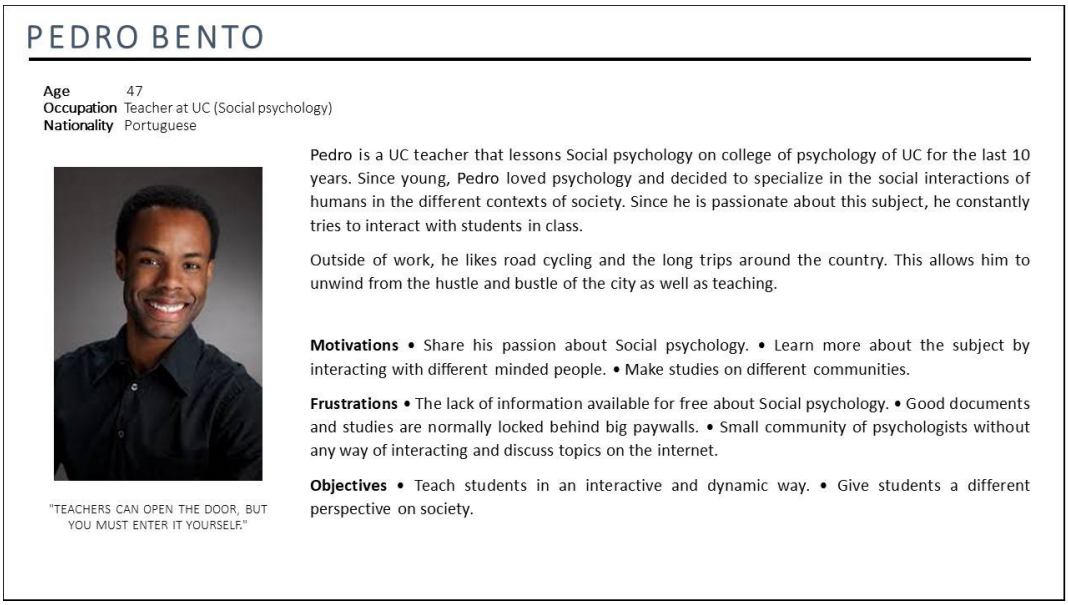

Figure 3.6: Professor Persona

3.6 Affinity Diagram

Based on the interviews made to several potential users, the affinity diagram was built based on their inputs, organized into the main groups that constitute the platform.

Figure 3.7: Affinity Diagram

4 Use Cases

A use case is a written description of how users will perform tasks on your website. It outlines, from a user's point of view, a system's behavior as it responds to a request. Each use case is represented as a sequence of simple steps, beginning with a user's goal and ending when that goal is fulfilled.

In this chapter, we will explain how the users will interact and perform tasks on the platform by displaying the use cases. Before showing the use cases, we first needed to create the context diagrams, which are divided into three levels: level 0, level 1 and level 2.

Level 0 is the utmost top level, which only shows the actors and the platform. Level 1 is a bit more detailed and shows the interaction between the actors and the use cases. As for level 2, it presents more details of certain use cases that extend or include some others. Not all possible use cases for the platform were written, but those are the ones we considered the most important and they are a good baseline to understand how the interaction between the users and the platform will work.

4.1 Context Diagram (level 0)

This context diagram makes a basic understanding of the different actors that exist in the StackOverflow4UC system. There isn't yet a distinction between all of them, which will be made available at the level 1.

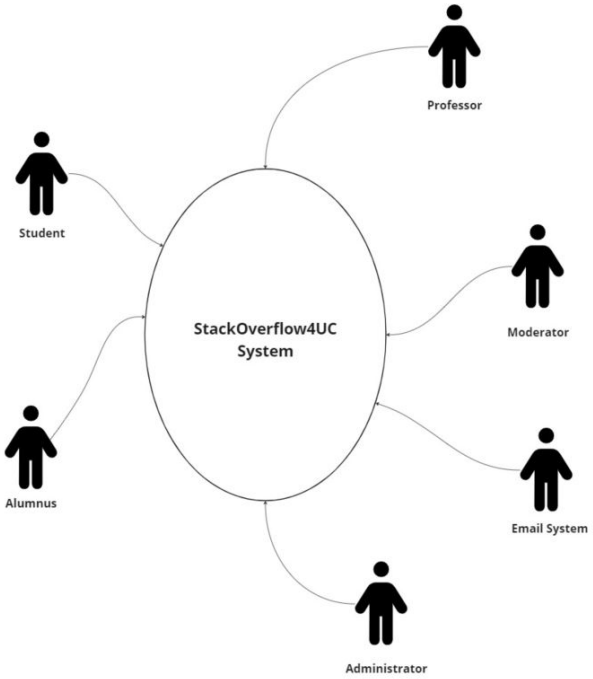

Figure 4.1: Context Diagram (level 0)

The following table exists in order to better explain what are the responsibilities of each role in the context diagram:

Actor	**Description**
Student	Responsible for using the system after delivery **Assumption:** It has limited knowledge of a given field
Alumnus	Responsible for using the system after delivery **Assumption:** It has medium knowledge of a given field
Professor	Responsible for using the system after delivery **Assumption:** It has high knowledge of a given field
Moderator	Responsible for the deletion of content and for banning users
Administrator	Responsible for managing the Moderators, the database and tags as well as for the security of the platform
Email System	Responsible for sending different types of emails: notification, registration, amongst others

4.2 Context Diagram (level 1)

For this diagram the main actors are the *User*, *Moderator* and *Administrator*, whilst the secondary actor is the *Email System*. Taking a closer look at the *User* actor, it contains three entities that play the same role but are different types of people: alumnus, student and professor. Every entity from this role has the same use cases as the other two. **There is a hidden role, which is that of a Guest** that needs to register to the platform to become a User. Since the Guest doesn't have any other interaction with the platform besides the registration process, we didn't find it necessary to include it in this diagram.

Regarding the system, **twelve major use cases** were attributed to the main actors and only one was assigned to the secondary actor.

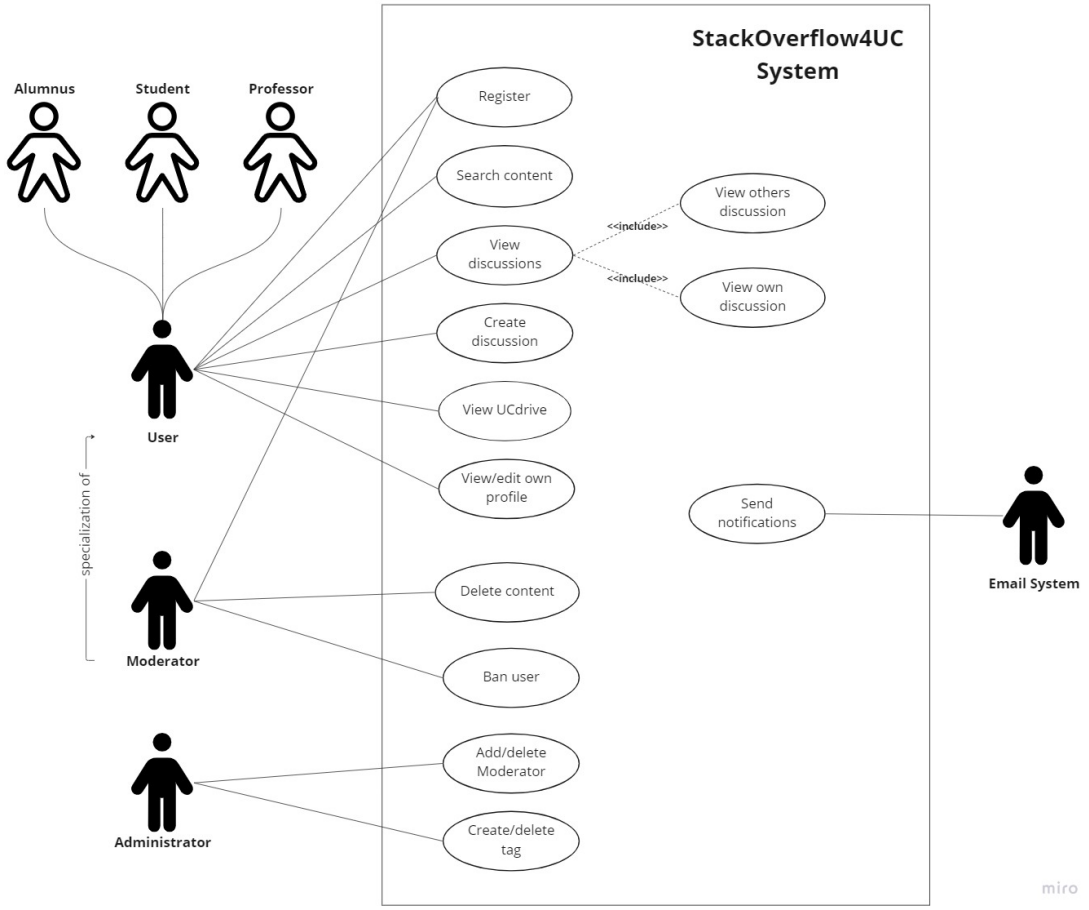

Figure 4.2: Context Diagram (level 1)

A normal user can be upgraded to the moderator role via an Administrator.

Furthermore, when a user is banned, he remains in the database but with no access to the platform until a moderator decides to remove the ban.

4.3 Context Diagram (level 2)

Because the platform contains several use cases with nuances, some of them were taken from the level 1 and were expanded for this level to be further analysed and to get a deeper understanding of them. One example is that of the two use cases "View own discussion" and "View other discussion", which share the inclusion of a few use cases but not others.

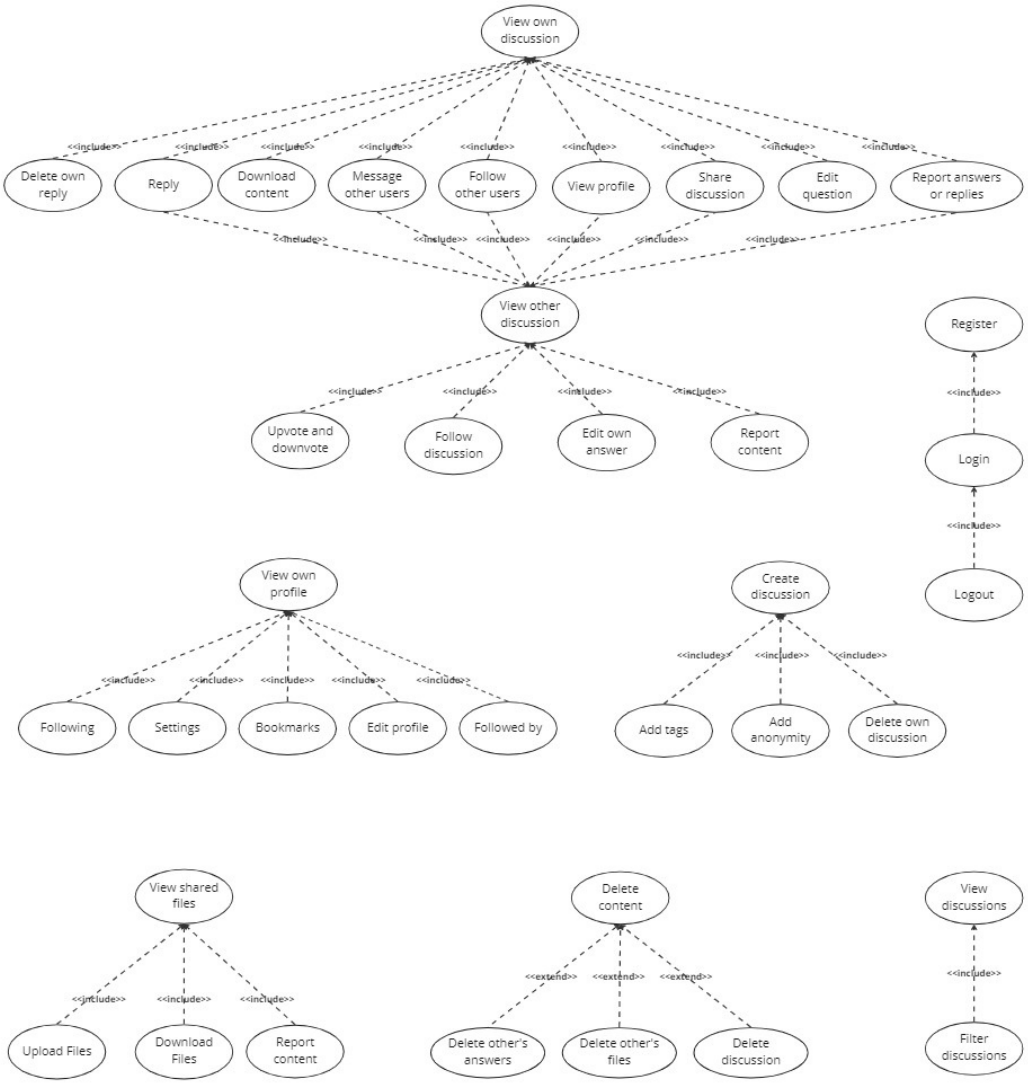

Figure 4.3: Context Diagram (level 2)

The feature "add anonymity" exists so that people can comment, create discussions, amongst other things without other's knowing who wrote/posted it. This feature is thought to eliminate the majority of the shame people carry for posing "stupid" questions/answers.

If a user decides to put some kind of offensive/funny content that deviates from the purpose of the platform then, the moderator can delete such content has he wishes. The content to be deleted can be a discussion, an answer and/or a file.

4.4 Use Cases

Out of the thirteen use cases identified on the contextual diagram (level 1), we chose the nine most important that better display the core of the platform.

Use Case #1	View shared files.		
Level	Sea.		
Description	See list of shared files.		
Assumption	There is a list of files.		
Pre-condition	The files are available.		
Trigger	Click button to view shared files.		
Primary actor	User.		
Secondary actor			
Main success scenario	**Step**	**Main actor**	**StackOverflow4UC system**
	1		Displays list of shared files sorted by most recent.
	2	Selects specific tag.	
	3		Displays list of shared files sorted by most recent from the specific tag selected by the user.
	4	Searches for a specific file/content/subject/keyword.	
	5		Displays list of shared files related to the user's search
Exceptions	**Description**		
	1a. System is down. 1a1. Displays a message informing that the system is not available **2a. Database is down.** 2a1. Displays a message informing that there are no discussions avaiable. **5a. There are no results to the specific search** 5a1. Display information message		

Figure 4.4: Use Case #1 - View shared files

Use Case #2	Create discussion.		
Level	Sea.		
Description	Create a discussion. After the user writes the title to the discussion, a list of similar questions appears.		
Assumption	There's an option to create a discussion.		
Pre-condition	Must be on the homepage.		
Trigger	Clicks on the button to create a discussion.		
Primary actor	User.		
Secondary actor			
Main success scenario	**Step**	**Main actor**	**StackOverflow4UC system**
	1		Displays all the fields the user must fill.
	2	Fills the title of the discussion.	
	3		Displays list of discussions related to the title inserted by the user.
	4	Checks the displayed list to see if any previous discussion might have the solution to their problem. If not, continues writing the discussion.	
	5	Fills the rest of the fields (discussion's body and tags).	
	6	Presses the submit button.	
	7		Verifies the data (minimum characters in each field).
	8		Saves discussion.
Exceptions	**Description**		
	1a. System is down. 1a1. Displays a message informing that the system is not available. **3a. Database is down.** 3a1. Displays a message informing that there are no discussions avaiable. **4a. User finds one solution to their problem** 4a1. Goes to the solution found and the discussion is not created **5a. There are no tags that fit the purpose of the discussion.** 5a1. Displays information message. **7a. Some of the fields do not have the minimum number of characters** 7a1. Redirects to the same page with an information message about the issue without deleting already written content.		

Figure 4.5: Use Case #2- Create Discussion

Use Case #3	View discussion.		
Level	Sea.		
Description	The user sees the content of a specific discussion.		
Assumption	The discussion exists.		
Pre-condition	The user must be on the homepage.		
Trigger	The user clicked on the discussion page.		
Primary actor	User.		
Secondary actor			
Main success scenario	**Step**	**Primary actor**	**StackOverflow4UC system**
	1		Displays the discussion information, answers, solution and comments.
Exceptions	**Description**		
	1a. System is down. 1a1. Displays a message informing that the system is not available.		

Figure 4.6: Use Case #3 - View discussion

Use Case #4	Search content.		
Level	Sea.		
Description	The user searches for content.		
Assumption	There's a tool to search for content.		
Pre-condition	Must be in any page.		
Trigger	Click on the search bar.		
Primary actor	User.		
Secondary actor			
Main success scenario	**Step**	**Main actor**	**StackOverflow4UC system**
	1	Types text/keywords/phrase that they want to search for.	
	2		While the user is typing, sends suggestions for content related to what is being written.
	3	The user hits enter.	
	4		Displays to the user the results that match the text/keywords/phrase first, followed by the most similar ones.
Exceptions	**Description**		
	2a. There are no suggestions for the content that the user typed. 3a1. Displays a message informing that there are no suggestions. **4a. No content was found.** 4a1. Displays a message informing that no content was found. **4b. There are no text/keywords/phrase that fit the search.** 4b1. Displays a message informing that no content was found. **4c. There are no similar results to the search.** 4c1. Displays a message informing that no content was found.		

Figure 4.7: Use Case #4 - Search content

Use Case #5	Edit own profile.		
Level	Sea.		
Description	The user clicks on their user icon to open their profile and edit their information.		
Assumption	The information is editable.		
Pre-condition	Must be in the user's profile page.		
Trigger	Click on the "Edit profile" button.		
Primary actor	User.		
Secondary actor			
Main success scenario	**Step**	**Main actor**	**StackOverflow4UC system**
	1		The user personal information becomes editable (username, profile picture, bio).
	2	Changes the fields.	
	3	Clicks on the apply button.	
	4		Validates the updated information.
	5		Updates the user personal information.
Exceptions	**Description**		
	1a. The system is down. 1a1. Notifies user that the system couldn't be reached. **4a. The new information is not valid.** 4a1. Marks invalid input and display reason to be invalid. 4a2. The user corrects the information and clicks again to update information. **5a. Internal server error updating information.** 5a1. Notifies user that an internal server error occurred and the operation was canceled. 5a2. Redirects user to last page visited.		

Figure 4.8: Use Case #5 - Edit own profile, data dictionary available in 4.13

Use Case #6	Register a new user.		
Level	Sea.		
Description	The person registrates to the platform to become a user.		
Assumption	The user has an email with the UC domain and no account created.		
Pre-condition	The email belongs to the UC domain.		
Trigger	Click on the button to register.		
Primary actor	Guest.		
Secondary actor			
Main success scenario	**Step**	**Primary actor**	**StackOverflow4UC system**
	1		Prompts a registration form.
	2	Inserts data in the form. If the email inserted is specific to students, the guest either chooses an Alumnu or Student tag. If the email inserted is specific to teachers, the tag Teacher is automatically associated.	
	3		Verifies if data is valid.
	4		Sends a confirmation email to the guest.
	5	Confirms by clicking the received link.	
	6		Adds the guest as a User to the database.
Exceptions	**Description**		
	1a. System is down. 1a1. Displays a message informing that the system is not available. **2a. Database is down.** 2a1. Displays a message informing that the operation is not available. **3a. The data inserted is invalid.** 3a1. If he username already exists, displays an error and refreshes the page. 3a2. If the password doesn't meet specific criterias, displays an error and refreshes the page. 3a3. If the he email does not belong to the UC domain, displays an error and refreshes the page. **4a. The guest doesn't receive the confirmation email.**		

Figure 4.9: Use Case #6 - Register a new user, data dictionary available in 4.13

Use Case #7	Ban user.		
Level	Sea.		
Description	The moderator bans a user temporarily or permanently.		
Assumption	A policy has been infringed.		
Pre-condition	Must be on the moderation panel.		
Trigger	Selects moderation panel.		
Primary actor	Moderator		
Secondary actor			
Main success scenario	**Step**	**Main actor**	**StackOverflow4UC system**
	1	Opens the list of reported content sorted by number of reports in descending order.	
	2		Displays the list
	3	Clicks the hyperlink leading to the specific reported content (reply, private message, etc)	
	4		Redirects to the page, in the exact place where the reported content is.
	5	Verifies the report causes and checks if the content should be deleted, according to the platform rules. If so, deletes it.	
	6		Deletes content, if moderator decided so.
	7	Sends warning with message.	
	8		Forwards the warning to the user.
	9		Upon 3 warnings, bans the user.
Exceptions	**Description**		
	1a. System is down. 1a1. Displays a message informing that the system is not available. **1b. Database is down.** 1b1. Displays a message informing that there are no discussions available. **5a. None of the content the user has been flagged for is inappropriate** 5a1. The moderator doesn't agree with the report and removes it.		

Figure 4.10: Use Case #7 - Ban user

Use Case #8	Manage tags.		
Level	Sea.		
Description	The administrator creates and deletes content tags.		
Assumption	The tag to create doesn't exist. The tag to delete exists.		
Pre-condition	Must be on the administrator panel.		
Trigger	Select button to manage content tags.		
Primary actor	Administrator.		
Secondary actor			
Main success scenario	**Step**	**Main actor**	**StackOverflow4UC system**
	1		Displays content tags page.
	2	Presses button to add a new tag, write a name.	
	3		Adds tag to database.
	4	Finds the tag to delete.	
	5		Displays available tags.
	6	Selects a tag and deletes it.	
	7		Removes the tag from the database.
Exceptions	**Description**		
	1a. Internal server error loaging page. 1a1. Notifies the user that an internal server error occured. 1a2. Returns the user to the last page. **2a. The tag name written already exists.** 2a1. Writes another name or chooses not to create the tag. **3a. Invalid information on the tag.** 3a1. Marks invalid input and display the reason to be invalid. 3a2. Administrator clicks again to add tag. **6a. The tag doesn't exist**. 6a1. Looks up for the right tag name.		

Figure 4.11: Use Case #8 - Manage tags

Use Case #9	Upvote/downvote discussion.		
Level	Fish.		
Description	The user evaluates a discussion.		
Assumption	There is an option to upvote or downvote.		
Pre-condition	The user must be in the discussion view.		
Trigger	Click on the button to upvote/downvote.		
Primary actor	User.		
Secondary actor			
Main success scenario	**Step**	**Primary actor**	**StackOverflow4UC system**
	1		Sums 1 to the discussion score if the user presses the upvote button or subtracts 1 to the discussion score if the user presses the downvote button.
	2		Updates the discussion creator's reputation score.
Exceptions	**Description**		
	2a. Database is down. 2a.1 Displays a message informing that the operation is not available.		

Figure 4.12: Use Case #9 - Upvote/downvote discussion

User Information	Definition	Data type	Required	Acceptable Values
Username	Name of the user to be displayed in profile.	String	Yes	String with length greater than 0 and less than 100.
Email	Unique academic email that identifies the specific user.	String	Yes	Unique email with the UC domain.
Password	String that allows the user to authenticate in the platform.	String	Yes	String with length greater than 4 and less than 30. Needs to contain at least one letter and one digit.
Description	Small description to be presented in the user profile.	String	No	String with length from 0 to 1000.
Image	Image to be used in the user profile.	BLOB	No	Image file with less than 64 KB. If the file is larger, compress the image.

Figure 4.13: Data dictionary of user - Information the user can manipulate

5 Goal-based Requirements

To implement our goal-based requirements we used KAOS, a notation for goal-oriented software requirements that allows them to be made from goal-models. In those diagrams, the goals are the objectives a system wants to achieve by having actors cooperate in the environment. To achieve all of the above, the Objectiver software was used.

5.1 Goal Model

This type of model allows, through the decomposition of major goals into specific requirements, to clarify the various non-functional requirements of the system. With this in mind, our system has three major goals that will be decomposed.

5.1.1 System is reliable

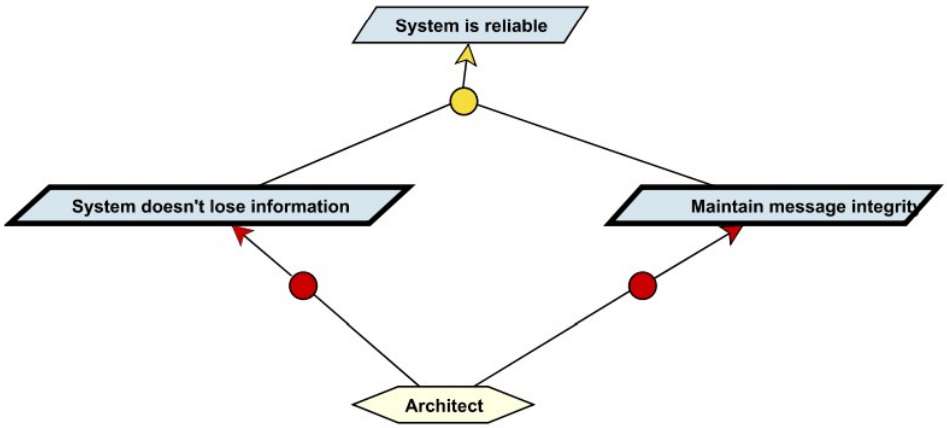

Figure 5.1: "System is reliable" goal diagram

Having a reliable system implies having a software with a high probability of fulfilling its assigned tasks without failures, for a specific period or time, in a specified environment.

It is necessary that the system doesn't lose information and allows content to be accessible. Another aspect necessary to have a reliable system is to ensure message integrity, which consists on delivering a message entirely without losing any information whilst also preventing corrupting it.

All of these requirements are the architect's responsibilities when designing the platform.

5.1.2 Maintain Secure System

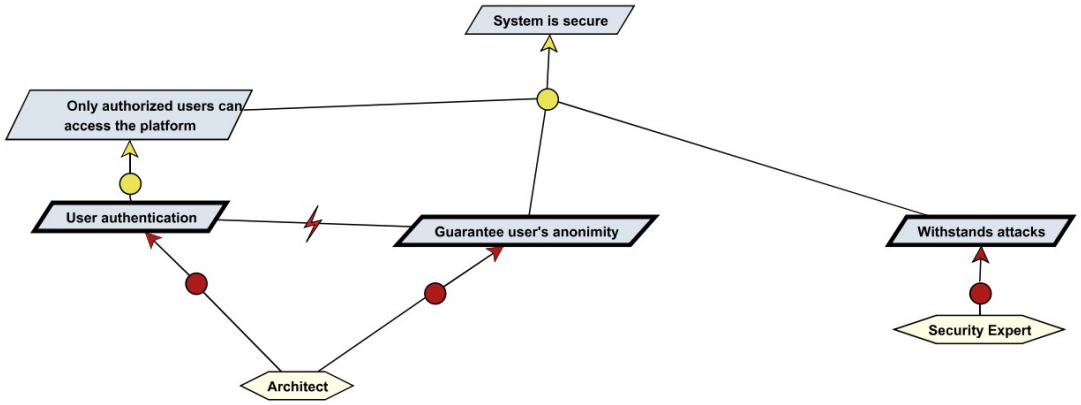

Figure 5.2: "Maintain Secure System" goal diagram

This security diagram gives an overview of what needs to be achieved to get a secure software, which involves having its operations and functionalities remain intact or work as intended when subjected to malicious attacks, meaning that withstanding those attack will keep the system and resources safe in the environment.

For the first goal, the architect must implement the user authentication to only allow authorized users to use the platform. Any account that doesn't belong to the UC domain won't be able to authenticate on the platform.

The second goal involves the user, since he may want to remain anonymous when adding content to the platform, making it the architect's responsibility to guarantee that no personal information is showed in those instances. This is also important to prevent private or confidential information being leaked. There is a conflict between this goal and "User authentication" seeing that you lose some anonymity when creating an account and logging in.

The third and most important goal consists of keeping the system secure so that the system can withstand attacks from malicious agents. That responsibility comes from

the security expert, who needs to prepare and protect the system for those situations.

5.1.3 Usability

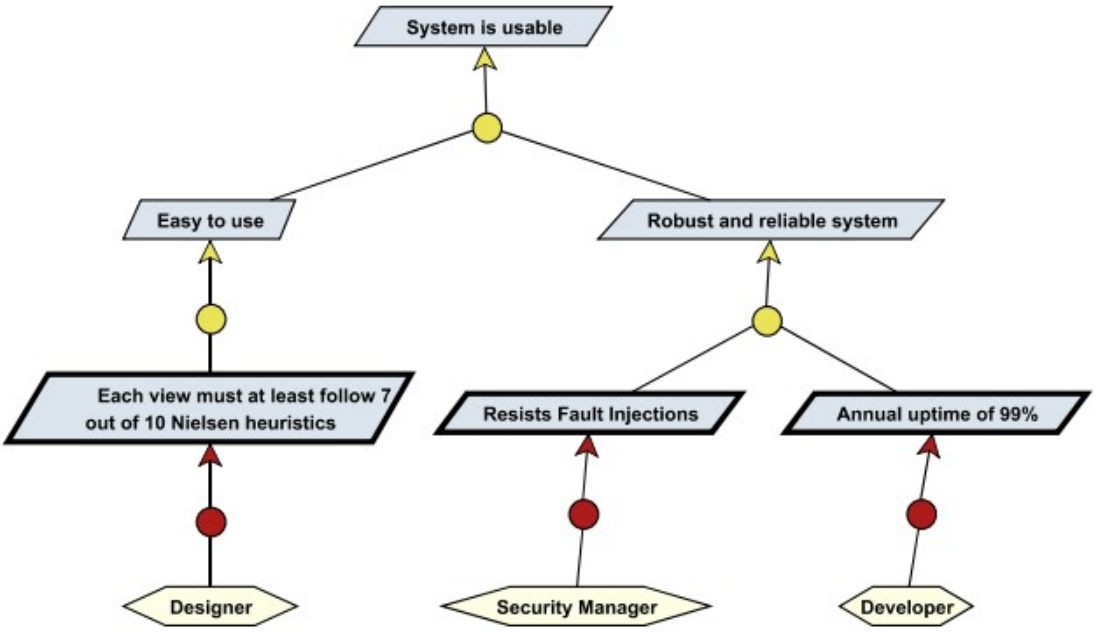

Figure 5.3: "Usability" goal diagram

Usability consists on having a user-friendly platform. This can be achieved by following the Nielsen heuristics and guarantee a minimum of 7 heuristics present on the platform. Those heuristics are 10 principles for evaluating a platform or website interfaces and ease of use [7].

Another requirement to have a usable system is to make sure that the system is robust and reliable. To achieve this, the system should be able to resist fault injections (security manager's responsibility) and the system should have an annual uptime of 99% (developer's responsibility).

5.2 Responsibility Model

The responsibility model aims to identify the agent responsible for each of the non-functional requirements. Agents are either human beings or automated components

that are responsible for achieving requirements and expectations.

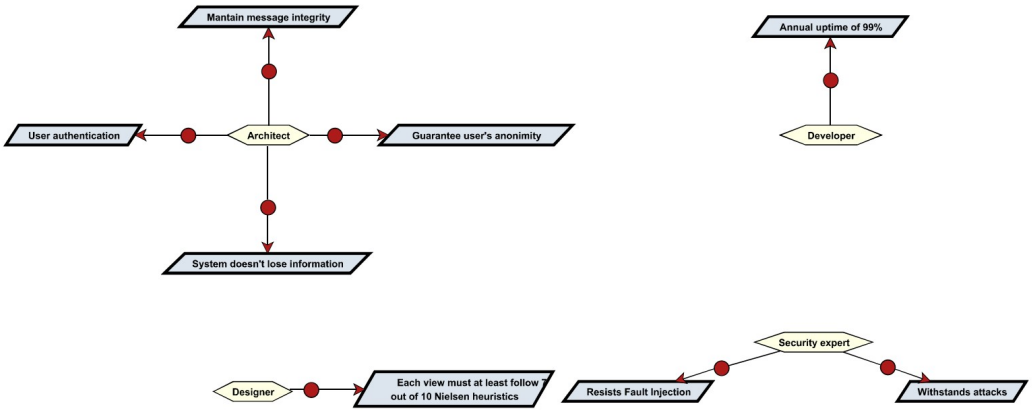

Figure 5.4: KAOS Responsibility Model

Only one agent can be responsible for the implementation of a requirement. After analyzing the requirements, four agents were identified: architect, developer, designer and the security expert. In the figure 5.4 it is noticeable which requirements are attributed to each agent.

5.3 Non-Functional Requirements

1. RFN1 - (Reliability) System doesn't lose information
2. RFN2 - (Reliability) Maintain message integrity
3. RFN3 - (Security) User authentication
4. RFN4 - (Security) Guarantee user's anonimity
5. RFN5 - (Security) Withstands attacks
6. RFN6 - (Usability) Each view must at least follow 7 out of 10 Nielsen heuristics
7. RFN7 - (Usability) Resists fault injections
8. RFN8 - (Usability) Annual uptime of 99

6 User Interaction

Our User Interaction will consist in five major parts: the UED, paper prototype, the Cognitive Walkthrough, the mockups and respective usability checklist.

6.1 UED

The User Environment Design lets us make the transition from a vision into a coherent design, making the use cases more visual and explicit. It's a way to imagine and observe which services the platform will support and how to navigate between them, as well as showing several functionalities that work between them.

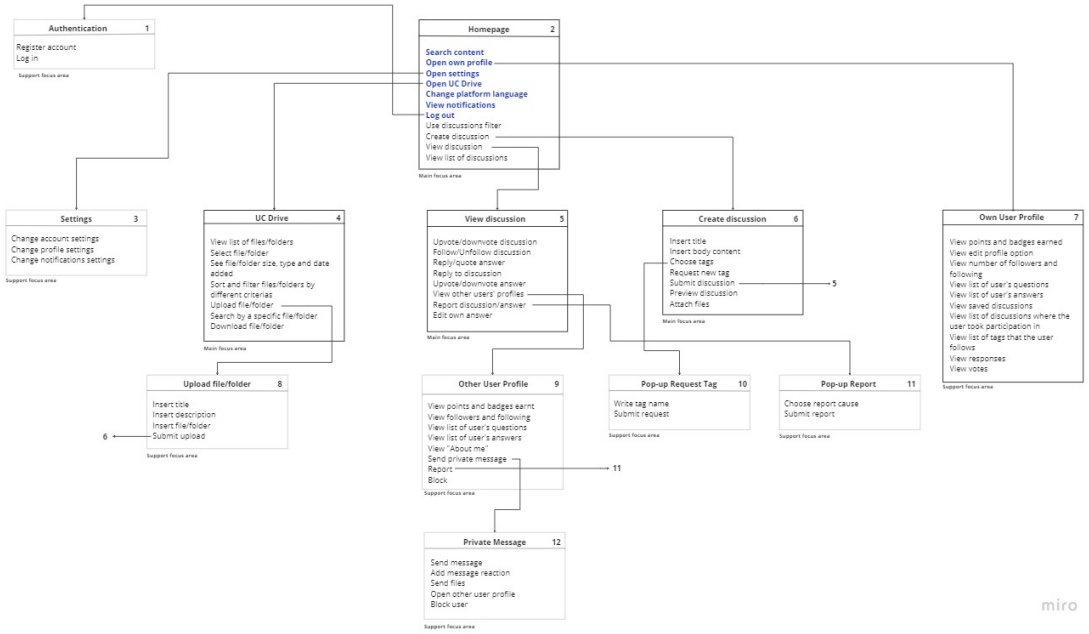

Figure 6.1: User Environment Design (UED)

What we show is the UED as a regular user, since the moderator and administrator will have additional functionalities and different views, such as "Create tag" or "Ban user".

In order to avoid a confusing display of the diagram, the Homepage view has several tasks with a blue color, which represent fixed tasks that appear in the header or navigation bar, elements that appear in all of the views. For example, "Open settings" will always lead to the same view: the Settings view. We also find it relevant to organize the UED in a top-to-bottom hierarchy. The top areas are the first things that can be seen when entering the platform, and their successors are the remaining features in order of access.

A specific case of the blue tasks is the "View notifications", which will lead to multiple views that are specific to the notification. For example, if the user receives a notification about an reply they received a discussion they created, they can click on the notification and directly access that reply, which will lead them to the "View discussion" view.

The UC Drive and Homepage work differently. While the Homepage shows a list of discussions where you need to open a new web page in order to see its content, the UC Drive doesn't need you to open new pages, everything you see remains in the same page, even if you apply new filters, sort folders by size, etc; the content of the page is updated in real time. The only functionality that would take you to a new web page is the "Upload file/folder".

6.2 Paper Prototype

Bearing in mind that the solution to our problem is digital, we thought that making paper prototypes was a good, and also more practical, way to validate our platform.

The design was made considering not only the client's requirements but also the end user's needs, and therefore, receiving feedback on it is an important step of the process.

For the construction of the paper prototype we decided to address not only our main focus areas but also on the views that support them. We made a set of 6 main views and, in addition to that, we decided to represent the pop-ups, warning messages and filters, since we found them to be essential by taking into account Nielson's usability checklist when building the views.

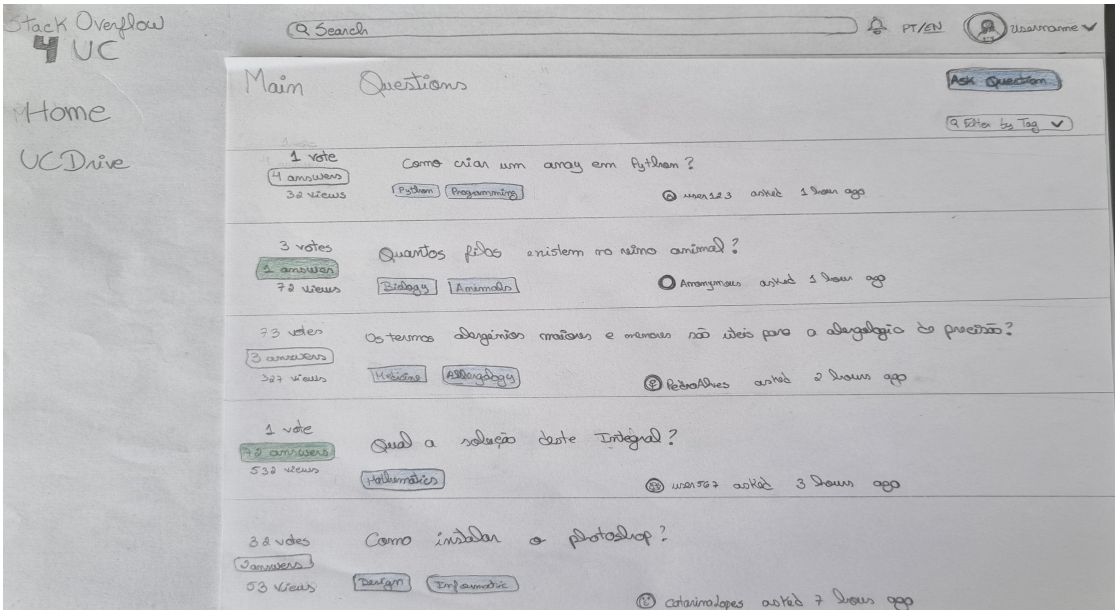

Figure 6.2: Paper Prototype of the HomePage view

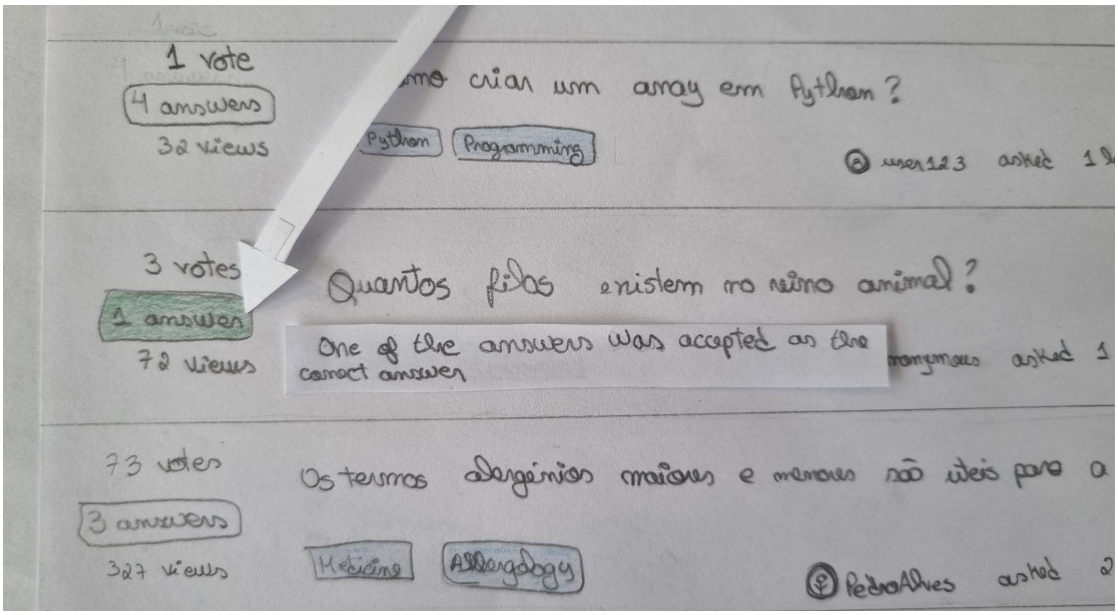

Figure 6.3: Paper Prototype of the HomePage view with answer accepted pop-up

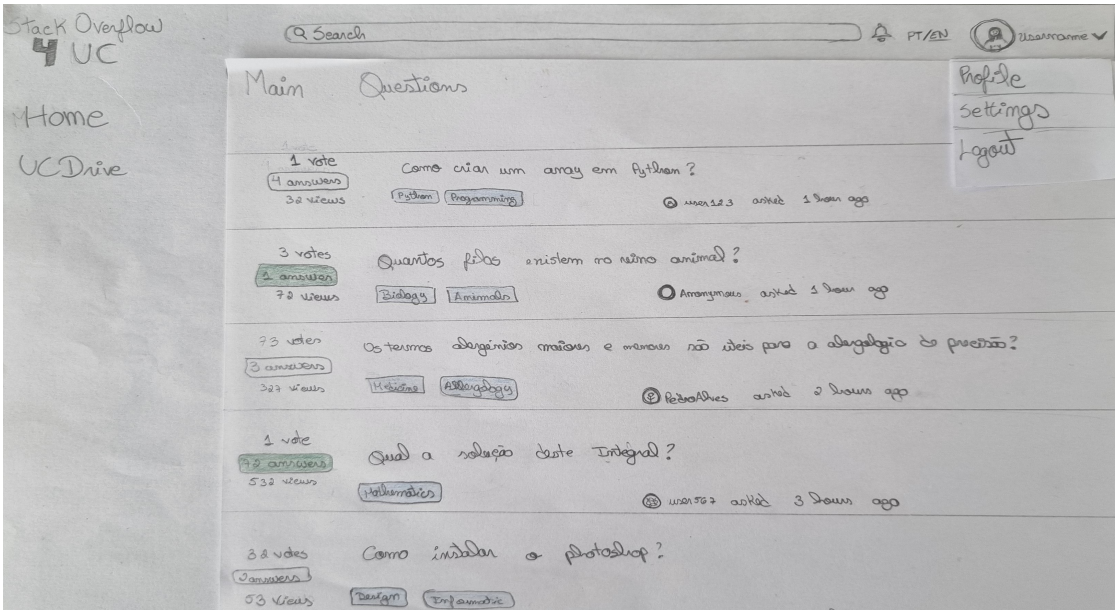

Figure 6.4: Paper Prototype of the HomePage view with profile's pop-up

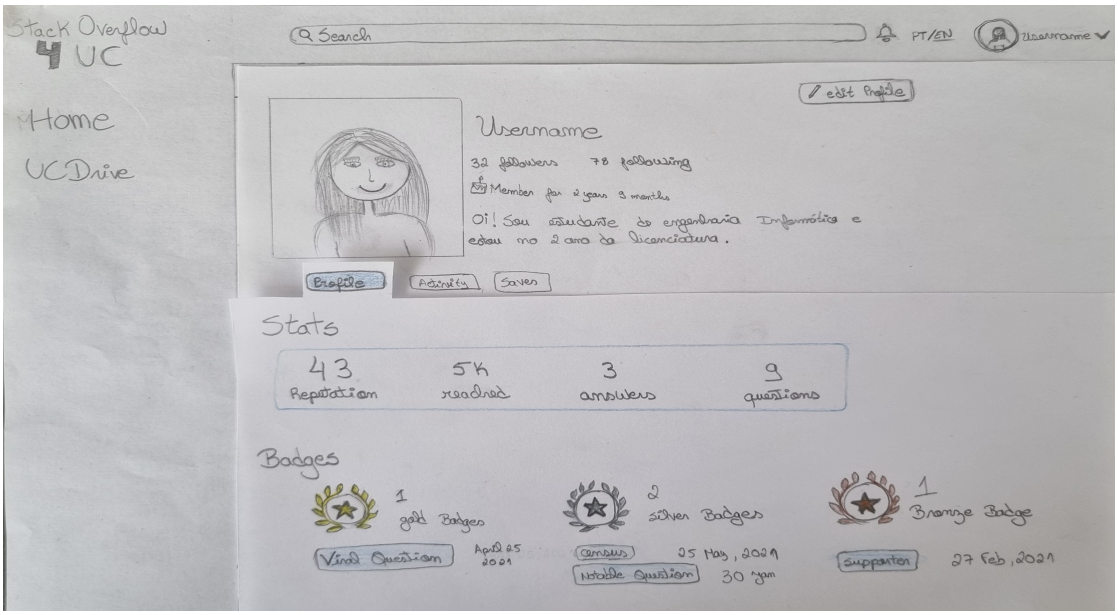

Figure 6.5: Paper Prototype of the Profile view

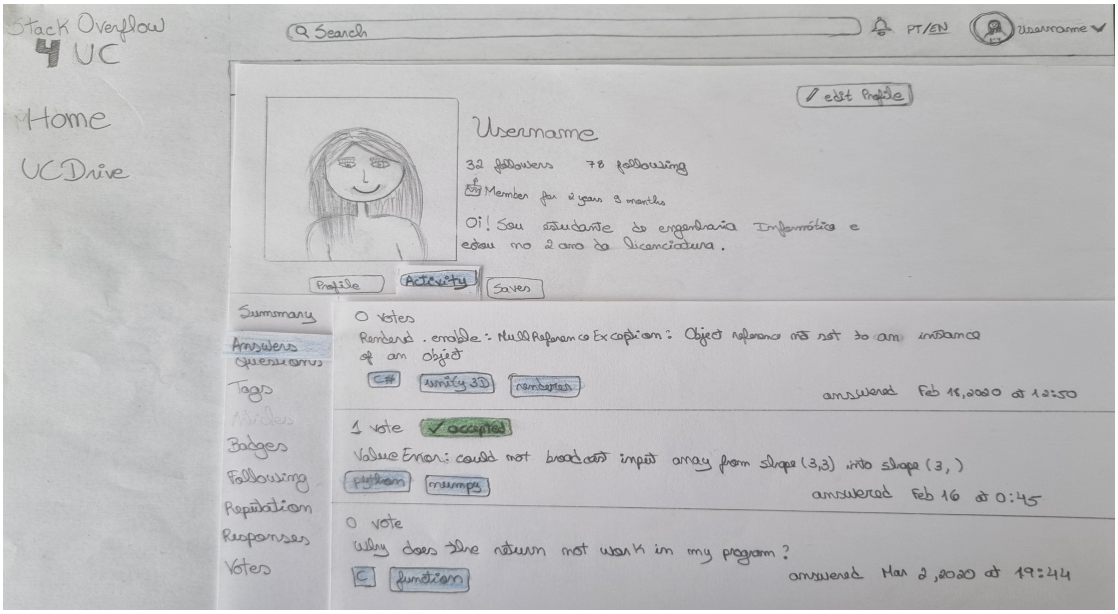

Figure 6.6: Paper Prototype of the Profile view with user's activity

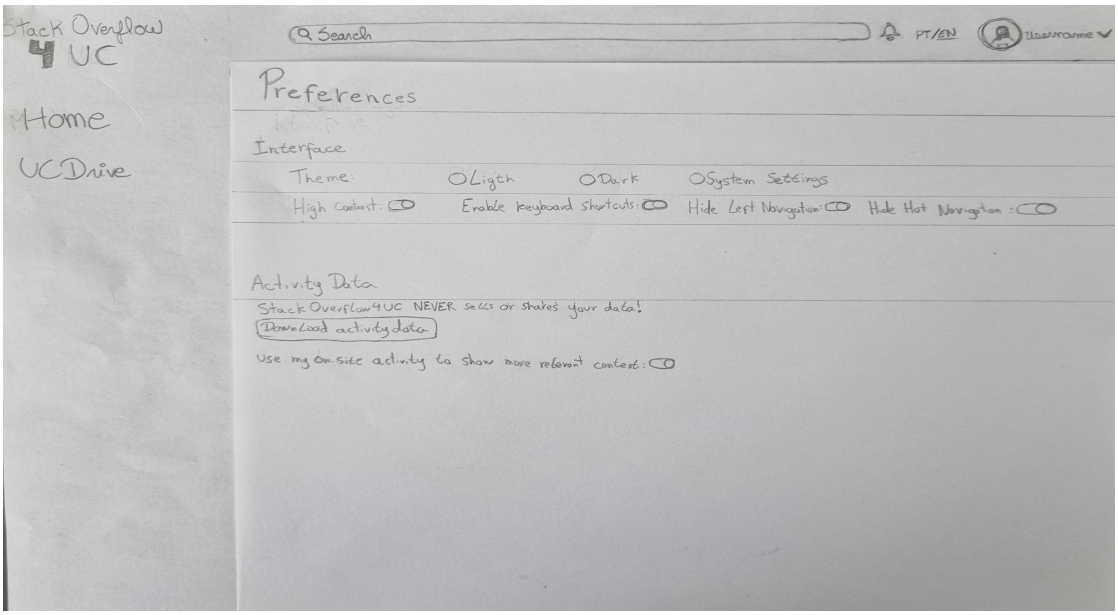

Figure 6.7: Paper Prototype of the settings view

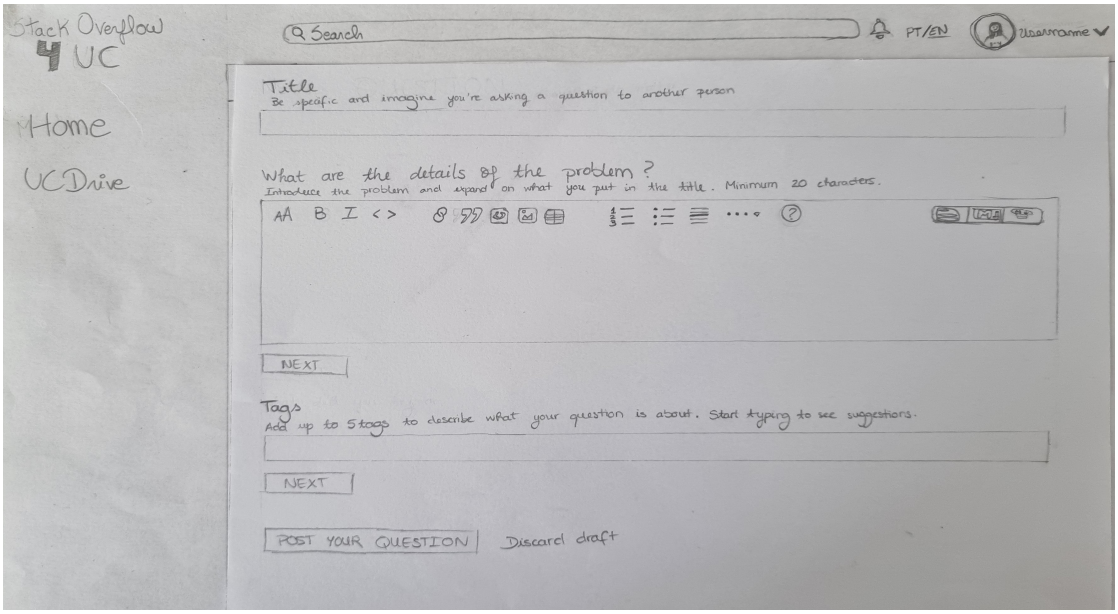

Figure 6.8: Paper Prototype of the create discussion view

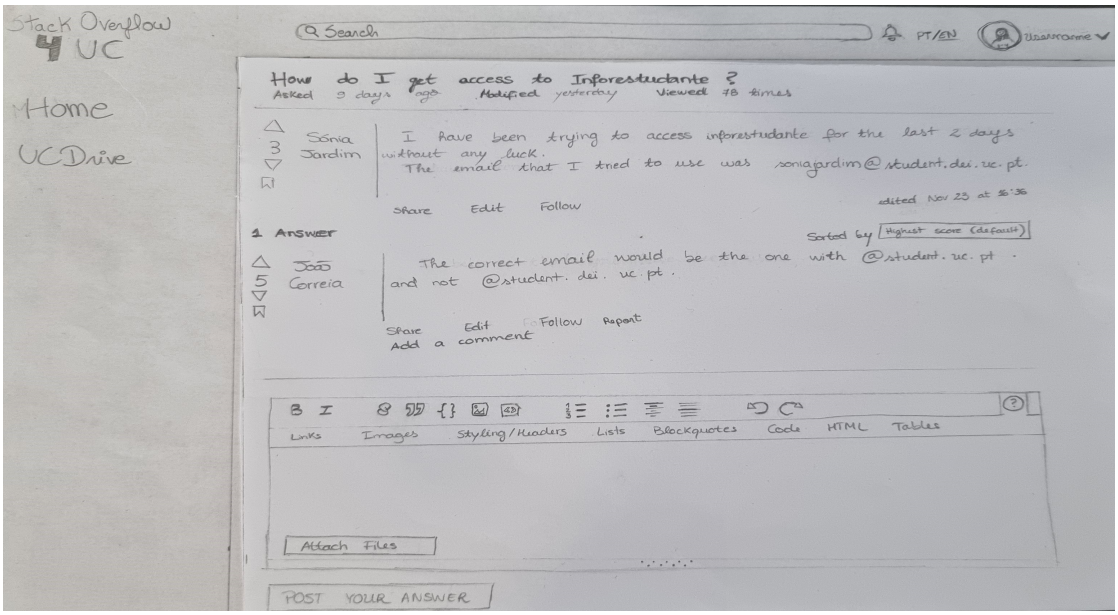

Figure 6.9: Paper Prototype of the reply view

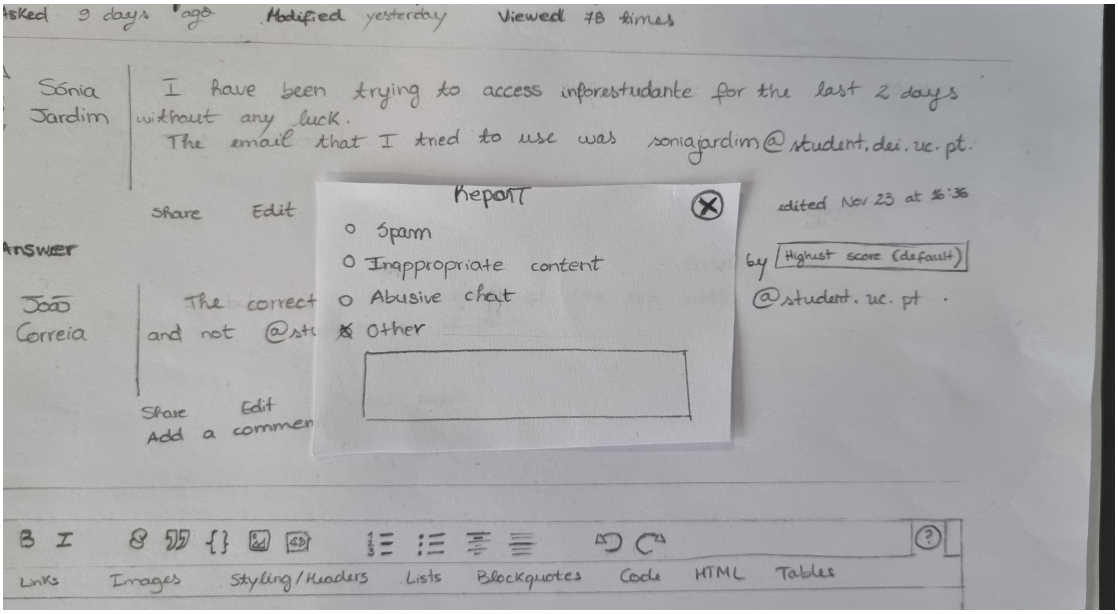

Figure 6.10: Paper Prototype of the discussion view with pop-up report

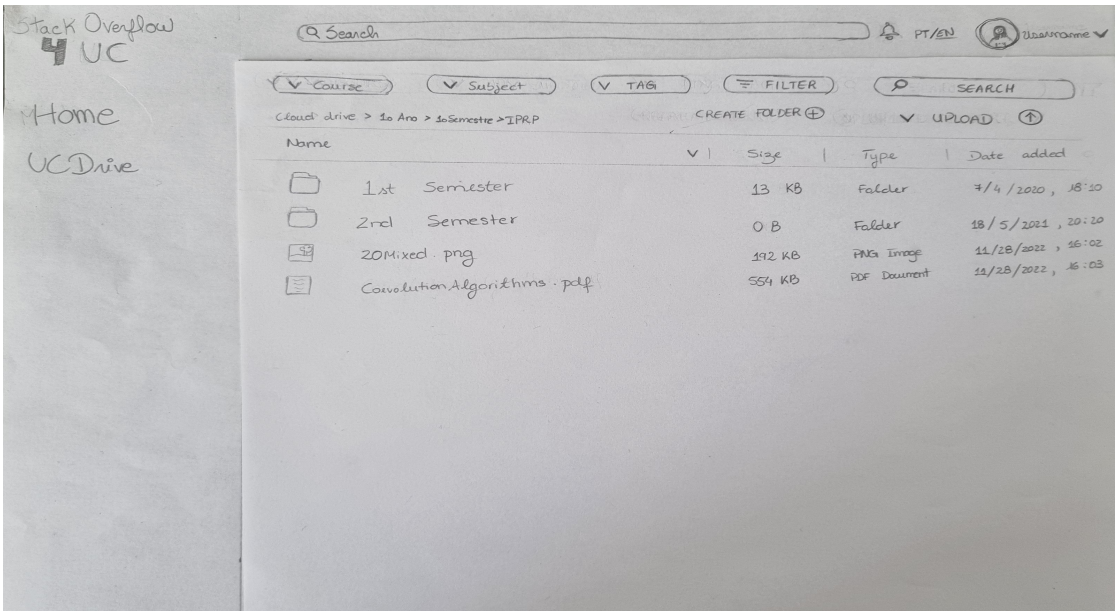

Figure 6.11: Paper Prototype of the UCDrive view

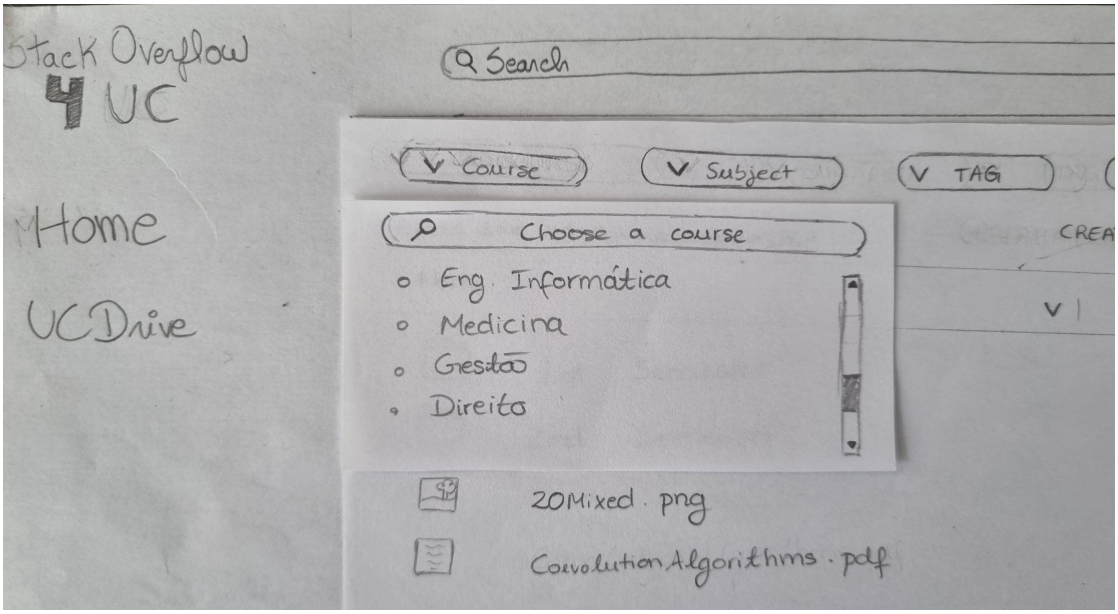

Figure 6.12: Paper Prototype of the UCDrive view with dropdown course

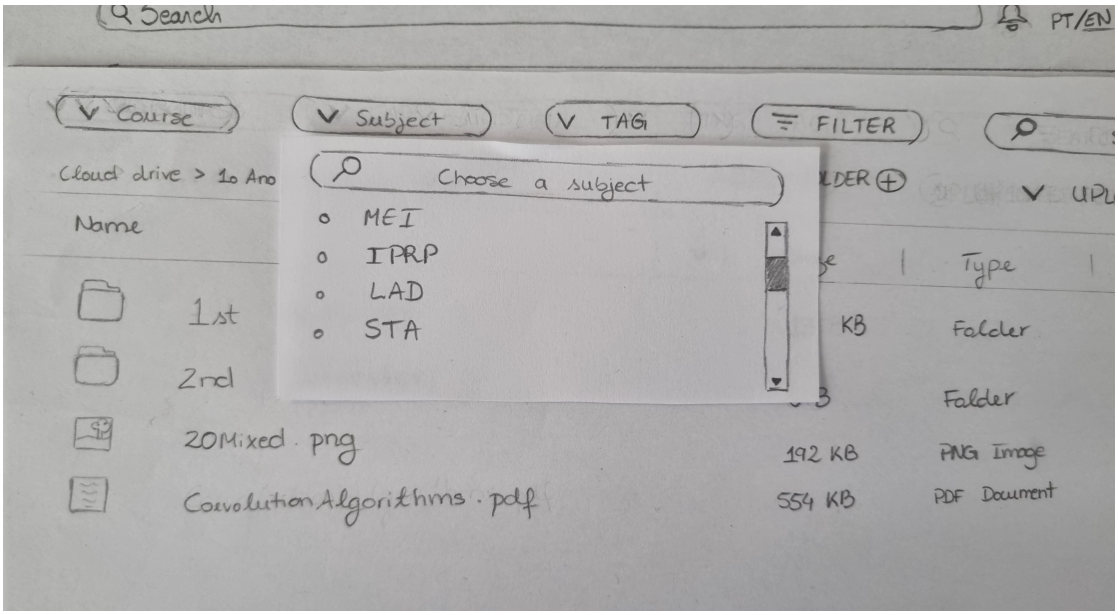

Figure 6.13: Paper Prototype of the UCDrive view with dropdown subject

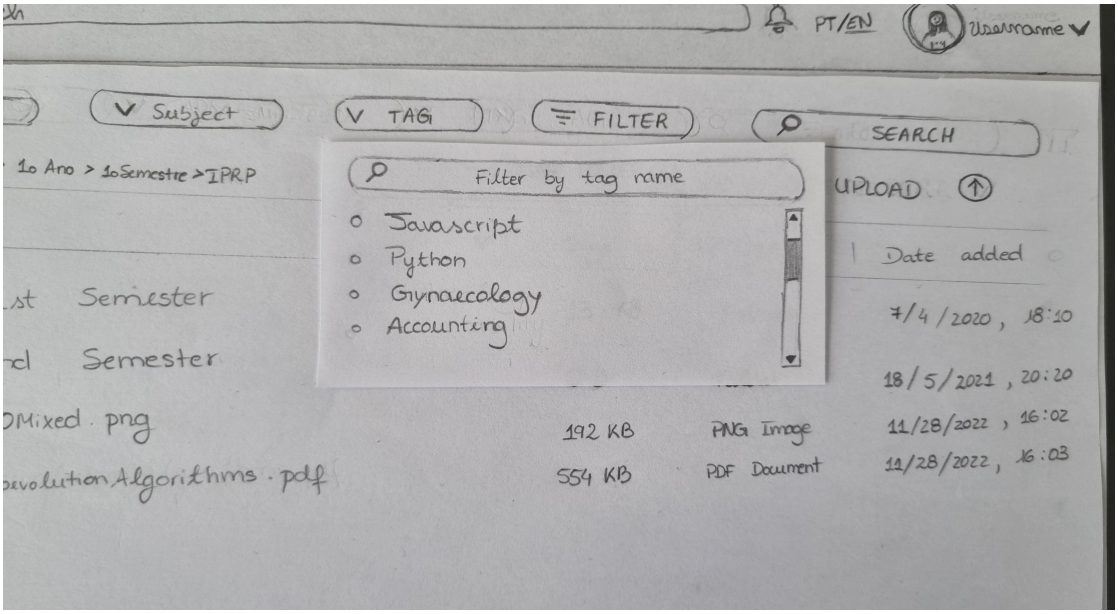

Figure 6.14: Paper Prototype of the UCDrive view with dropdown tag

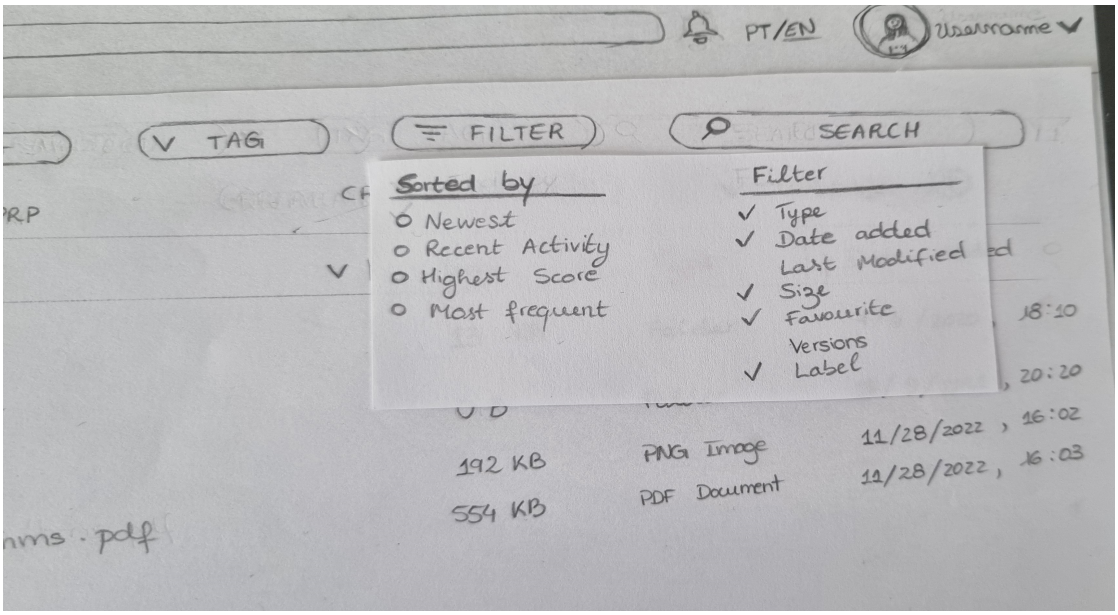

Figure 6.15: Paper Prototype of the UCDrive view with dropdown filter

6.3 Cognitive Walkthrough

In order to evaluate the usability of our platform, we decided to do a cognitive walkthrough. Taking into account the scope of our platform, we decided to ask university students from different courses to give us their opinion regarding the views presented. Our choice fell on one student from Management, one Master's student from Computer Engineering and one PhD student (who's also a professor) from Mechanical Engineering.

6.3.1 Plan

Before starting the tests, we created a process to clarify what approach we would take with the testers and decided that the ideal path would be:

- Make a brief introduction about the purpose of the platform;
- Explain some assumptions we had when we developed the paper prototype;
  - The user is already registered on the platform;
  - The user is a computer engineering student who has been using the platform for some time.
- Ask the tester to put himself in the user role and to develop some tasks.

We then proceeded to which tasks we should ask the testers. We thought it would be important to ask them to perform the most important activities of our platform:

- View a discussion;
- Use a discussion's filter;
- Ask a question;
- Reply a question;
- See what your badges and stats are;
- Download a file.

6.3.2 Results

All the tasks were done successfully and, overall, the user experience on the platform was positive. The testers showed good feedback towards the platform visual presentation

and the interactions between each view.

One change recommended by the Management student was to use the same filters on the UCDrive view and for the Homepage view to maintain consistency across the platform, and so we chose to follow the suggestion.

As for the main complaints, some testers said that the navigation bar (left bar) was too empty and big, making it an hindrance. Others complained about the Settings page, saying that it looked bland and uninspired.

As far as the questions are concerned, the following was noted during testing:

1. Will the user try and achieve the right outcome? By observing the interaction the testers had with the paper prototypes, we can conclude that all of them were available to achieve the right outcome without any major problems or mistakes.

2. Will the user notice that the correct action is available to them? All the testers, except one that had a hard time finding the UCDrive button, were able to notice the actions they needed to use.

3. Will the user associate the correct action with the outcome they expect to achieve? All testers were able to use the correct buttons for the desired outcome without any major errors.

4. If the correct action is performed; will the user see that progress is being made towards their intended outcome? No tester had any complaint about not having feedback after performing an action. The website has been made in such a way that the user has constant feedback about all the actions they make.

6.4 Mockups

Mockups are visual drafts of the product design that showcase a possible look of the finished product. Four mockups were designed from the main focus areas. They follow the same structure and can be divided in 3 parts: The **top bar**, the **left sidebar** and the **main content zone**. This structure turns the platform more consistent and familiar to the user, allowing for better usability.

On the left part of the top bar is displayed the platform name, which is clickable and redirects the user to the Homepage. On the center there is a search bar that allows the user to search for any kind of content, such as a username, discussion, file, etc. On the right, there are clickable icons that show messages with other users, notifications, changing the language of the platform and the profile icon that is expandable and displays several options. The sidebar contains two tabs that allow the user to change between

the discussions part and the repository part of the platform. The latter allows users to consult files from courses and repositories.

As for the structure inside the main content zone, it varies depending on the page shown.

We decided to only make mockups of the main views without worrying about popups or filters, seeing that these are already represented in the paper prototypes. In the construction of the mockups, we had into consideration the feedback obtained from the cognitive walkthrough.

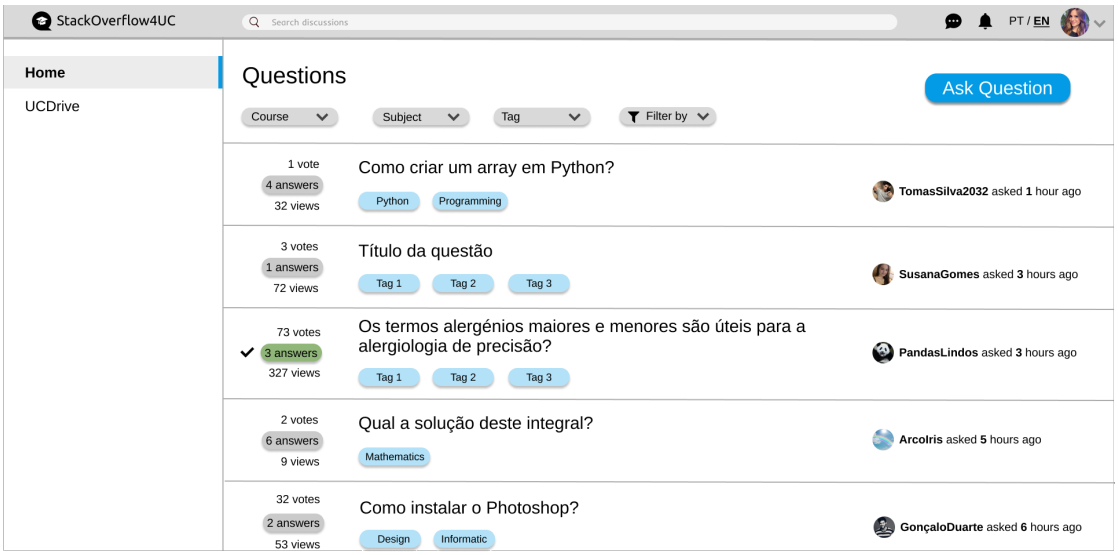

Figure 6.16: Mockup of the HomePage view

The Homepage mockup is the most important, since it's the first page the users will see after login. The main content zone in this view shows, by default, the most recent questions and it's possible to apply several filters to display the questions in another way. For example, a filter could be "Top questions" or "Highest Score" to display the questions with the most upvotes. On the top right there's a "Ask Question" button that redirects the user to another view and allows to ask a new question, i.e, create a discussion.

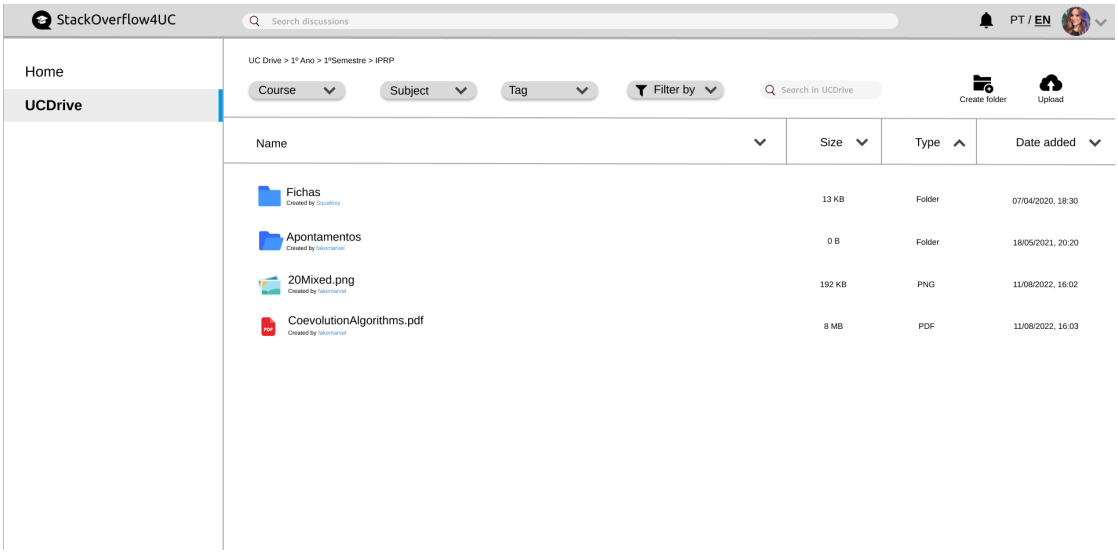

Figure 6.17: Mockup of the UCDrive view

Similar to the Homepage view, the UC Drive view allows to choose several filters to display folders and files from different repositories. The user is able to create instantly new folders by clicking on the top right corner button, but to be able to upload files and folders containing files they need to wait for a moderator's validation to do so.

There's also a smaller search bar that allows for the user to search for specific names of files and folders. The main difference between the small and the big search bars is that the first one starts suggesting files/folders based on what we started to write. For example, if we start to write "Alg" it will show files from the Algebra course or files containing those letters, such as, for example, "AlgebraApontamentos.pdf".

The user can also click on the arrows on the right of each column to sort the content by that same column.

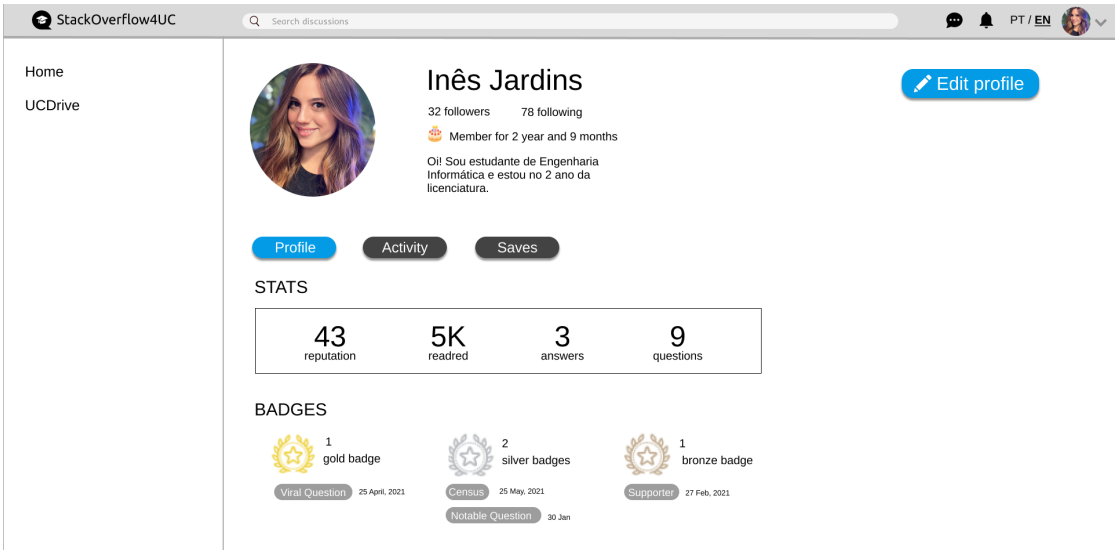

Figure 6.18: Mockup of the profile view

The profile view shows what a user's own profile will look like. The user can click on Edit profile to start editing several settings, which will lead to another support focus view. The user can click on the Profile, Activity and Saves buttons, which will change their color from black to blue to indicate the type of information that is being displayed.

When "Profile" is selected, the page displays several stats, such as the number of questions answered or asked, and also displays the badges that the user has obtained for fulfilling certain requirements on the platform. For example, if the user asked a question that was upvoted a lot of times in a short amount of time, they will receive the badge "Viral question".

When "Activity" is selected, the page shows their most recent (by default) questions, answers, upvotes, follows etc, i.e, any activity that the user wants to show on their feed (which is customizable). Finally, when "Saves" is selected, it simply displays discussions or tags the user saved. It works the same way as bookmarks on a browser, but for the platform instead.

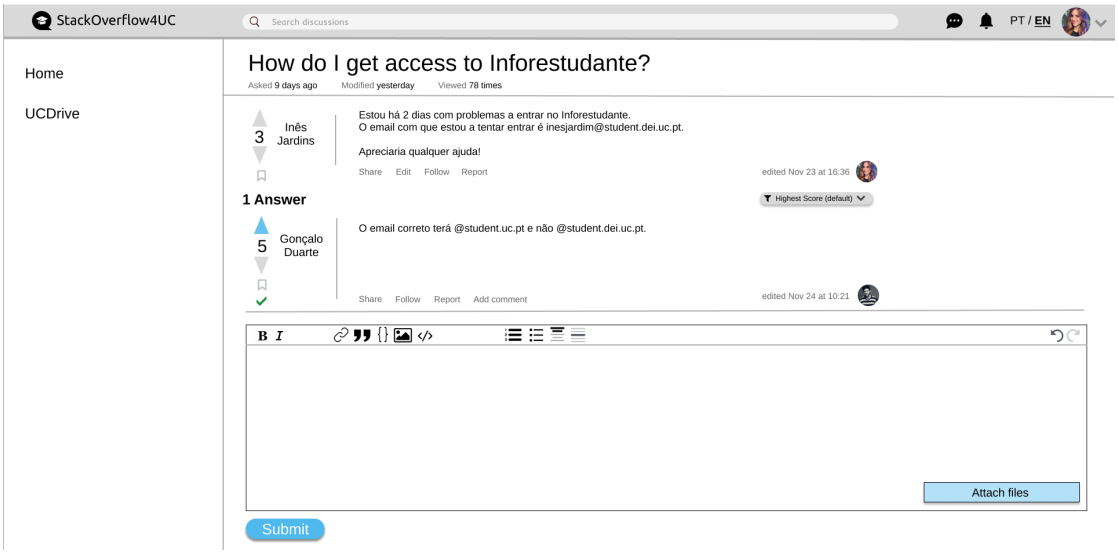

Figure 6.19: Mockup of the discussion view

This view is very familiar to the one of a Stack Overflow question. The first answer is the main question asked by the author of the discussion and the answers below are displayed by order of upvotes, by default. This behaviour can be changed by clicking on the "Highest Score (default)" filter below the name of the discussion author.

On top of this, any user can propose changes to the question asked by the author. We can see who modified the answer, the time it happened, how many times the discussion has been viewed and other informations.

Below the answers, the user can write its own answer with all kinds of text properties and submit it.

6.5 Usability Checklist

For every view mockup, we checked which Nielsen usability heuristics they follow:

View	Simple and Natural Dialogue	Speak the user's language	Minimize user memory load	Consistency	Feedback	Clearly marked exits	Shortcuts	Good error messages	Prevent errors	Help and Documentation
Homepage	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
UcDrive	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
Profile	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
Discussion	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗

- Simple and natural dialogue

  The language used is clear and straightforward. The user can easily understand

what each of the buttons can do as well as the other components.

- Speak the user's language

  Taking into account the market segment of our platform, students and professors at the University of Coimbra, we thought it would be important that the platform supports two languages. In all views, an option to change the language between Portuguese and English is possible. Even so, the language used in all the different views ensures that the user can easily understand and comprehend them without much effort or knowledge of specific terms. For example, the following words are common and easily understood: votes, answers, views, follow, share, and so many others.

- Minimize user memory load

  There are several filters and search bars that indicate exactly what it is about, as well as simple menus. For example, if the user wants to filter the display of files from the course "Algebra", they will see a dropdown button saying "Courses" on which they can click and select that course. If they don't want to waste time looking for a specific course, they can start writing the beginning of that word (ex: "Alg") and the course will appear.

  We also avoided displaying too much information in the same view, which would make the user lost and confused. A balance had to be found between displaying too little and too many options.

- Consistency

  All the views will maintain their positions and format between uses. For example, when you enter the "View discussion" page, the title and discussion's body will always appear in the same place with the same size and format, the filters available will be the same in all discussions and you will always have to scroll down to the answers.

  We are also consistent in the way we let the user filter information in either the "View discussions" or UCDrive views. Those will always be the following: course, subject, tag and filter.

  The header bar and the navigation bar are also fixed across every view, except for the ones where the user isn't registered (for example: the Login view).

- Feedback

  When a user upvotes a discussion or answer, the arrow used for the upvote is colored green to give a clear indication that it was upvoted. The same goes for a downvote, which will display a red color.

If a user wants to use a specific tag filter and starts writing its name (for example: "IP"), the tag will display the correct tags associated with that category, such as IPRP. On the other hand, writing those same words on another tag, the names displayed will be different. This is an indication that the system correctly identifies the name tags in their own category and it makes it perceptible to the user.

Another feedback can be given to the user when they are replying in a discussion. When the user submits the answer, it is inserted below the last answer and is quickly displayed to let the user know it was successfully submitted.

When receiving notifications, the number of unread notifications will be displayed on the notifications icon to let the user know how many notifications he hasn't read. The same goes for the language system, if a user chooses to switch to English they only need to click on "EN" on the header bar and the displayed language will change.

- Clearly marked exits

  Every view has a header bar and a navigation bar on the left side. The navigation bar has both the homepage and the ucdrive, so whenever a user is a specific view, they can get back to those at anytime.

  When navigating the UCDrive, if the user is inside several folders and wants to get back to a specific one, the folder names are displayed on top of the view for easy access. The user only needs to click any of those folder names to get back to its location.

  As for logging out or accessing its own profile, the user only need to click on the top right profile button and a dropdown menu will be shown to do either of those tasks. That button will be shared between any of the views.

  If a user wants to access someone's profile, they have several ways to do it. They can either search for the username on the top search bar, by clicking below a discussion's name on the "View discussions" view, by clicking below the upload file's name on the UCDrive and so many other ways.

- Shortcuts

  Shortcuts can be used to find content across the platform. One way is to use filters to quickly access specific discussions, by choosing the course, the date or the associated tag. For example, if a user wants to access a specific Python tutorial that was posted 2 years ago, instead of looking it up through every page, they only need to use the search bar or filter the page by date, tags and so on.

  Bookmarks of saved discussions or answers are also used for quick access. For

example, if a user finds an answer relevant but has no time to read at that moment, they can bookmark it and read it later.

- Good error messages

  Whenever an error is supposed to show, a text message will be displayed in red next to the location of the error. For example, if the user replies to a discussion but doesn't write anything and clicks on the "Submit" button, a red error will appear next to the button saying "Cannot submit an empty reply. Please insert text before submitting."

- Prevent errors

  Errors such as logging in without the right credentials or submitting an empty reply are prevented. For example, the user cannot create a new discussion and submit it without writing anything in the body text or the title, so the submission button won't work without these conditions being fulfilled.

- Help and Documentation

  There's no help and documentation. The platform is very familiar to already similar solutions, which don't require specific help or documentation to get used to it.

7 Non-functional Issues

7.1 Business Issues

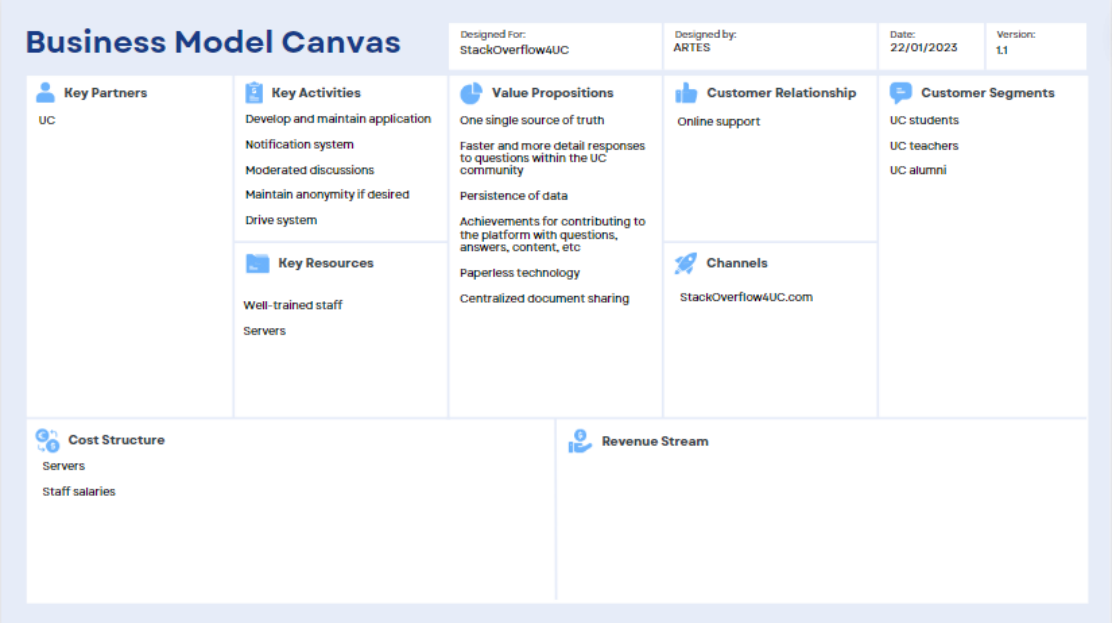

Figure 7.1: Business Model Canvas - StackOverflow4UC

For the study of the business model canvas we have:

- Key partners - We considered UC to be of utmost importance since they would provide resources for helping the platform exist/grow.
- Key activities - Some activities are: (i) Develop and maintain the application; (ii) Notification system for sending notifications to the user; (iii) Moderate discussions so as to not have inadequate content that is not correlated to the context of the application happen; (iv) anonymity for added privacy and (v) drive system for ensuring that the user has access to the files.
- Key resources - In order for the platform to exist it is imperative that we have well-trained staff working for it as well as servers.

- Value propositions - One single source of truth could help with problems such as not knowing where the information is at/stored. Given that more users are going to be exposed to a question/answer, there is a higher probability of faster and quicker responses to the questions. The persistence of data occurs because all questions and answers will be up for reading through our platform. Our achievement system permits the user to have some type of reward for questioning and answering questions thus helping our platform grow. Paperless technology is eco-friendly. Uc drive - centralized document sharing - would allow something never seen within the UC community: sharing of files through a common platform all organized in a single place.

- Customer relationship - Our relationship with the customers could be maintained through online support just in case any question or problem arose during the user's session.

- Channels - The only channel that was going to exist is our platform that would be under the domain *StackOverflow4UC.com*.

- Customer segments - Since our platform would only be visible for consultation to the UC community, our only customer segments would be UC: students, teachers and alumni.

- Cost structure - An added cost would be the servers that would be used for storing up all of the content to be added to the tables on the database(s) and, additionally, to help provide the user with a service - our platform. And, lastly, for the staff's salaries whom would be building/maintaining the platform.

- Revenue Stream - There would be no stream of money since AAC is a non-profitable organization.

Considering the Business Model Canvas, revenues streams are empty because the main purpose of the platform is to help the UC community and not to generate any income, thus no requirements were found for the business section.

7.1.1 Revenue

Looking at the business model canvas presented above, we can observe that the revenue stream and key partners sections are empty, since it is not the purpose of our platform to generate any type of revenue. Bearing in mind that our client (AAC) only intends to contribute to a better academic life for students at the university of Coimbra, they are not interested in profiting from the platform.

7.1.2 Costs

As far as costs are concerned, there are some that we have to consider

- Human resources
  - Development: For the development of the application, some human resources will be needed so that, within an estimated period of 1 year, they can collect requirements, design the application, develop the application and test it:

Role	How many	€/Month	€/Year
Requirements Engineer	1	1200	14400
Developer	2	2*1200	28800
Designer	1	1200	14400
Tester	1	1200	14400

    Total/Year : 72 000 €

  - Maintenance: After the development, it is necessary that the platform is maintained and, for that purpose, we have a team composed of:

Role	How many	€/Month	€/Year
Developer	1	1200	14400
Tester	1	1200	14400

  - Others:
    * Despite AAC already having several people who are in charge of supporting its various applications, these same people would be the ones to provide online support for stackoverflow4uc.

- Operation
  - Regarding the servers, considering that the University of Coimbra currently has 24 817 students, counting alumni and teachers, then the maximum number of active accounts would be 35,000. However, it is very unlikely that all the users are going to be using the platform at the same time. Taking that number into account, the most suitable servers would have a total of 96GB of RAM, 48 CPU's and 1920GB of storage. A Domain won't be needed since the AAC already has one and the stackoverflow4UC platform could use a subdomain.

Operation	Type	€/Month	€/Year
Server[8]	96GB RAM - 48 CPU's - 1920GB	720	8640

7.2 Legal and Regulatory Issues

The StackOverFlow4UC platform needs to follow a particular set of laws and norms. In this section, the necessary regulations that the StackOverFlow4UC platform needs to guarantee will be presented.

7.2.1 General Data Protection Regulation

The General Data Protection Regulation (GDPR) is an European regulation on data protection that became mandatory in 2018 for any system that processes information with more than 5000 individuals. The main objective is to protect any user data that the system stores or uses. The StackOverFlow4UC platform will store the personal information of all accounts and their activity inside the database. Since the platform will have more than 5000 users, it will need to obey to the GDPR regulation.

- **Consent**
  During the process of creating an account, the user will be asked to read our terms conditions and our privacy policy so as to explicitly consent to it by clicking a checkbox that confirms such acceptance. There's no need for parental consent since our platform will be destined to university students and teachers.

- **Direct identifiers**
  The user will share some personal information when creating the account: a username, their university email and a password. It's also possible to upload a profile picture.

- **Data Breaches**
  Following the regulation, if a data breach occurs and there is an integrity violation or the safety of our database is compromised, the StackOverFlow4UC should notify the data protection authority (DPA) within 72 hours. If the violation presents a risk to the users they should be notified as well.

- **Right to be forgotten**
  The user may request for the deletion of their account. In doing so, the user personal information will be deleted, and the user will lose access.

- **Right of access**
  At any point, the user can request to see all the data stored under its name and can even make a request to eliminate it if they see fit.

Following the GDPR, the user will be able to request his personal information or request for the account deletion in the profile definitions.

8 Conclusion

To create the requirements for StackOverflow4UC, we started by introducing the Problem Statement to contextualize the problem. Then, we made the State of art to familiarize the reader with multiple similar well-known solutions, both successful and unsuccessful, in order to give an idea of what an ideal solution could be. From there on, we started to apply several techniques to formulate the requirements for our problem:

- KJ method
- Use Case
- KOAL (Goal Modeling)
- Prototyping

The most useful and helpful techniques employed were the KJ method and the Use cases. In order to understand how the platform would work, we needed to interview potential users to analyse their needs and feedback, make a list of the main features, define the platform's users and how they perform the tasks, as well as to describe the interaction in the whole system using context diagrams and use cases.

The most noticeable problem we came across was understanding this interaction, i.e, how each requirement was going to be executed, how each user's role and tasks were going to be defined, along with taking into account several pre-conditions and potential scenarios that could emerge.

It is also important to notice that our project tends towards the *closed* side of the axis, given that our project does not have anything new to add to this domain - an academic forum.

Moreover, the other methods were also useful to complement and complete our work, but we didn't feel like they were as essential as the Affinity Diagram and the Use cases because of the project's more *creative* and *dynamic* nature. For example, the Goal Model wasn't the most successful technique in our project due to its more logical and systematic approach.

Bibliography

[1] https://pt.wikipedia.org/wiki/Stack_Overflow

[2] https://uniarea.com/sobre/

[3] https://en.wikipedia.org/wiki/Bloomfire

[4] https://en.wikipedia.org/wiki/Reddit

[5] https://www.quora.com/Why-did-Yahoo-Answers-fail

[6] https://www.speeli.com/why-google-answers-failed/

[7] https://bootcamp.uxdesign.cc/jakobs-10-usability-heuristics-their-importance-6f0ddec8c938

[8] https://www.linode.com/pricing/

[9] https://www.proalojamento.pt/

[10] *https://www.sciencedirect.com/topics/computer-science/flow-model#:~:text=A%20flow%20model%20gives%20you,the%20result%20of%20user%20actions.*

9 Appendix

9.1 Goal Model

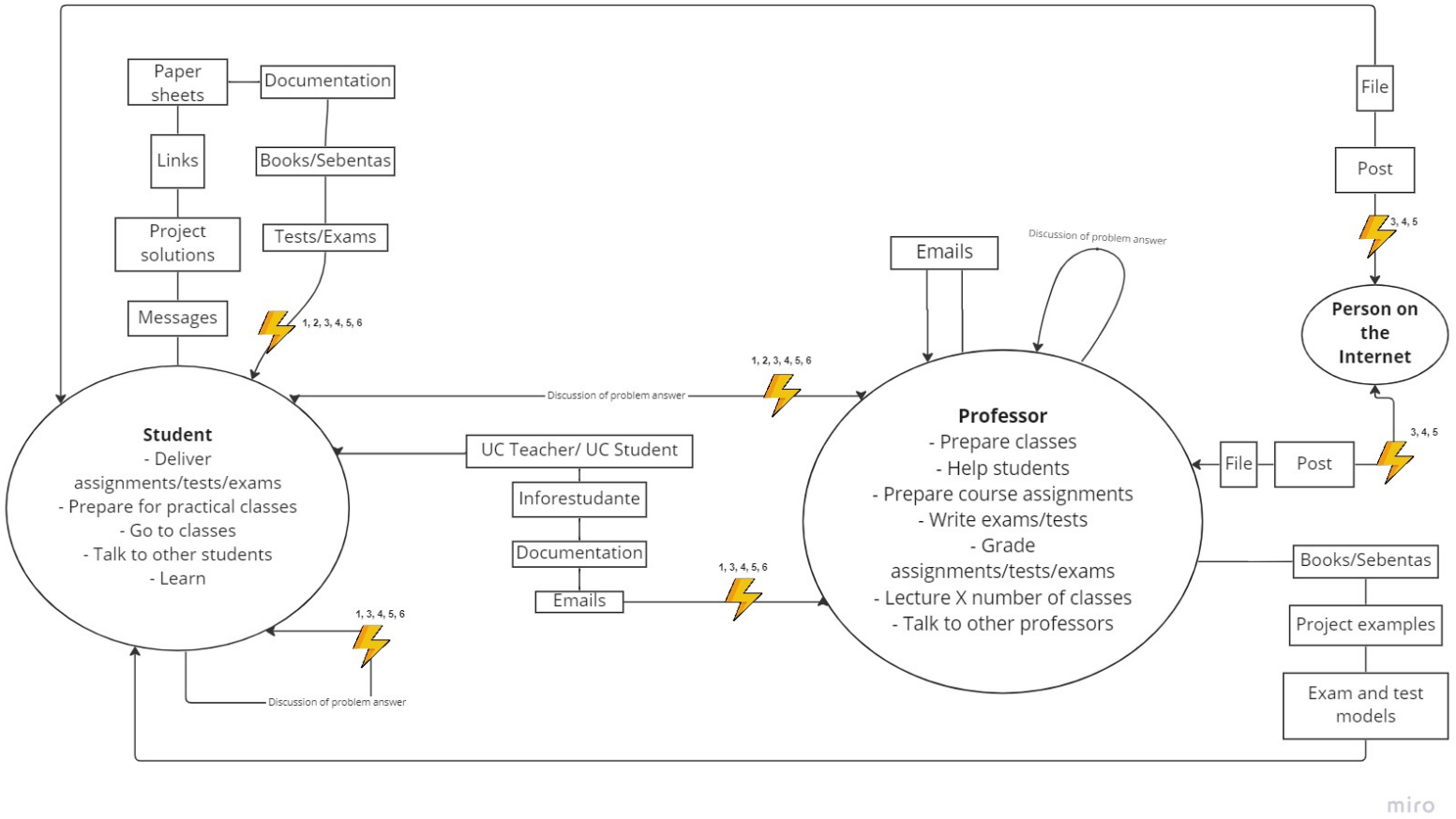

Figure 9.1: Flow Model - StackOverflow4UC

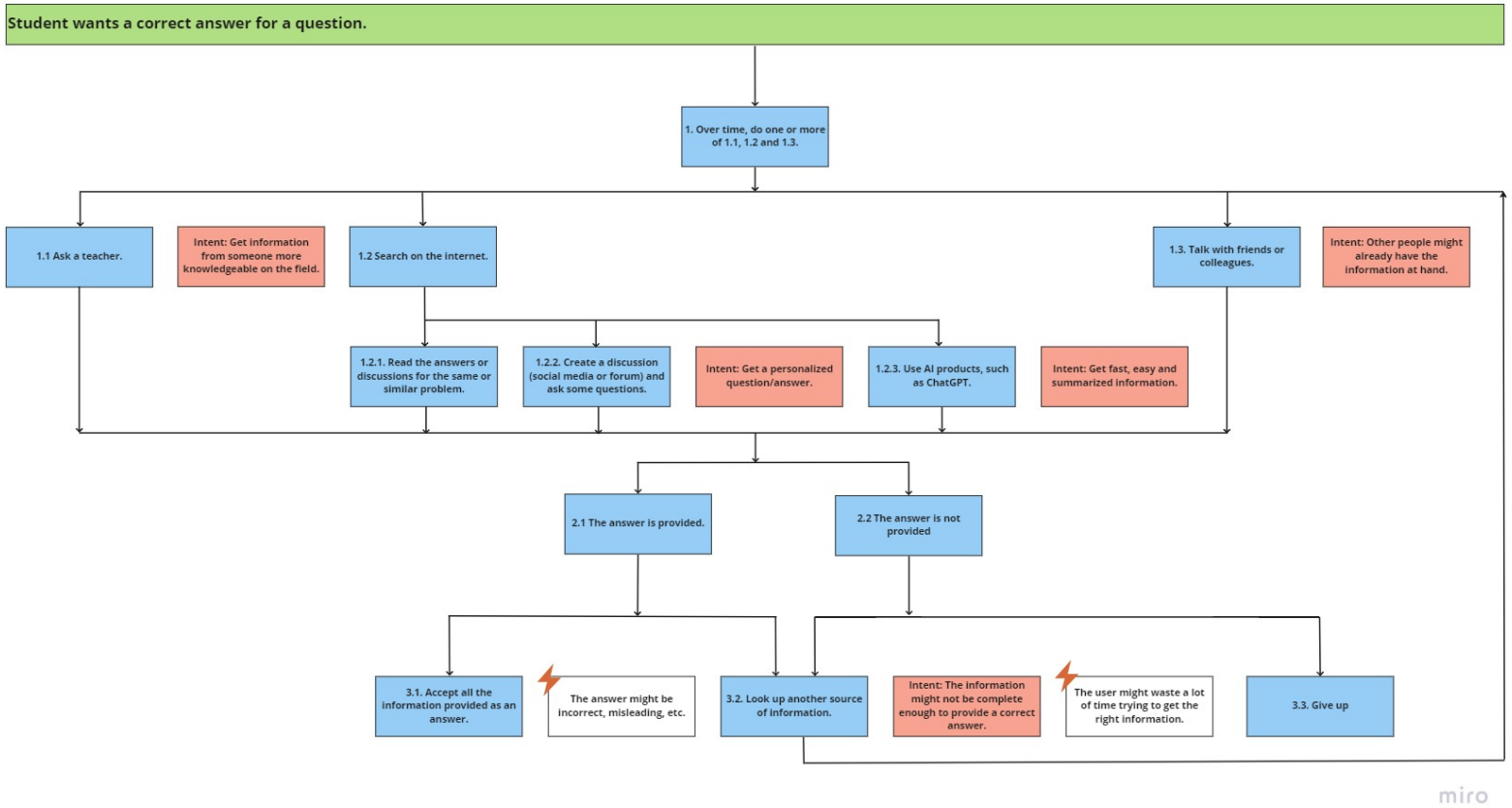

Figure 9.2: Sequence Model - StackOverflow4UC

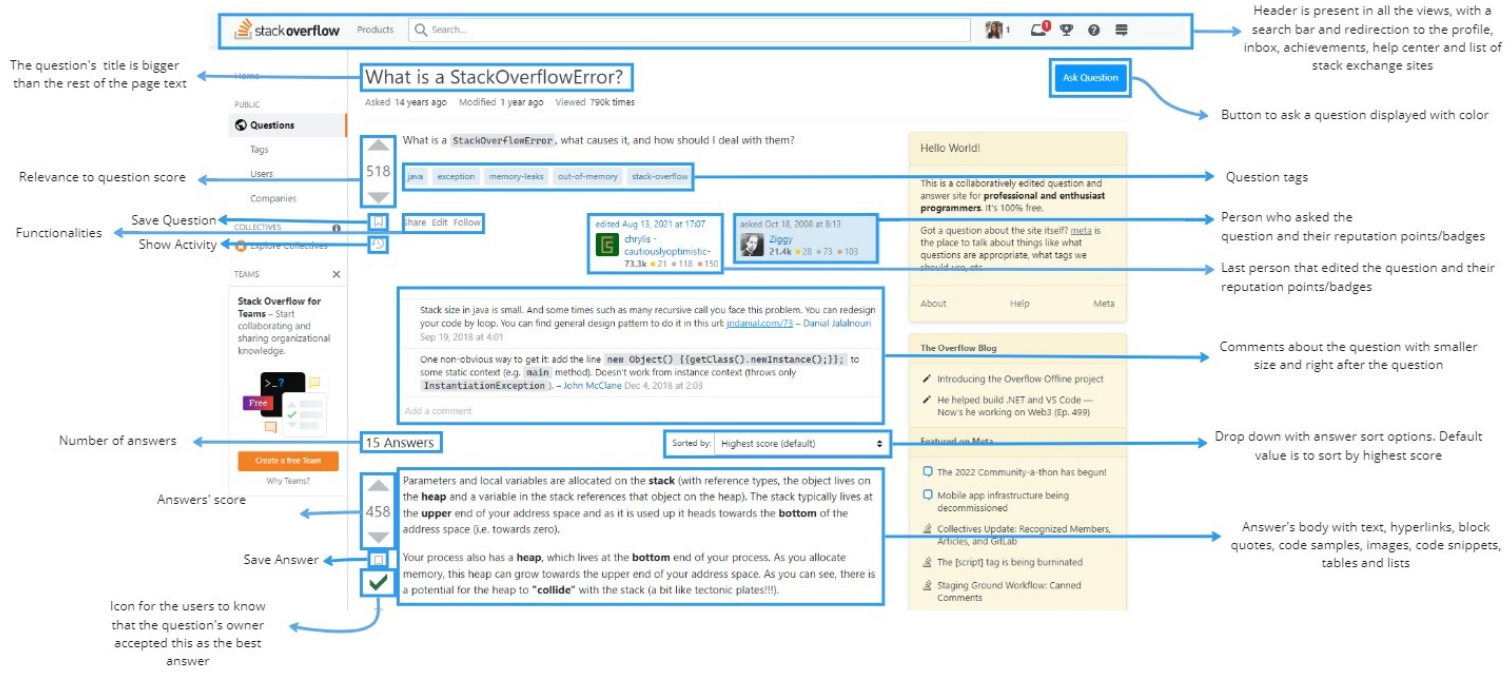

Figure 9.3: Artifact Model - StackOverflow4UC

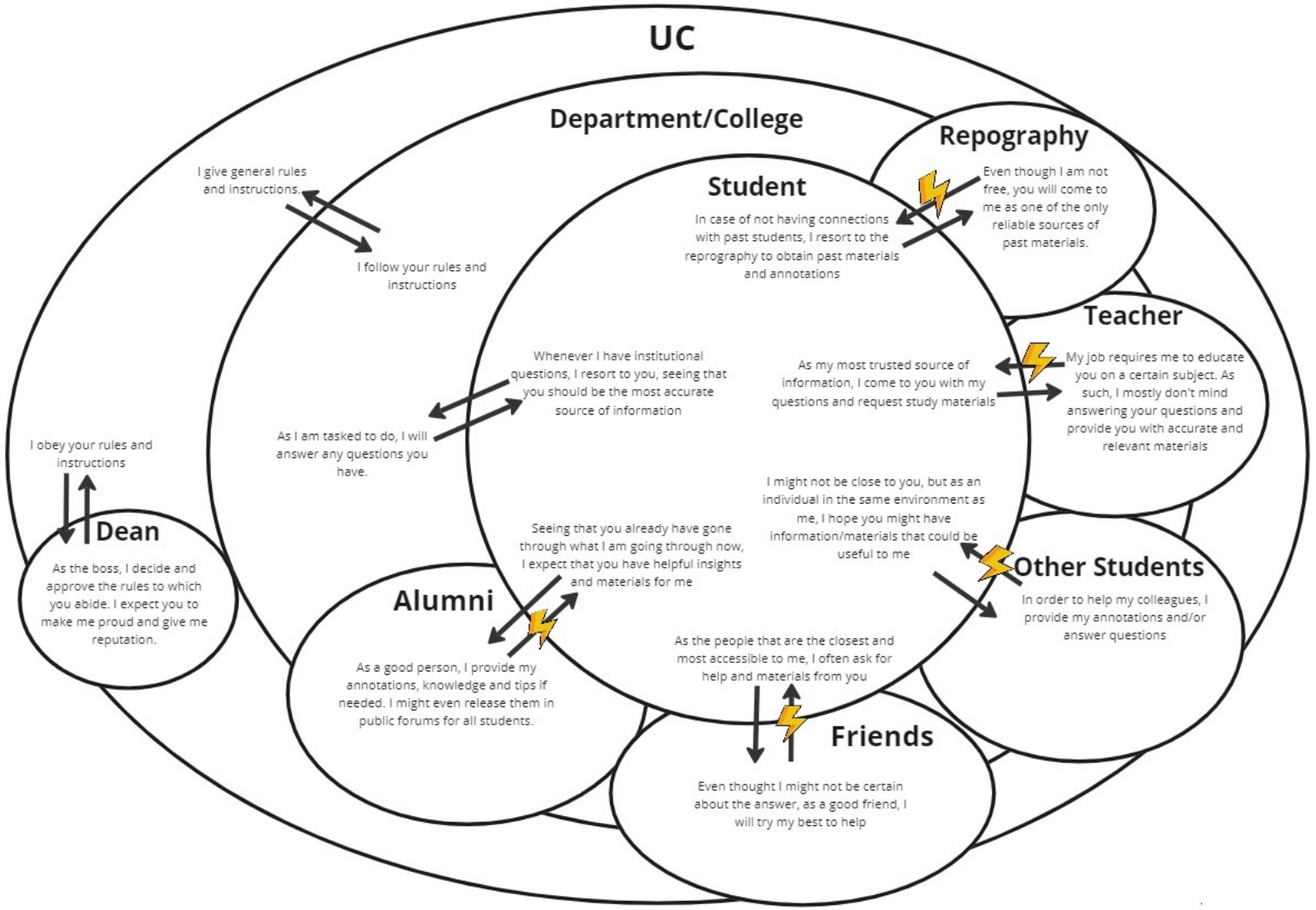

Figure 9.4: Cultural Model - StackOverflow4UC

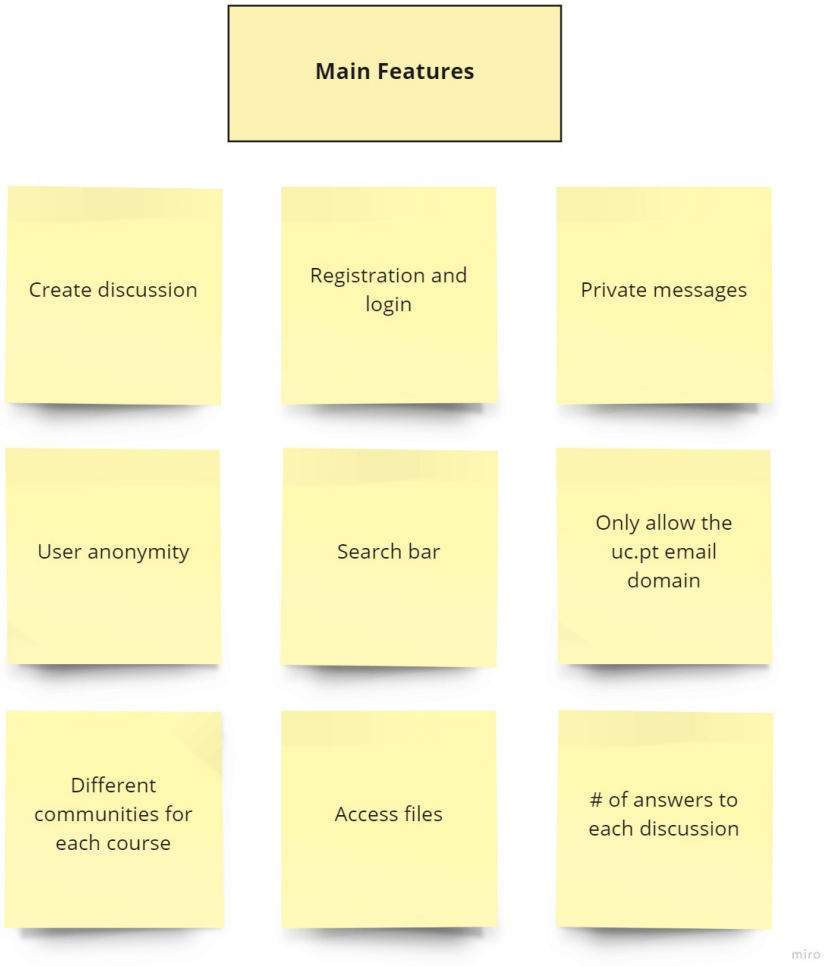

Figure 9.5: Affinity Diagram StackOverflow4UC - Main Features

Figure 9.6: Affinity Diagram StackOverflow4UC - Profile

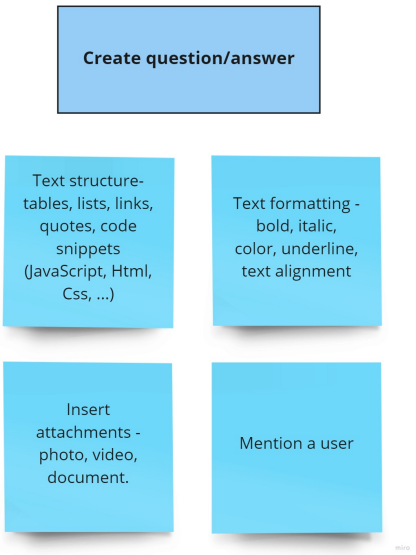

Figure 9.7: Affinity Diagram StackOverflow4UC - Create question/answer

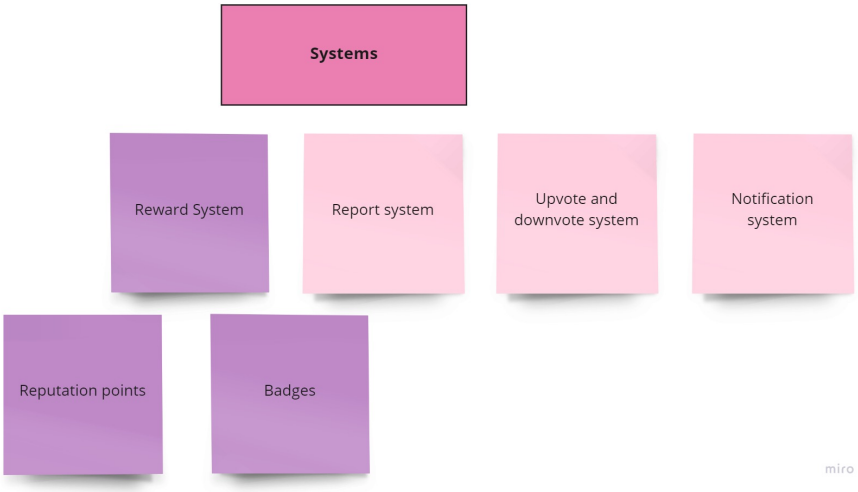

Figure 9.8: Affinity Diagram StackOverflow4UC - Systems

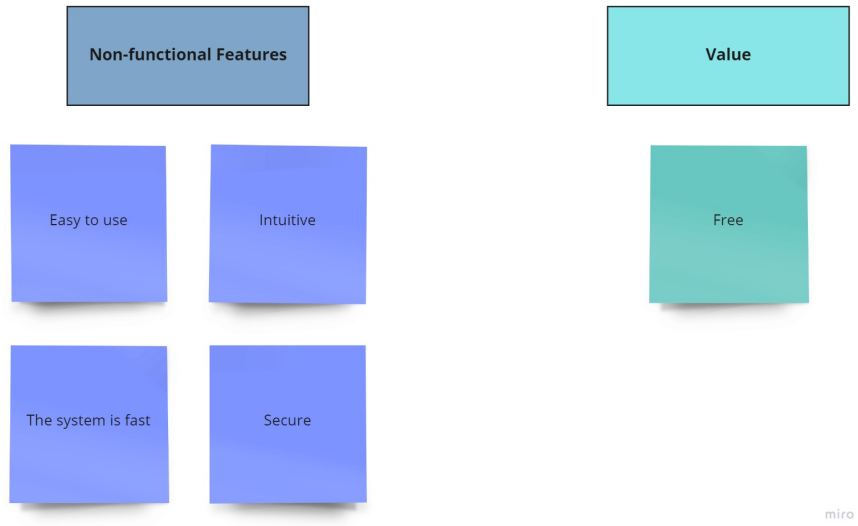

Figure 9.9: Affinity Diagram StackOverflow4UC - Non-funcional Features Value

Figure 9.10: Affinity Diagram StackOverflow4UC - Discussion Page

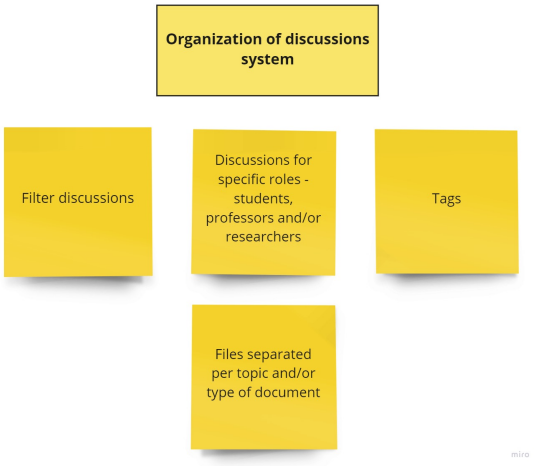

Figure 9.11: Affinity Diagram StackOverflow4UC - Organization of discussions system

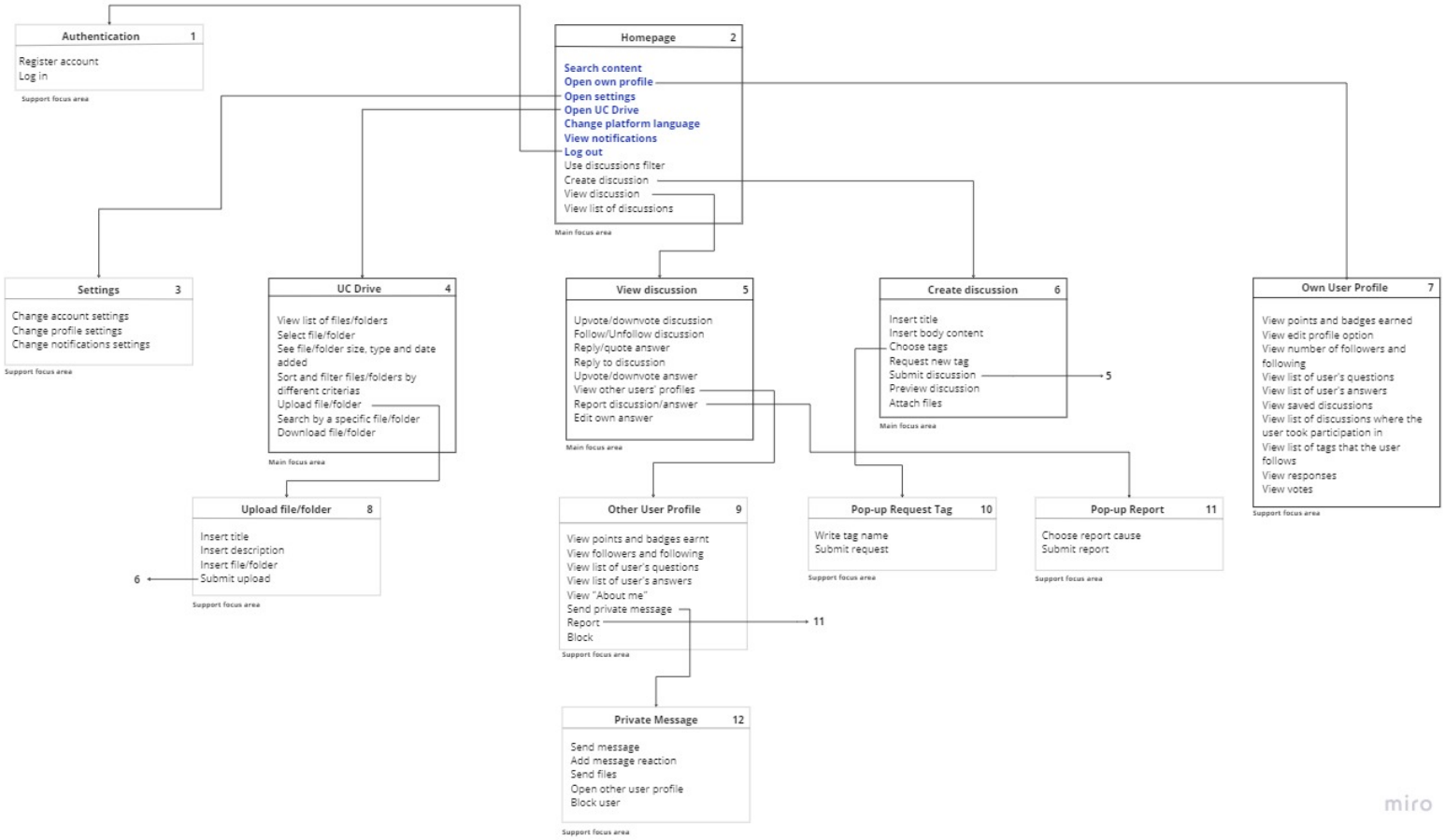

Figure 9.12: User Environment Design (UED)