Politecnico di Torino

Human Computer Interaction [02JSKOV]

Final Report - QuadraMente

Academic Year 2023/2024

Lorenzo Bertetto S309618 s309618@studenti.polito.it
Davide Colaiacomo S313372 s313372@studenti.polito.it
Daniele De Rossi S314796 s314796@studenti.polito.it
Giulia Di Fede S307746 s307746@studenti.polito.it

Contents

1 Project and group overview **3**

2 Problem overview **3**

3 Needfinding **4**
- 3.1 Domain of interest . . . 4
  - 3.1.1 Questions . . . 4
- 3.2 Interviews . . . 5
  - 3.2.1 Interview 1 - Lead researcher . . . 5
  - 3.2.2 Interview 2 - Immediate Researcher . . . 6
  - 3.2.3 Interview 3 – Domain expert . . . 6
  - 3.2.4 Interview 4 – Lead researcher . . . 7
  - 3.2.5 Relevant artifacts . . . 8
- 3.3 Needs extraction . . . 9
  - 3.3.1 Deep user needs . . . 10
- 3.4 Identifying solutions . . . 10
- 3.5 Pre-feedback final proposed solution . . . 12
- 3.6 Post-feedback final proposed solution . . . 12

4 Tasks and Storyboard **13**
- 4.1 Task Analysis . . . 13
- 4.2 Storyboard . . . 13

5 Low-Fi Prototypes **15**
- 5.1 Modalities . . . 15
- 5.2 Lo-Fi Prototypes . . . 16
  - 5.2.1 “ChatBot” Prototype . . . 16
  - 5.2.2 “Desktop Application” Prototype . . . 24
  - 5.2.3 Connection with storyboard, project goal and three tasks . . . 25
- 5.3 Low-fi prototype: Desktop Application . . . 25
- 5.4 Heuristic Evaluations . . . 27
- 5.5 “ChatBot” Prototype - Heuristics . . . 33
- 5.6 “Desktop Application” prototype - Heuristics . . . 36
- 5.7 Selected prototype . . . 39

6 Medium-fidelity prototype **40**

7 High-fidelity prototype **42**
- 7.1 Tools . . . 42
- 7.2 Significant Screens . . . 42
  - 7.2.1 Chat . . . 42
  - 7.2.2 Document . . . 42
  - 7.2.3 Popover . . . 42
  - 7.2.4 Documentation . . . 43
- 7.3 Limitations . . . 43

7.3.1 Pre-stored data . . . 43
7.3.2 Hard coded features . . . 43
7.3.3 Other limitations . . . 43

8 Usability testing **45**
8.1 Preparation and run . . . 45
8.1.1 Methodology . . . 45
8.1.2 Equipment . . . 45
8.1.3 Material . . . 45
8.1.4 Tasks . . . 46
8.1.5 Tests . . . 47
8.2 Results . . . 49
8.2.1 List of potential changes . . . 49

9 Conclusions **52**
9.1 What we learnt . . . 52
9.2 Group feedback . . . 52

10 Appendix **53**

Project and group overview

The goal of this report is to build a comprehensive reflection on the evolution of our project, undertaken during the **Human-Computer Interaction [02JSKOV]** course at **Politecnico di Torino** for the academic year **2023/2024**. Together, we are **QuadraMente**, comprising:

- **Lorenzo Bertetto - s309618**
- **Davide Colaiacomo - s313372**
- **Daniele De Rossi - s314796**
- **Giulia Di Fede - s307746**

We are going to detail a retrospective analysis of our progression, reflecting on the insights gained, the challenges overcome, and the results obtained. Our aim is not just to recount our work, but to delve into the depths of our collective learning experience and underline what we will take away from our participation in the aforementioned course.

Our project culminated into a prototype that was eventually refined to more address the interaction between the researcher and AI. We propose, "PaperCraft", a Text-editing helper designed to assist researchers by improving text quality and style by enhancing the tone and giving editing suggestions, it will be able to help the researchers by boosting their productivity in their research process of writing their article.

Problem overview

The main topic of our 5-months spanning activity was the world of **human-AI interaction**.

From decision-making to prompting creativity, we have been presented with how AI-driven interfaces are reshaping human experiences in recent years; consequentially, the starting reflection point has been specifically the role of **Human-Computer Interaction** (HCI) in encouraging user trust and facilitating AI adoption, culminating in a prototype for an application aiming at materializing this initial considerations.

Each phase of this project will be thoroughly covered, from the early ideas to the final result, in order to shed a light on how and why we have chosen our path.

Needfinding

3.1 Domain of interest

Our journey with this project began with a seemingly straightforward question: **which domain should we select?** To make an informed decision, we engaged in numerous collaborative discussions, recognizing that AI, rather than a mere challenge, presents a versatile solution applicable across various fields. Our considerations spanned diverse domains, from providing support for elderly individuals and healthcare professionals struggling with psychological needs, to assisting schoolteachers with their daily instructional tasks.

Going on with our reflections, it became evident that our focus was in leveraging AI to enhance support for individuals in their day-to-day lives. After much consideration, we arrived at a unanimous decision to center our efforts on the following domain: **Support for Computer Science researchers in writing articles, searching for sources, checking and processing information**.

Our confidence in this choice derived also from the collective expertise within our group. We acknowledged the many challenges faced by researchers and were intrigued by the prospect of developing an AI-powered application to alleviate some of the time-consuming tasks they encounter.

3.1.1 Questions

In order to extract pertinent information essential for investigating the selected domain, we carefully selected the following questions to be posed to our interviewees. Our main goal was to obtain a **comprehensive understanding** of the challenges encountered within the academic community, encompassing perspectives from both researchers and reviewers as domain experts.

Questions for the researchers

- *What is your research field?*
- *What led you to be a researcher?*
- *Where do you find inspiration about the main ideas of your research work?*
- *What strategies do you use that you think could make you more productive?*
- *How do you manage the finding of information needed to write an article?*
- *What challenges do you meet during the process of writing an article?*
- *If you can't find the information you need. What do you do?*
- *Tell me about the last time you had to write an article.*
- *How much attention to details is needed when writing an article?*
- *How do you organize and process the information you have gathered?*

- *Do you think that any aspects of the current research process you follow could be improved or changed? If yes, what?*
- *What methods do you use to verify the information you have found?*
- *Have you ever found yourself collaborating with other people who are not researchers? If yes, which kind of companies?*
- *How do you collaborate and interact with other researchers? What is your relationship with them?*
- *Have you ever been in the situation in which the deadline is getting closer? How did you cope? If not, what strategies...*
- *What else should we have asked about?*

Questions for the reviewer

- *What aspects do you consider when you review an article?*
- *If you have ever found yourself refusing an article, why did you do it?*
- *Are there any differences between different conferences you have reviewed for?*
- *How do you handle situations where a paper may not be a perfect fit for a conference's theme, but is still high quality and relevant to the field?*
- *How do you ensure that your review is fair and unbiased?*
- *What role does the quality of writing play in your evaluation of a paper?*
- *How do you assess the novelty and significance of the research presented in a paper?*

3.2 Interviews

We deemed it essential to conduct interviews with four distinct individuals with experience in the scientific research field. This approach aimed to provide a comprehensive and insightful perspective on the domain, offering valuable insights into the daily challenges faced by researchers. Our interviewees included one **Immediate User** (a researcher or a PhD student), two **Lead Users** (expert researchers or Associate/Ordinary professors), and one **Domain Expert** (a reviewer). Our objective was to identify the common needs of individuals working as researchers, thus establishing a robust foundation for our project.

3.2.1 Interview 1 - Lead researcher

Our first interview was with a seasoned lead researcher, renowned for their expertise in writing and publishing scientific articles. Conducted online using the Audacity tool for audio recording, the interview was facilitated by *Lorenzo Bertetto* as the interviewer, with *Giulia Di Fede* providing assistance. Through this initial discussion, we gained valuable insights into the dynamics of the research profession.

Primarily, we learned that **creativity** plays a pivotal role in research, not only during the writing process but also in the initial stages of idea generation for new articles. Attempting to suppress creativity could compromise the authenticity of research. Additionally, **simplifying**

complex concepts to make them more accessible poses a significant challenge for researchers. We were informed of instances where papers failed to convey their content effectively due to their complexity.

Furthermore, it became evident that **time** is a constant constraint for researchers. Our interviewee shared their experiences of time shortages when acquiring new knowledge for research or searching for essential information. They emphasized the potential benefits of tools that streamline certain aspects of the research process, thereby alleviating cognitive burdens and enhancing productivity.

Interview 1 - Key quotes

1. *"Writing a scientific paper is somewhat like creating a painting."*
2. *"Trying to fit research into a box might compromise its authenticity and integrity."*
3. *"Inspiration is an unconscious thing; one has to feel unconsciously mature."*

3.2.2 Interview 2 - Immediate Researcher

Our second interview featured an immediate researcher, a dedicated PhD student immersed in the process of writing scientific articles for various projects. Conducted in person at Politecnico di Torino and utilizing a simple audio recorder for further reflection, the interview was facilitated by *Davide Colaiacomo* as the interviewer, with *Daniele De Rossi* offering assistance. Much like our initial interview, this session provided invaluable insights into the field we aimed to explore.

We gained a profound understanding of the challenges inherent in the scientific article writing process. Our interviewee highlighted the multiple challenges encountered at each stage, necessitating constant **refactoring and reorganization**. Once again, the issue of **time** scarcity emerged as a significant obstacle throughout the endeavor.

Furthermore, our interviewee emphasized the escalating **workload and pressure** as publication deadlines get closer. They shared instances where incomplete research led to the failure to publish articles, particularly problematic given the specific formatting requirements of different conferences. Recognizing the importance of **self-care**, our interviewee stressed the need to strike a balance between diligence and counter-productivity, underscoring the importance of avoiding burnout in the pursuit of academic excellence.

Interview 2 – Key quotes

1. *"(Writing articles) is a process that never ends."*
2. *"In one of our papers, we couldn't compare with anyone because there was no one who worked on such things."*
3. *"You have to respect certain rules which may be different from conference to conference, (like for example) a different template."*

3.2.3 Interview 3 – Domain expert

For our third interview, we had the privilege of engaging with an expert reviewer boasting extensive experience in evaluating articles for various conferences and journals. Conducted online with the assistance of the OBS tool for audio recording, the interview was facilitated by *Giulia Di Fede*

as the interviewer, with *Lorenzo Bertetto* providing support. Unlike our previous interviews, this session offered valuable insights from a distinct perspective.

This interaction shed light on the multifaceted considerations researchers must weigh, particularly regarding the audience on the receiving end of their work. The reviewer's role was elucidated, emphasizing the precise examination of technical aspects within papers prior to potential approval.

The interviewee began with highlighting the importance of the **novelty** that an article should convey, underlining that imperfect work can still be accepted if it manages to bring innovation and new open issues in the topic of the research. However, in the process of providing the results, the reviewer stressed on the contrast between the **rigor** required by a scientific paper and the **urgency** as deadlines come close, which can cause oversights that lead to the rejection of the paper, such as the failure of explaining the contribution of their study.

Interview 3 – Key quotes

1. *"The article should convey novelty and should cover urgent topics."*
2. *"Rejections can be beneficial, leading authors to improve."*
3. *"It's important to know that even imperfect studies can open new questions."*

3.2.4 Interview 4 – Lead researcher

We had our fourth and final interview with an experienced lead researcher in the field of data analysis, who is well-known for their major contribution in the scientific community, spanning from organizing conferences to writing novel work. This was performed online, using OBS to record the audio. Davide Colaiacomo facilitated the interview and *Daniele De Rossi* provided support.

Given the extensive experience of the interviewee, they pinpointed the evolution of the **methodology of studying** existing literature, stating that, with the advent of search engines, it is unrealistic nowadays to unintentionally produce similar outcomes; however they still posed the issue that similar papers can be accepted by the same conference under the condition that they follow different methodologies.

Another important issue that emerged was that researchers should be **creative**, but they usually have limited time to find the right inspiration and cultivate their own creativity; this is a crucial point, because they highlighted the importance of the **novelty** needed by the research community.

As with other researchers, they stated that **urgency** can be decisive in the act of the final sprint of writing an article; however a trade-off between urgency and rigor emerges, as the researcher has to strive for accuracy even under limited time conditions. Similarly, rigor also introduces a contrast with **accessibility**, which is important in other kind of contexts.

Finally, other minor aspects that we extracted from the interview are related to the collaboration between researchers and the importance of the repeatability of the experiments.

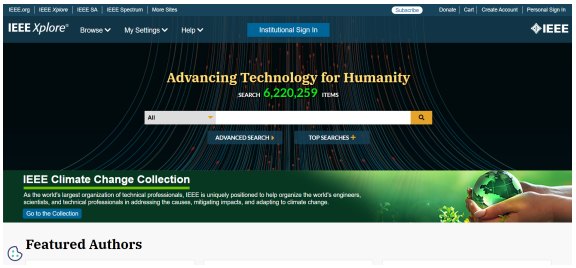
Interview 4 – Key quotes

1. *"The goal is to combine tasks and not have too many."*
2. *"If I can't find what I need, it's because it doesn't exist."*
3. *"When you need to determine if the idea is good or not, you need to see what's currently trending."*

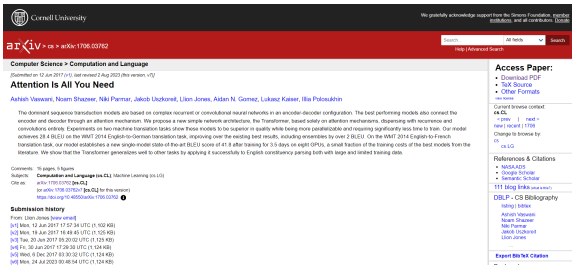
3.2.5 Relevant artifacts

Before moving on, we should consider the most important tools that our interviewees have brought to our attention; pinning up these artifacts has been useful for us to understand what kind of support scientific researchers tend to seek.

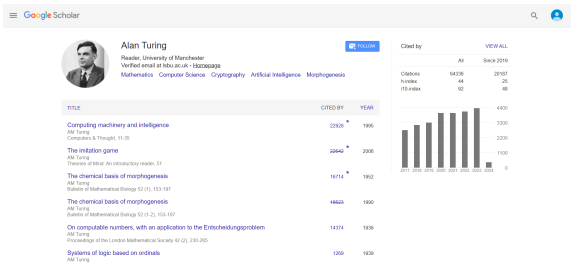
- **IEEE Xplore digital library** in Figure 3.1a is a research database for discovery and access to journal articles, conference proceedings, technical standards, and related materials on computer science, electrical engineering and electronics, and allied fields.
- **arXiv** in Figure 3.1b is a free distribution service and an open-access archive for nearly 2.4 million scholarly articles in the fields of physics, mathematics, computer science, quantitative biology, quantitative finance, statistics, electrical engineering and systems science, and economics. Materials on this site are not peer-reviewed by arXiv.
- **Google Scholar** in Figure 3.1c is a freely accessible web search engine that indexes the full text or metadata of scholarly literature across an array of publishing formats and disciplines.
- **LaTeX** is a software system for document preparation. When writing, the writer uses plain text as opposed to the formatted text found in word processors like LibreOffice Writer.
- **Overleaf** in Figure 3.1d is a collaborative cloud-based LaTeX editor used for writing, editing and publishing scientific documents. It partners with a wide range of scientific publishers to provide official journal LaTeX templates, and direct submission links.
- **Anaconda** is a distribution of the Python for scientific computing, that aims to simplify package management and deployment. It contains useful libraries for graphing and data elaboration.
- And other tools that help with collaboration between researchers and other team members (Microsoft Teams, Zoom...).



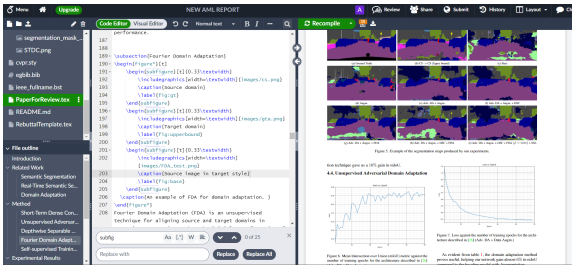
(a) IEEE Xplore digital library



(b) arXiv.org e-Print archive



(c) Google Scholar



(d) Overleaf LaTeXEditor

Figure 3.1: Some tools suggested by the interviewees.

3.3 Needs extraction

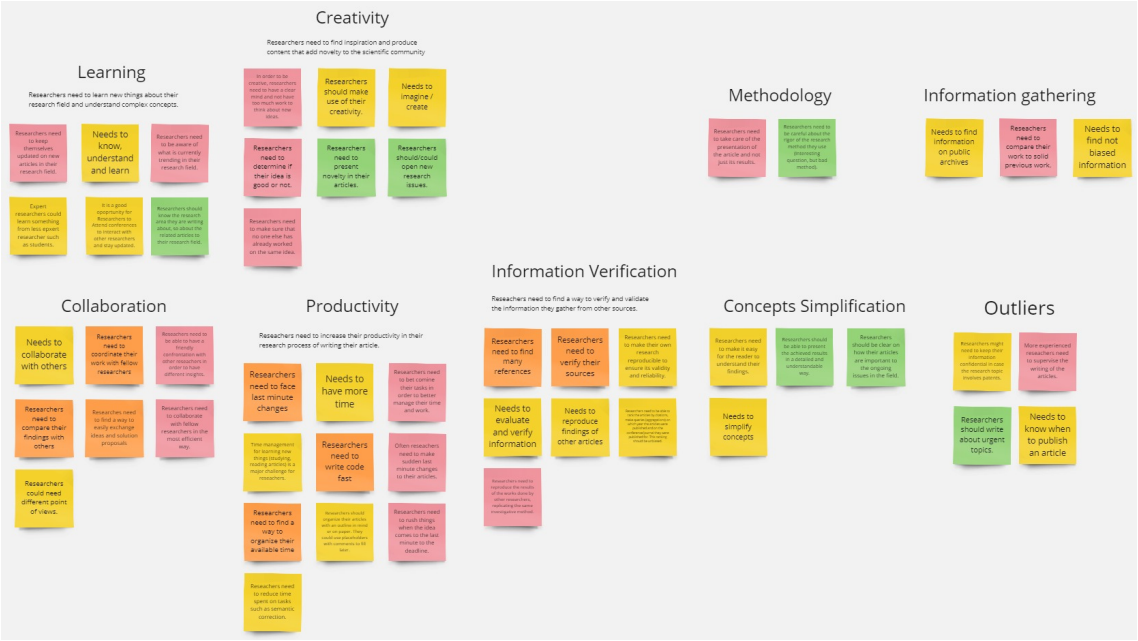


Figure 3.2: Miro: a digital collaboration platform used for our brainstorming session.

With the interviews completed, it came the proper moment to focus on some brainstorming sessions, in order to understand what are the most urgent necessities for scientific researchers; in other words, we had to focus on extracting the **needs** of our stakeholders.

In order to carry out the aforementioned task, we conducted a long documentation process, capturing a spectrum of needs arising from our interviews. Whether basic or complex, we tried to unpack each insight and translate it into user-centric needs. With the support of Miro's platform (Figure 3.2), we concentrated on the organization and categorization of our findings.

In our brainstorming session [1], we extracted all the emerged needs, connecting them to each interview by the use of colors (Green: reviewer, Yellow: lead researcher 1, Orange: immediate researcher, Pink: lead researcher 2), then, in order to facilitate clearer understanding and organization, we grouped together the similar ones into distinct categories:

- **Learning**: This category underlines the needs related to the process of finding new knowledge and staying updated with the latest developments in the field of research.
- **Creativity**: This category focuses on the needs associated with the novelty of the article and research questions.
- **Collaboration**: This category underlines the requirement for effective teamwork and communication among scientific researchers.
- **Productivity**: Refers to the optimization of the researcher's time, resources, and efforts towards achieving their research goals efficiently and effectively.

[1] To better visualize the brainstorming process, please follow this link to see the work done on Miro: `https://miro.com/app/board/uXjVNXTalJU=/?share_link_id=202886498359`

- **Information verification**: Information verification needs address the necessity for good methodologies to assess the reliability, and validity of scientific information and research findings.
- **Methodology**: This category comprises the requirements for methodologies to design, conduct, and evaluate scientific research in a rigorous way.
- **Concept Simplification**: Concept simplification needs relate to the necessity for clear, concise, and accessible explanations of complex scientific concepts.
- **Information Gathering**: This category refers to the needs for comprehensive and efficient methods of finding and accessing the information relevant to the research goal.

We also gathered some outliers which we realized didn't fit in any of the other categories, but were still important to note.

3.3.1 Deep user needs

By examining the needs and the categories they belonged to, we extracted the four user needs we felt embraced the widest possible range of concerns and preferences by our stakeholders:

- *Researchers need to learn new things about their research field and understand complex concepts.*
- *Researchers need to find inspiration and produce content that add novelty to the scientific community.*
- *Researchers need to increase their productivity in their research process of writing their article.*
- *Researchers need to find a way to verify and validate the information they gather from other sources.*

3.4 Identifying solutions

After having unanimously identified the user needs for our project, we felt ready to come up with some **solutions** tailored for our users; again, by using Miro, we employed a structured approach to brainstorming involving a 5-minute timer to individually generate as many ideas as possible. Then, we adopted a voting process using post-it notes, wherein each of us was allocated three votes to designate the ideas we deemed most valuable; for the sake of impartiality, we accepted not to vote for our own ideas.

- Researchers need to **learn** new things about their research field and **understand complex concepts**.
  - *Daily updates about chosen research topics.*
  - *Online courses to learn about new things.*
  - *Mentorship with more experienced researchers.*
  - *Summaries of papers that can be interesting so that reading doesn't take too much time.*
  - *Conversation that allow for the exchange of new information among researchers.*
  - *Explainability grades evaluated before the publication of the article.*

- Researchers need to find **inspiration** and produce content that add **novelty** to the scientific community.
  - *Generation of creative content from reliable sources.*
  - *Comparison between titles and/or abstracts of articles to see if there are similar ones.*
  - *Merging multiple articles to identify potential points of contact that can be developed.*
  - *Search for 'open questions' that can be found in the articles.*
  - *Suggestions for editing the article.*
  - *Public "news and updates" service visible to all researchers updated daily and always available.*
  - *Researchers get automatic notified about articles and conferences about their field.*
- Researchers need to increase their **productivity** in their research process of writing their article.
  - *Aggregate articles by conference, topic, author.*
  - *Generate text given a prompt for the scientific question.*
  - *Reduce time spent on tasks such as semantic correction.*
  - *Plan the research steps beforehand.*
  - *Scheduler of daily activities according to a best-fit solution.*
  - *Generator of specific programming language starting from pseudo-code.*
  - *Researchers get automatic notified about articles and conferences about their field.*
- Researchers need to find a way to **verify and validate** the information they gather from other sources.
  - *Crowd-sourcing on scientific articles.*
  - *Researching opinions and consensus on scientific articles.*
  - *Filtering system for certified information.*
  - *Automated source validator.*
  - *A place where researchers can express their opinions on articles.*
  - *Cross-validation of multiple articles that deal with things belonging to the same field.*
  - *Grading system of scientific articles according to specialized equipes.*
  - *Plagiarism check.*
  - *Indicators that tell the reader how many people have successfully reproduced the paper and how many tried without success.*
  - *Reduce testing time in case of algorithm.*
  - *Reliability grade to the article, to the author, to the journal.*

Through a series of voting rounds, ideas were systematically eliminated based on the number of received votes, ultimately culminating in the identification of the solutions mostly tailored to our users' specific needs.

- *Public "news and updates" service visible to all researchers updated daily and always available.*

- *Scheduler of daily activities according to a best-fit solution.*
- *A place where researchers can express their opinions on articles.*
- *Reliability grade to the article, to the author, to the journal.*

3.5 Pre-feedback final proposed solution

The voting rounds culminated into our final solution, a merge between the first and the fourth idea: a public **news and updates** service visible to all researchers updated daily and always available which adds a reliability grade to the article, to the author or/and to the journal. We supposed this solution would have been able to help the researchers by giving them inspiration and by keeping them updated, while the reliability grade would still help with their need to verify information.

SciFeed
(Scientific Feed)
"Daily Updates and Reliability Grades for Research Articles"

3.6 Post-feedback final proposed solution

After receiving some valuable feedback from our topic tutor, our final solution was eventually refined to more address the interaction between the researcher and AI. We supposed a **text-editing helper** designed to assist researchers by improving text quality and style by enhancing the tone and giving editing suggestions will be able to help the researchers by boosting their productivity in their research process of writing their article.

PaperCraft
(Crafting a Paper)
"Fast edits and tone tuning for your research paper"

Tasks and Storyboard

4.1 Task Analysis

Once our definitive solution was picked, we spent some time reasoning on what a researcher should actually be able to do with our potential application; the goal was to select three tasks, categorized by means of their complexity, to dive further into our prototypes development. This brainstorming process led us to come up with the aforementioned tasks as follows:

- **SIMPLE**: Request Editing Suggestion *to receive info about quality.*
- **MODERATE**: Edit Section automatically *to improve its quality.*
- **COMPLEX**: Refactor tone preferences *to tune the application's suggestions.*

The main reason that prompted us to accept these tasks as definitive lies in what we extrapolated from the needfinding experience: recurrent themes touched upon by the researchers we have had the opportunity to interview were the lack of **time** and excessive **pressure**. These issues were highlighted as crucial in contributing to low-quality text in articles, which is an issue that could be overcome with our proposed solution.

Since researchers may find difficult to cope with time to write down an effective, easy to understand, reliable article, we decided to base our idea of **PaperCraft**, starting with these tasks, on providing a fast, easy to use and fully customizable experience to the final user with the help of AI, so to give them the right inspiration and support with upcoming deadlines, and generally to provide a high quality standard (or custom tone) to research/scientific articles, as emerging from the goals of the tasks.

4.2 Storyboard

Figure 4.1: Storyboard: PaperCraft used for text quality improvement

After having identified the three main tasks for our project, we focused on the next step, which consisted of creating a simple **storyboard** to better visualize how a potential user could approach our hypothetical application. The storyboard shows a user struggling with a section of a scientific paper of theirs, while the deadline for delivery is close; as a consequence, they opt to rely on **PaperCraft** to request some editing suggestions and eventually get the section modified and revisioned by the AI. In the end, the researcher is able to complete the refactoring work with less difficulties and can dedicate his spare time to other activities.

It is to be noted that the storyboard was never intended to explicate the technical steps for a user to achieve the tasks, but only a generic interaction to communicate the fundamental purposes of the prototype.

The simple and moderate task appear in the storyboard we provided; it proposes the use of an AI-powered application on a PC (laptop or desktop) to enhance the quality and tone of written articles or research papers. The application identifies the sections of a paper that need refinement and the user consequentially seeks suggestions from the system; despite the awareness that a researcher in any kind of industry could play the role, in the final tile the context is narrowed down to academic researchers just for the sake of the example.

Thinking about the needs identified after the interviews, we are confident, as a **strength**, that the storyboard effectively illustrates the time constraints faced by researchers in terms of productivity and how an external tool could relieve them of some urgency-related concerns. On the other hand, we realize that a potential **weakness** lies in the absence of addressing the complex task, which is confined to customizing app settings, rather than aligning with the user's goal and interaction with AI.

Low-Fi Prototypes

5.1 Modalities

In this section we introduce the different alternatives we came up with to implement our idea. Satisfying the need of researchers is difficult, but we managed to come up with 3 different alternatives:

- **ChatBot**: Since the use of chatbots evolved rapidly in numerous fields in recent years, especially with the advent of ChatGPT, we propose to explore the modality represented by a chatbot application. A chatbot could be an interactive solution with the potential for real-time assistance. It could offer valuable suggestions to enhance text quality and style, providing researchers with a conversational and user-friendly experience.

  Pros:

  - The chatbot experience implies a certain level of collaboration, therefore this could potentially lead to a more interactive and conversational experience.
  - A chatbot application could indeed provide real-time assistance thereby enhancing efficiency and productivity in different fields of research.

  Cons:

  - A chatbot may be limited in terms of the user interface. In fact it may not be the optimal choice for detailed text editing tasks that require extensive formatting and layout customization.

- **Desktop Application**: A desktop application could offer a familiar platform for researchers. It could give a more robust user interface with more features which ensure a nice set of tools for text editing, enhancing the overall quality and style of the content.

  Pros:

  - A Desktop Application could definitely be a more familiar platform for researchers. In fact, tools like Overleaf are well known in the academic community and have gained popularity among researchers for their user-friendly interfaces and collaborative features.
  - Desktop applications often offer a wide range of advanced functionalities tailored to the specific needs of researchers. In fact, in the process of writing an article, a wider range of custom features might be necessary in order to enhance the quality of the article.

  Cons:

  - A Desktop application may lack real-time interaction compared to chatbots. Real-time interaction is a key feature of chatbots that allows researchers to receive immediate assistance, feedback, and guidance as they work on their projects.

- **Augmented Reality Mobile Application**: An augmented reality mobile application provides an immersive experience, allowing for creative approaches to text editing. While it may not align with the primary goal, it introduces a unique way to the process of text editing.

  Pros:

– An Augmented Reality Mobile Application could be an interesting experimental approach to text editing, potentially revolutionizing how researchers interact with and manipulate textual content.

Cons:

– An Augmented Reality Mobile Application might not align with the primary goal of improving text quality and style. In fact, this immersive experiences could divert attention from the core aspects of writing improvement.

– An Augmented Reality Mobile Application may be less practical for text-heavy tasks, in fact it might make the task of text manipulation effectively more challenging.

After a careful analysis of each possible modality described above we decided to focus on the two main following prototypes.

5.2 Lo-Fi Prototypes

In this section we present the two prototypes developed by our team, the *Chatbot*[1] and the *Desktop application*[2] . We display some of the most important features. For a more detailed view of the flow please refer to the dedicated links in the footnotes.

5.2.1 "ChatBot" Prototype

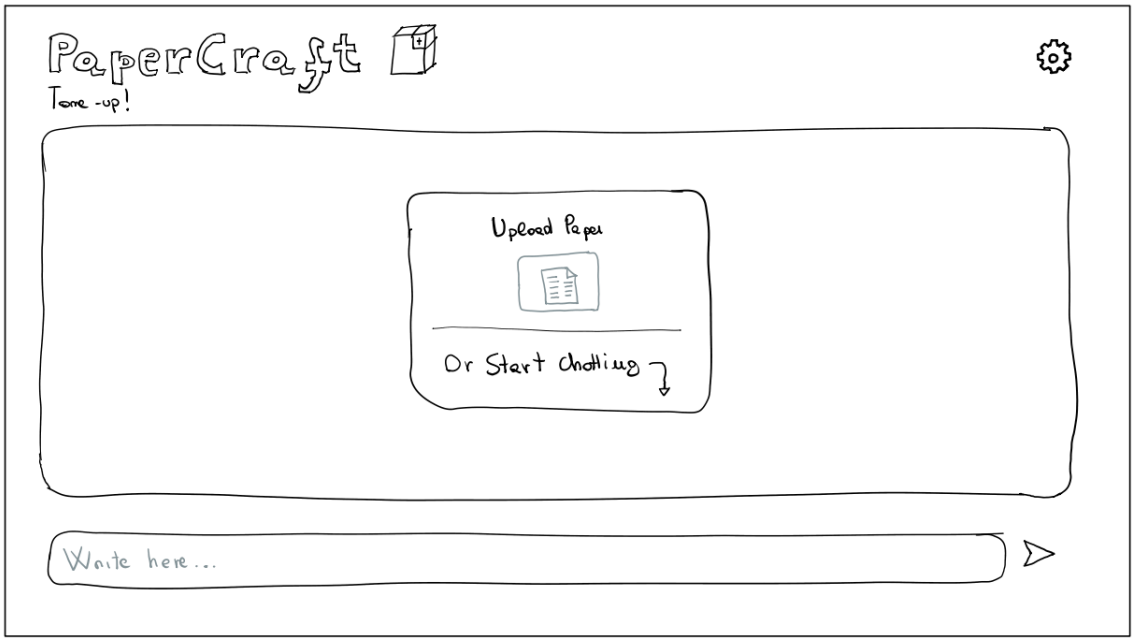


Figure 5.1: Landing page of the application, presenting the user the possibility to diverge into one of the two modalities.

[1]Link to the Chatbot Prototype: `https://github.com/polito-hci-2023/QuadraMente/blob/main/A2/Low_Fi_Prototype_Chatbot.pdf`

[2]Link to the Desktop application Prototype: `https://github.com/polito-hci-2023/QuadraMente/blob/main/A2/Low_Fi_Prototype_Application.pdf`

As mentioned above, we supposed a chatbot application would be able to foster a certain level of Human-AI collaboration. Guided by the context of the conversation, chatbots powered by Large Language Models (LLMs) are able to understand user queries and offer contextual information in real-time. Therefore, the following prototype tries to exploit the aforementioned characteristics.

To experiment two different kinds of interactions and to leverage the capability of the LLMs to learn from the context, this application adopts two main modes of interaction: the **Chat** mode and the **Document** mode.

As shown in Figure 5.1, before diverging into the two modes, the application's landing page presents itself with three main components:

- A **textarea** described by a *"Write here..."* placeholder. In the textarea, users can write their messages. Intuitively, the user could submit the message by either pressing on the *Enter* button on their keyboard or clicking on the *Arrow* button.
- A **chatbox** in which the chatbot's responses are displayed together with the user's messages, allowing for a coherent and chronological view of the conversation. A centered component gives the user a choice to upload a paper or start chatting. This is the component which gives the application the possibility to diverge into the two cited modes.
- A **navbar** which present's the application's name and slogan, together with a settings button which empowers users to customize their experience according to their preferences and requirements.

"Chat" mode

In the chat mode, the LLM would take as input the whole context of the conversation. Moreover, with the use of a thoroughly tuned Natural Language prompt, the LLM could potentially be able to understand the goal of the user and therefore distinguish between a text which has to the corrected and a request for information or a statement. Below, we present the mode with the following figures.

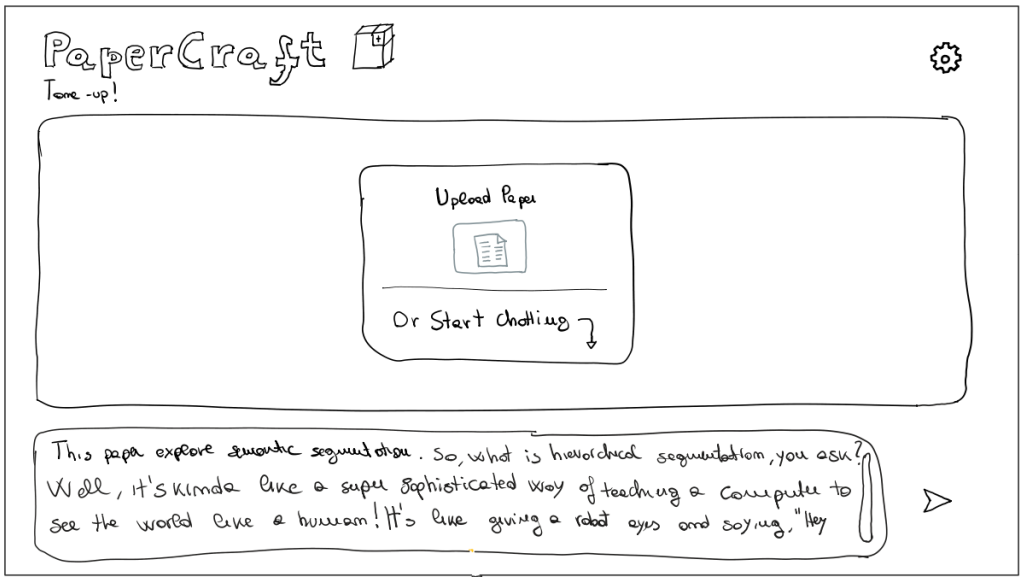


Figure 5.2: Behaviour of the textarea. If the text entered by the user is too long, the textarea expands in order to give the user a better view of their message.

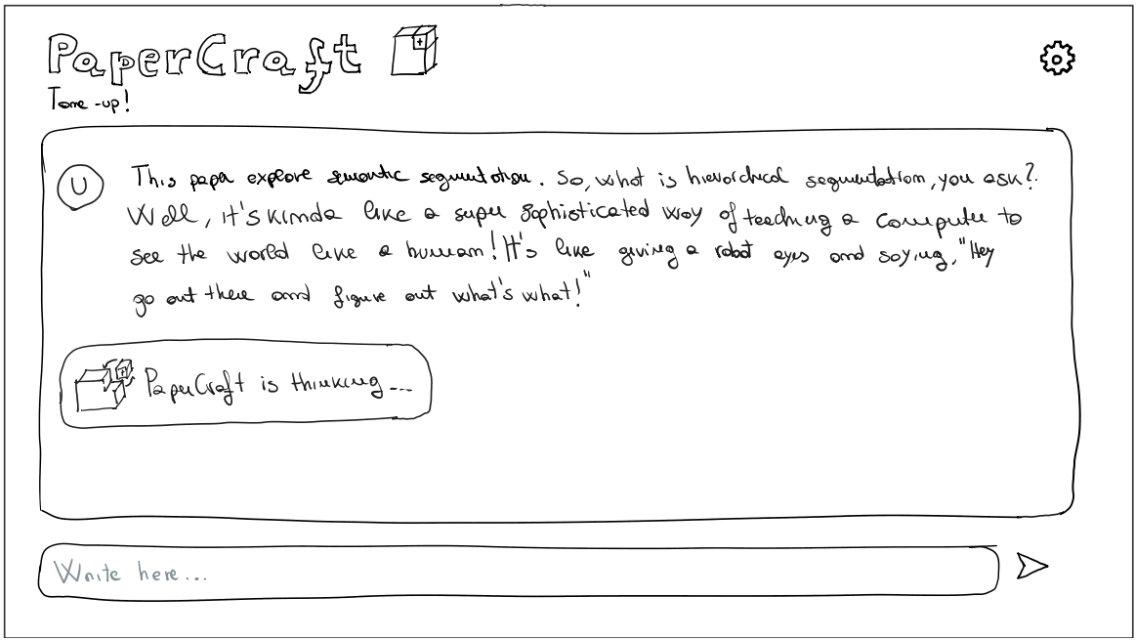


Figure 5.3: Submit Action. After submitting the message, the latter is inserted into the chatbox together with the user icon. The *"Papercraft is thinking..."* component suggests the user to wait for the analysis of the LLM on the message, it contains a dynamic icon which indicates that the system is actively analyzing the message.

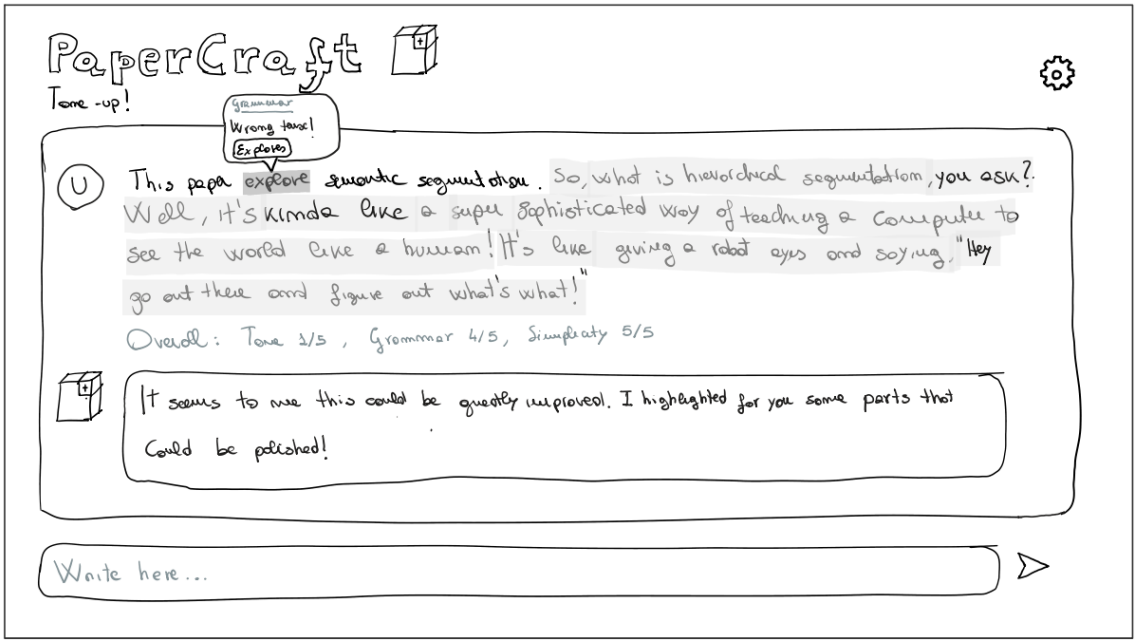


Figure 5.4: PaperCraft answers with a suggestion. If the message of the user is recognized to be a request for corrections, the portions of text PaperCraft considers to be improved are highlighted. Now the message is equipped with the metrics *Creativity, Grammar and Simplicity* statistics which are used to evaluate the text provided by the user. By hovering on the portion of text to be corrected, a popover presenting the correction together with an explanation is opened. The user can click on the corrected word to apply the correction.

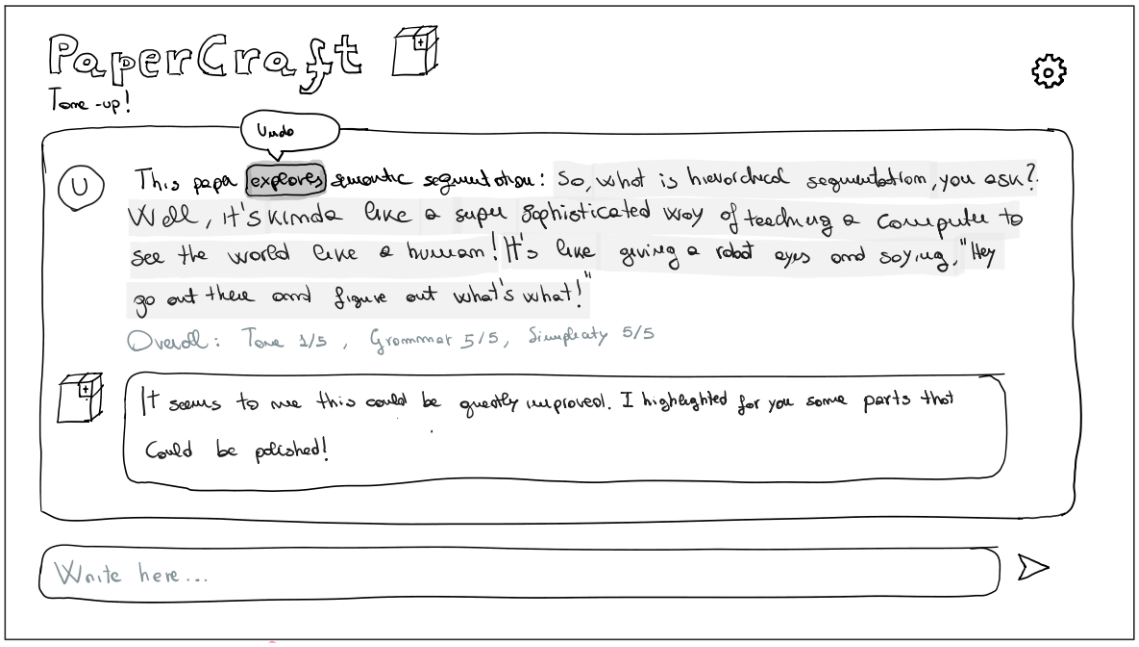


Figure 5.5: After selecting the correction, the related statistics (in this case: *Grammar*) are updated. The corrected word is shown with a solid border instead of a highlight. Then, the user would be able to hover on the corrected word to undo the correction.

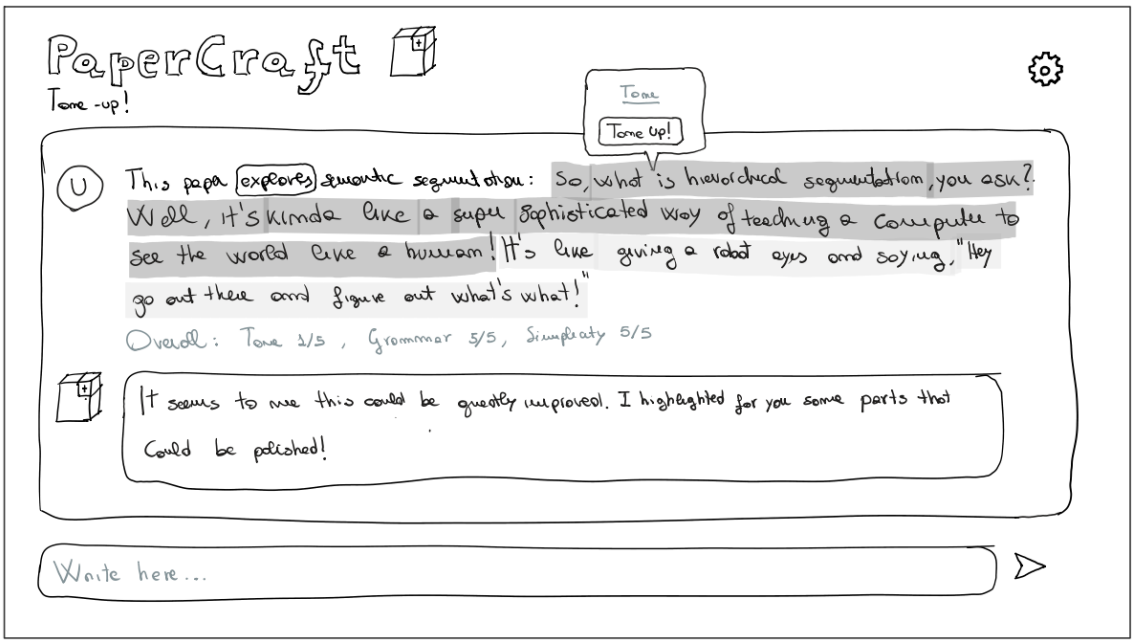


Figure 5.6: Hovering on the first segment, the application offers instead a tone correction by showing a popover with the *Tone Up!* button.The user can click on the button to improve the tone of the highlighted segment.

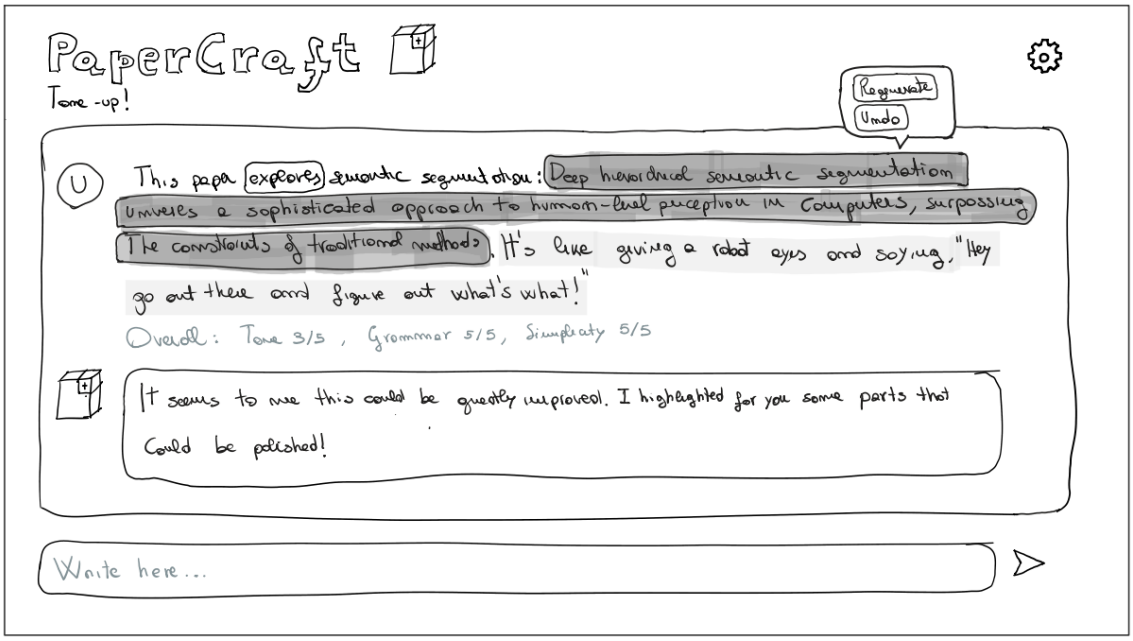


Figure 5.7: The corrected text is again presented with a solid border. By hovering on it, a popover will enable the user to choose between the *Undo* action, which gives the user the chance to revert to the original version, or *Regenerate* to get a rephrased correction.

"Document" mode

The document mode was instead purposefully developed with a more specific viewpoint which takes into account the whole structure of the document. In this case, we could hypothesize that the LLM would have as input context the whole uploaded content. From the interactivity point of view, the user would follow a more structured approach, typical of a scientific setting.

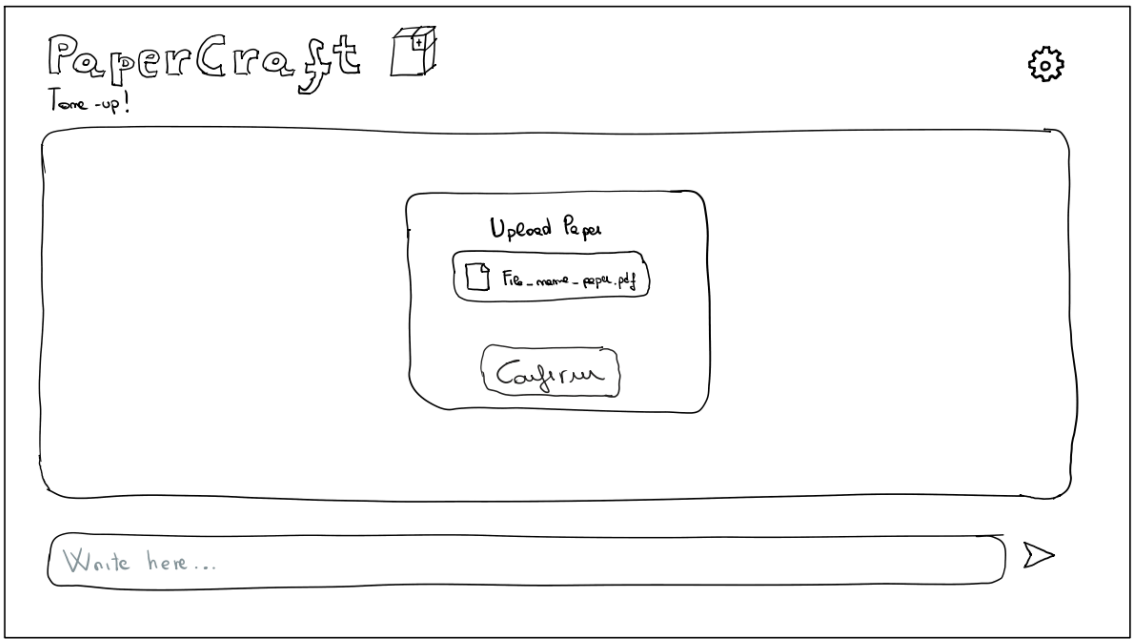


Figure 5.8: Upon uploading the document, the user is asked for confirmation to proceed.

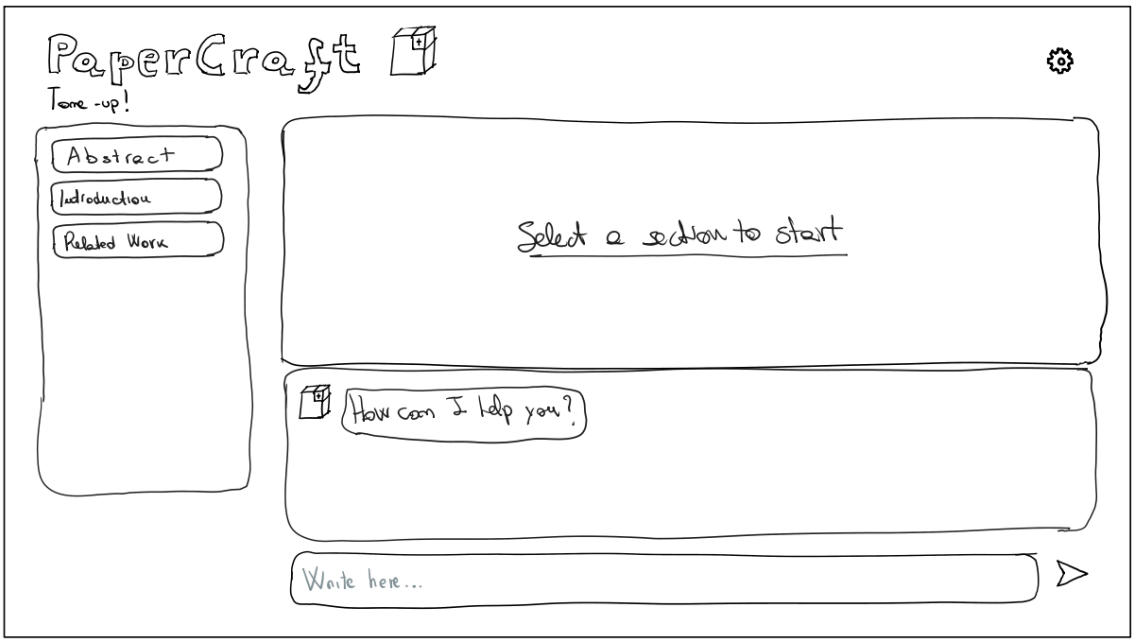


Figure 5.9: Upon confirmation, the user interface transforms and two new key components appear: the *Sidebar* which enables navigation between the sections of the document and the section view, presented with the placeholder *Select a section to start* in order to guide the user. In this modality, the messages of the chatbot are placed in a dedicated area below the section component. The chatbot politely asks the user *"How can I help you?"* in order to prompt the user to ask for suggestions.

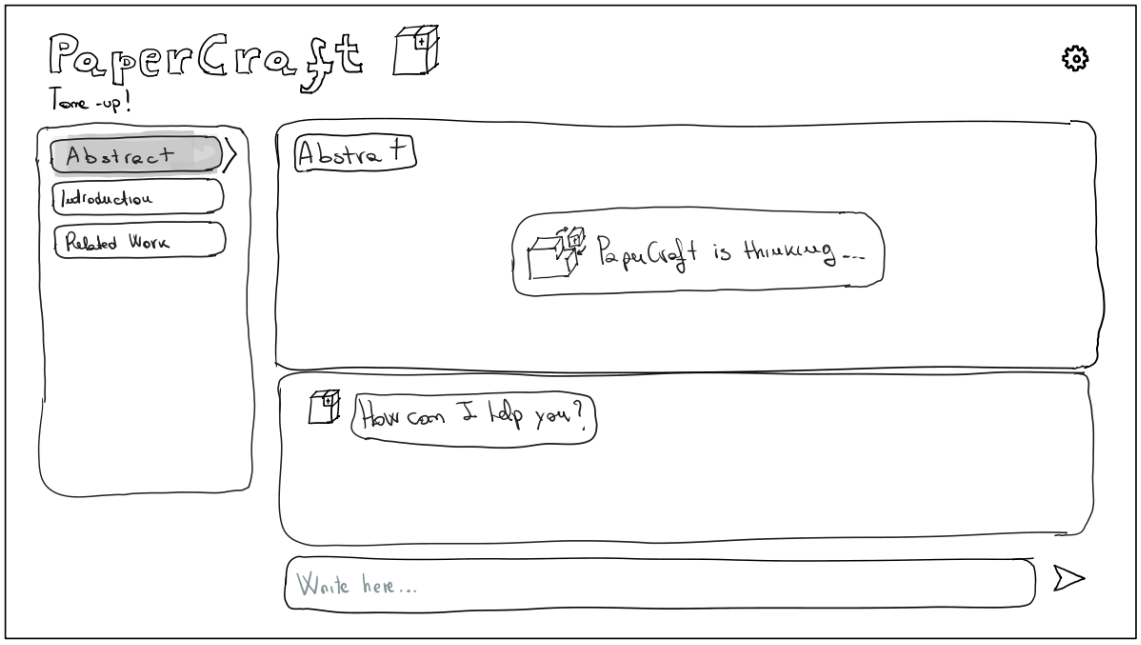


Figure 5.10: At the selection of a section in the sidebar, the title of the section appears in the section area together with a loading dialog which suggests that the application is analyzing the section.

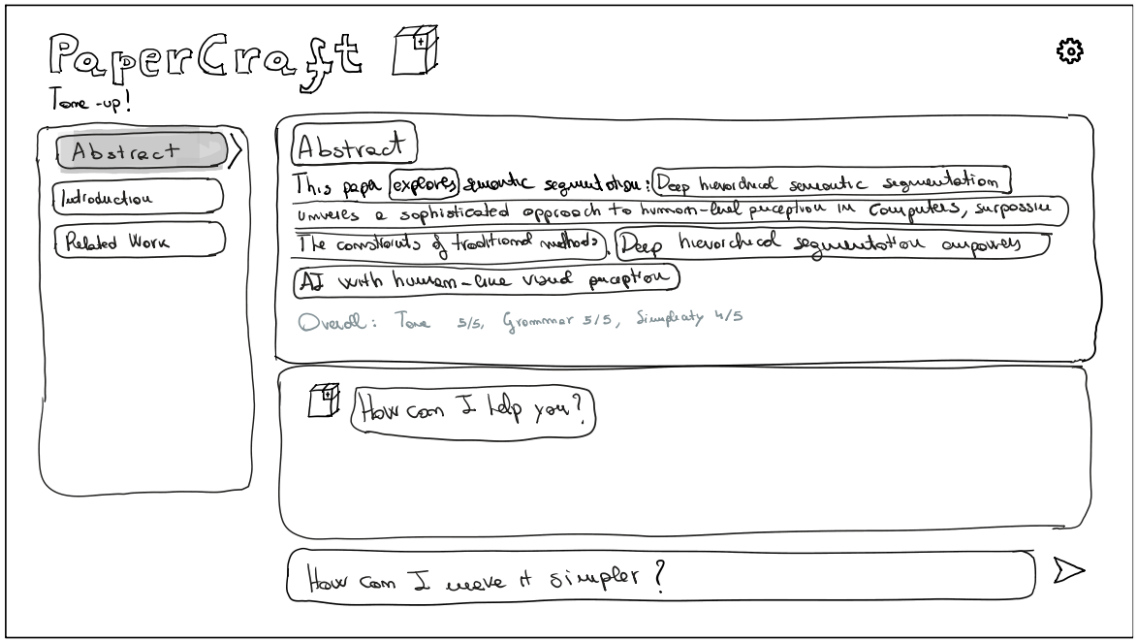


Figure 5.11: After the section has been loaded, if the *Auto-modify* option is enabled in the settings, the user is presented with a corrected text. However, the user is still free to eventually ask for more suggestions by either writing in the textarea or by hovering on the corrections.

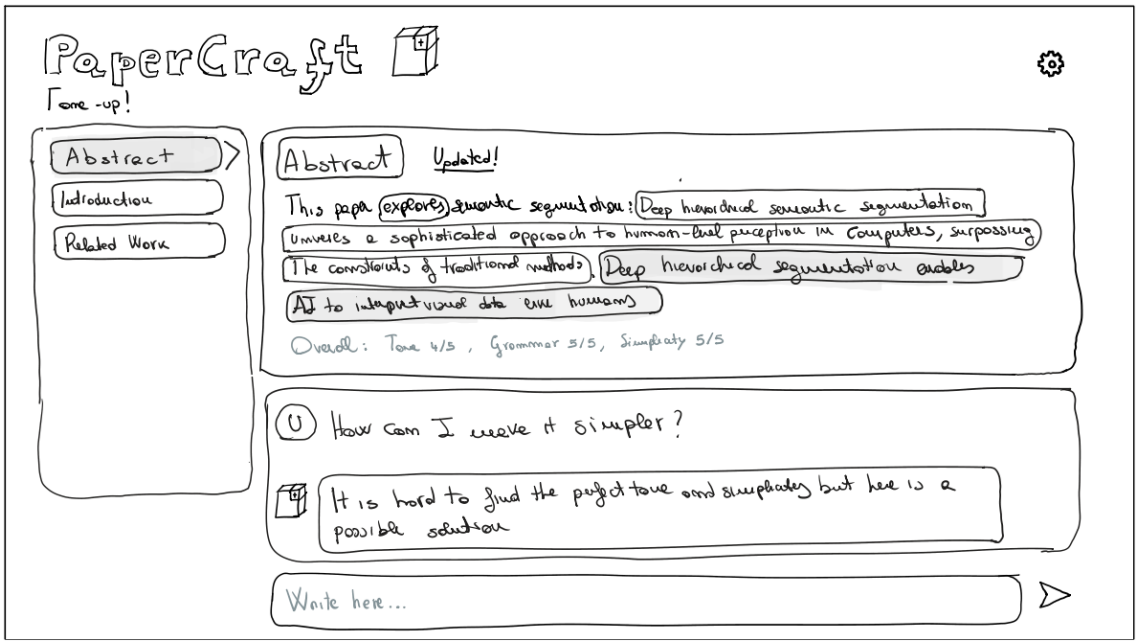


Figure 5.12: After asking for a suggestion, the related segments of text are highlighted in the section area and an *Updated!* note is displayed next to the title of the section. The chatbot answers by highlighting the trade-off between the need for a rigorous tone and the need for a simpler (accessible) style.

Settings interface

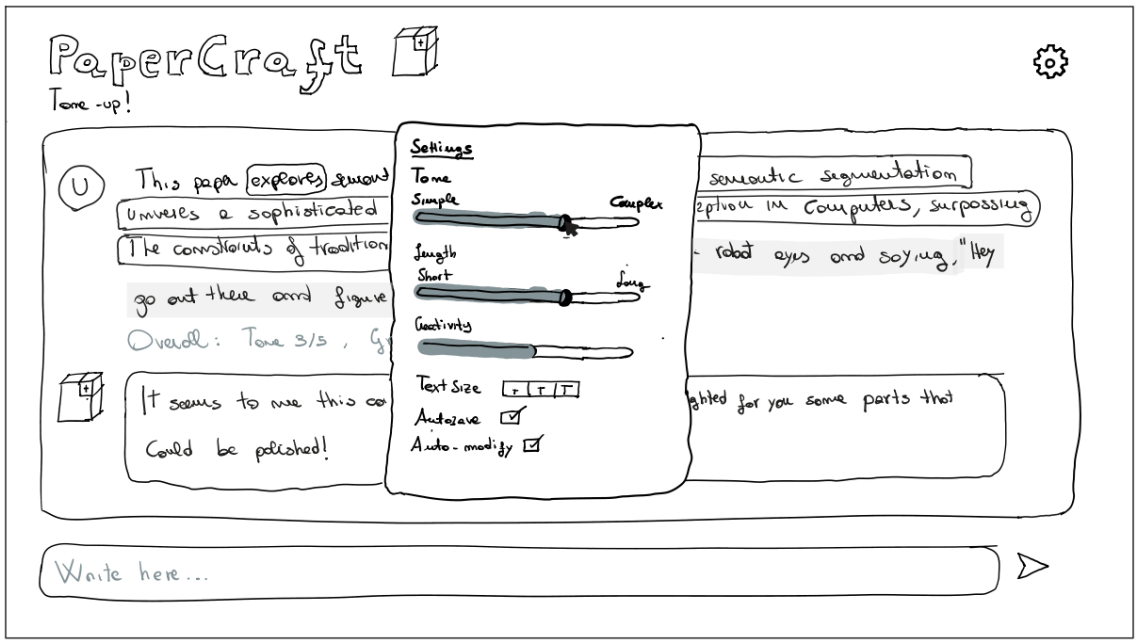


Figure 5.13: Settings view.

Since the settings interface is accessible in both modalities, we wanted to dedicate a specific subsection to it in order to better explain the thought process which led us to implement it, we present it in figure 5.13.

By clicking on the *Gear* icon on the top-right, the user is presented with the possibility to tune the settings of the application. This is possible by using three sliders in order to tune *Tone, Length* and *Creativity*. Moreover, the user is able to eventually modify the text size, enable an *Autosave* function (Related to the document version) and an *Automodify* function.

In particular, we justify the following settings:

- **Style**: The coherence and consistency of the language and writing characteristics within the text. A **High** value pushes the LLM to generate a formal correction characterized by the appropriate style for academia, which must be rigorous. A **Low** value instead helps the user obtain a more accessible writing style. We decided to introduce this setting since the need for a more accessible tone emerged from the interviews.

- **Length**: The maximum allowed length of the AI's generated modified text. This allows for the possible expansion or summarize of the content of a segment.

- **Creativity**: Tendency of the AI's generated text to adhere to predefined patterns or be novel and engaging. A **High** value will free the LLM from the constraint imposed by the context, therefore it would allow the user to obtain more innovative generations which could be beneficial to creativity and novelty. A **Low** value instead would prevent the model from diverging from the context of the chat, allowing more precise generations. We introduced this settings by inspiration from the interviews.

- **Text-size**: Introduced to aid for the accessibility of the application.

- **Autosave**: Appropriate for the document mode.

- **Auto-modify**: By checking on *Auto-modify*, the user will allow the chatbot to correct the message automatically, without waiting for the user to confirm a correction.

5.2.2 "Desktop Application" Prototype

The desktop application has been designed starting from a basic text editor. The user can decide to open an existing text file (using the operating system's file chooser) and edit it, or to start from scratch (in this case he is not asked to choose the file name until he decides to save it, to avoid unnecessary steps). The text editor is capable of keeping a history of modifications, letting the user jump back and forth between the changes, undoing or redoing them (actually the redo button has not been drawn in the paper prototype). The saving functionality has been thought by taking inspiration from existing text editors: if there are changes not saved, a window pops up asking the user what to do with them, save them or discard them. If the file is new and the user wants to save it, the editor asks him what is the name of the file, and shows him the operating system's file chooser to let him decide where it should be saved without having to specify the full path.

The AI functionalities have been designed to be integrated with this standard, well-known workflow, trying to interfere with it as little as possible. If the user wants to analyze a piece of text that he has written more in detail, considering not only the grammar but also the tone and clarity, he can simply highlight some text using the cursor. There is a label on the top of the window that informs the user that he has to perform this action. When finishing highlighting the text, immediately after having released the mouse button, a contextual window pops up showing two options: receive an automatic suggestion related to the piece of text just highlighted ("suggestions..."), or edit automatically the piece of text ("edit automatically").

If the user chooses to get a suggestion, a panel opens up on the right. It contains a suggestion given by the AI that can be regenerated so that the user can change it if he is not satisfied. If, after having regenerated a suggestion, the user wants to see the previous, he can press the button "Previous suggestion". In case of multiple regenerations for the same highlight, the button scrolls backwards through a list. The panel is closeable, so the user can return to focus on the text when he does not need the suggestions anymore.

If the user instead wants to edit the highlighted text automatically, the editor splits in two halves showing two different versions of the text: one with the piece of text written by the user, and another with the piece autocorrected by the AI. The user is prompted to choose among the two versions. Moreover, he can try another correction if he is not satisfied. In that case, the auto-corrected text changes, proposing a new AI version.

Finally, there is the settings screen. It can be opened by pressing a dedicated button on the home screen and shows three sliders. Each of them enables the user to adjust the overall tone, length, and creativity (named temperature in the prototype, it has been renamed in the final prototype to improve understandability) of the auto-corrected text. The tone can range between formal and informal, the length between short and long, and the creativity between rigid and creative.

There are some functionalities that were not implemented.

- The editor should give the user automatic suggestions regarding grammatical errors and typos, without his intervention. In fact, if there is a grammatical error, the user is not able to spot it without highlighting the affected text. This behavior should be avoided.
- The redo button is not shown in the prototype.

- The user should be able to scroll through all the suggestions he has generated for the text he has highlighted. Since there is only the "Previous suggestion" button, he can not. There should have been at least another button "Next suggestion".
- The user should also be able to scroll through all the auto-corrected versions of the text, in the split screen. That probably should have required a dedicated panel.
- The prototype does not cover the memorization of the suggestions given the same highlight. If the user highlights a piece of text for which a suggestion has already been generated in a previous interaction, what should happen is undefined.

5.2.3 Connection with storyboard, project goal and three tasks

Both of our prototypes are able to be used to perform all the three tasks:

- The user can request a suggestion (simple task) for a selected piece of text. However, we must note the difference between the two protoypes: in the chatbot, the badly written section is automatically recognized, while in the Desktop Application the user is free to select a piece of text by drag and drop.
- Correct the piece of text automatically (moderate task)
- Tune the settings in order to change the corrections' style and length (complex task).

Both prototypes are aligned with our project goal, because they help researchers in editing their documents and texts, trying to speed up the process in an automated way, with the help of AI.

The prototypes are also aligned with our storyboard: the user can in fact reduce the time of writing his papers with this desktop application, because of the automated text generation and improvement suggestions.

5.3 Low-fi prototype: Desktop Application

In this section, we show all the developed drawings of our low-fi prototype for the Desktop Application version of PaperCraft.

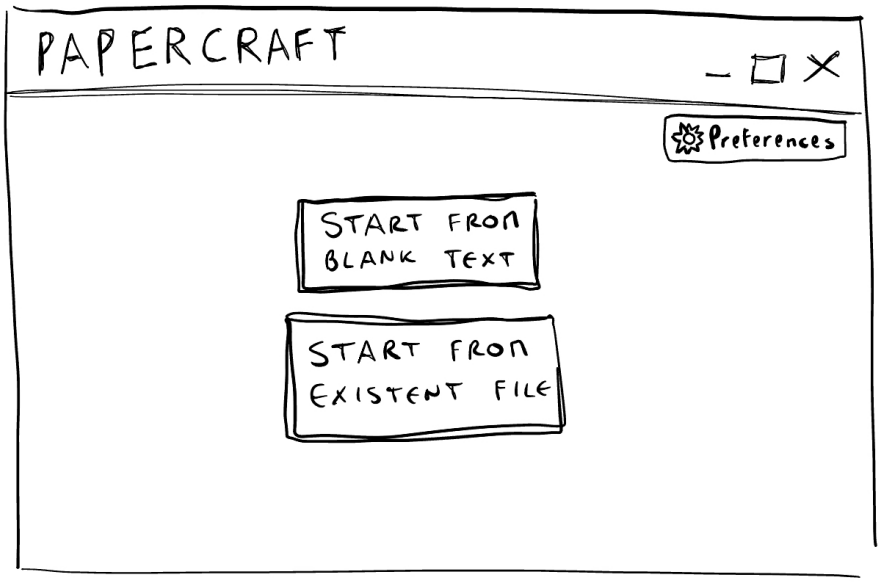


Figure 5.14: The home screen of the app

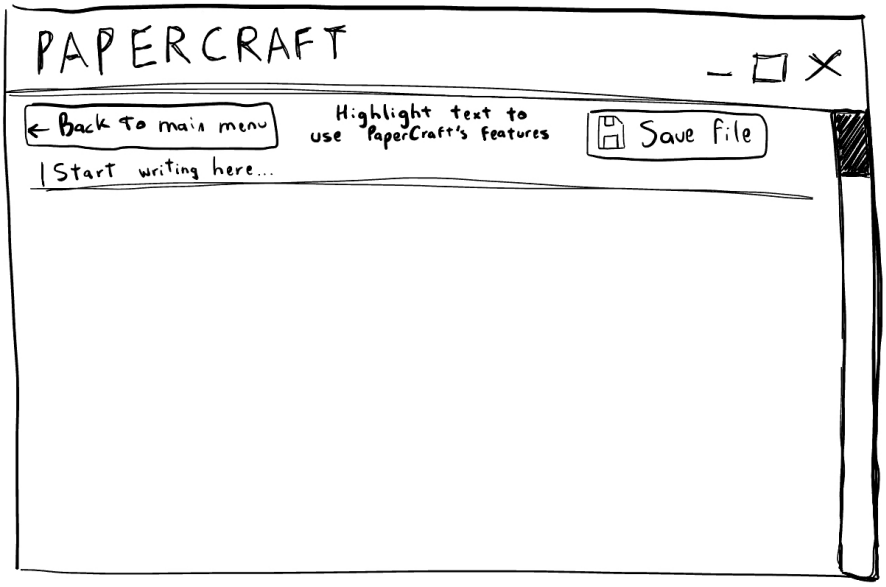


Figure 5.15: Here the user can start writing text, like in a normal text editor. There is a hint for the user explaining what to do to trigger AI features. If the text is empty, a placeholder suggests to start writing some text

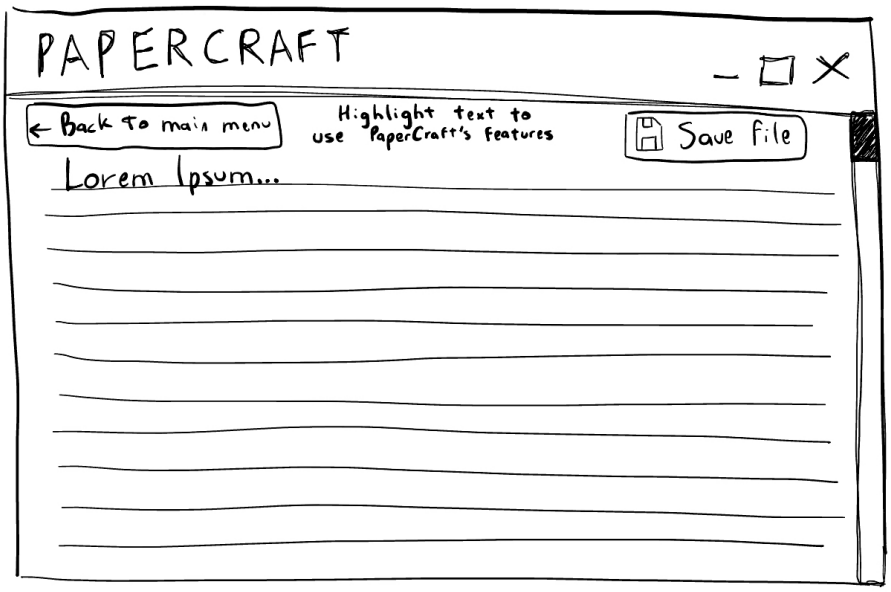


Figure 5.16: This is what happens if the user starts editing an existing text file. The screen is pre-filled with mutable text.

5.4 Heuristic Evaluations

In this chapter, we describe the heuristic evaluations received for our two low-fi prototypes. The received evaluations provided us with valuable insights into the usability and user experience of our prototypes and the violations helped us identify areas where the prototypes lacked usability. Each evaluation helped us understand how users interact with our prototypes and where potential obstacles or confusion may arise.

For both prototypes, a member of an external group has accepted the proposal to conduct a heuristic evaluation, which has been held during the laboratory hours at Politecnico; we decided to split our group into 2 couples as follows:

- *Giulia Di Fede* (Observer/Helper) and *Daniele De Rossi* (System response simulator) managing the evaluation session for the **ChatBot** prototype.
- *Davide Colaiacomo* (Observer/Helper) and *Lorenzo Bertetto* (System response simulator) managing the evaluation session for the **Desktop application** prototype.

During the execution of the evaluation, the observer/helper has taken some important notes about the general flow of the session and has been ready to answer all the possible doubts and guide the evaluator in case of necessity; on the other hand, the system response simulator has silently manipulated the interfaces of the paper prototype according to the evaluator's actions.

Once the sessions were completed, there has been a moment of friendly and formative confrontation with the evaluators, during which the major strengths and weaknesses of the prototype were discussed, concluding with the knowledge that the formally written heuristic evaluation would have brought up all the necessary violations of our work; this was supposed to guide us for the next steps of our project and lead to higher level prototypes.

Before moving on, here we recall Jakob Nielsen's ten heuristics [1] in order to later introduce the received heuristic evaluations.

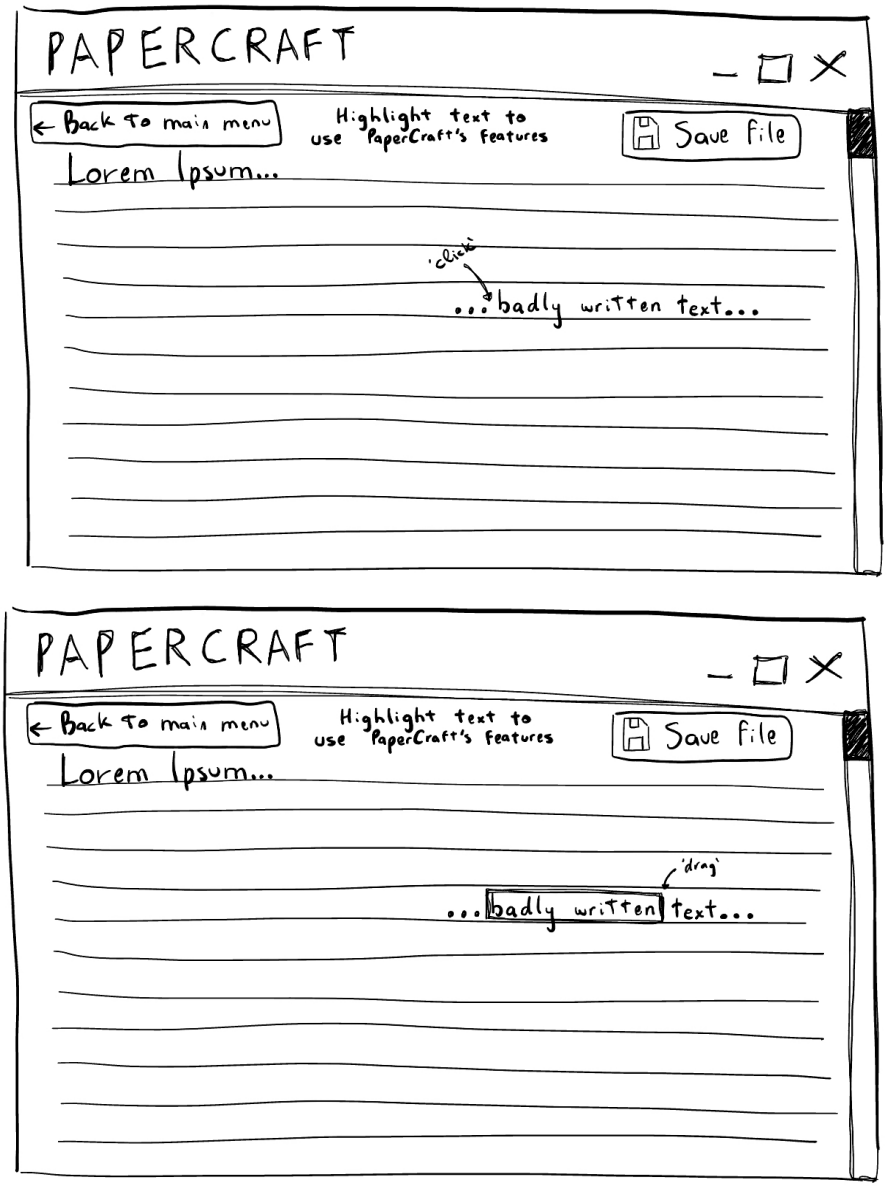


Figure 5.17: Click and drag motion performed by the user in order to highlight some text to ask for a suggestion.

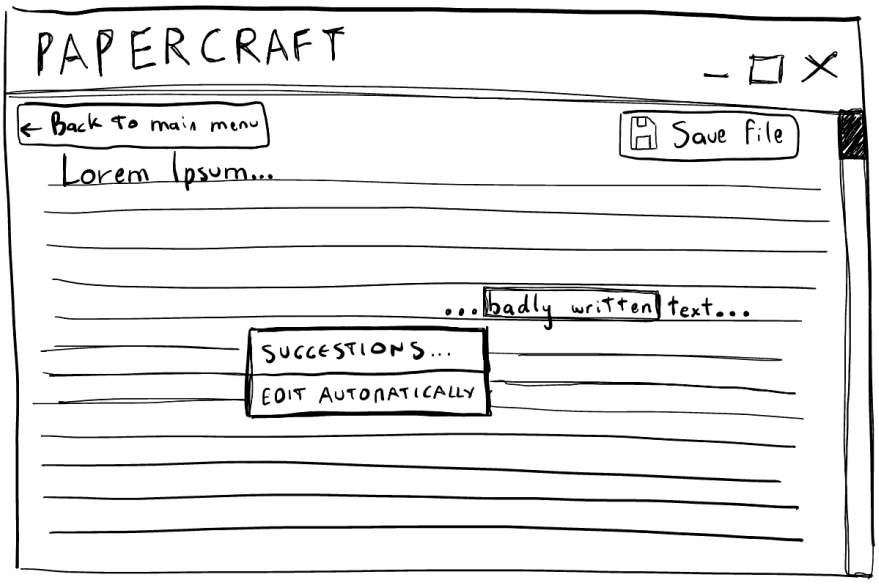


Figure 5.18: As soon as the user releases the mouse button, a window pops up showing him two possible actions: get an automatic suggestion for the highlighted text or edit it automatically.

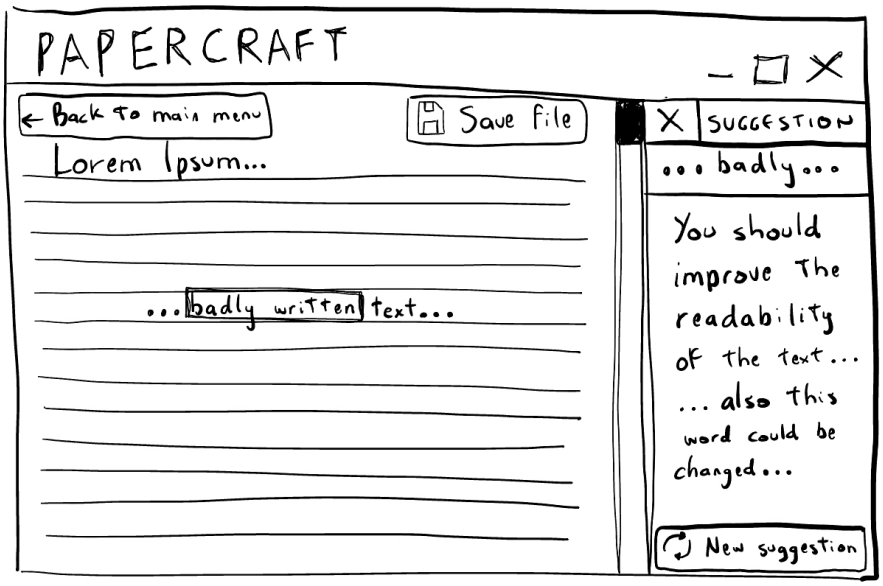


Figure 5.19: When requesting a suggestion, the application opens a right panel that shows the highlighted text and the AI suggestion. There is a button to generate a new suggestion, if the user is not satisfied with the current one. The panel is closeable.

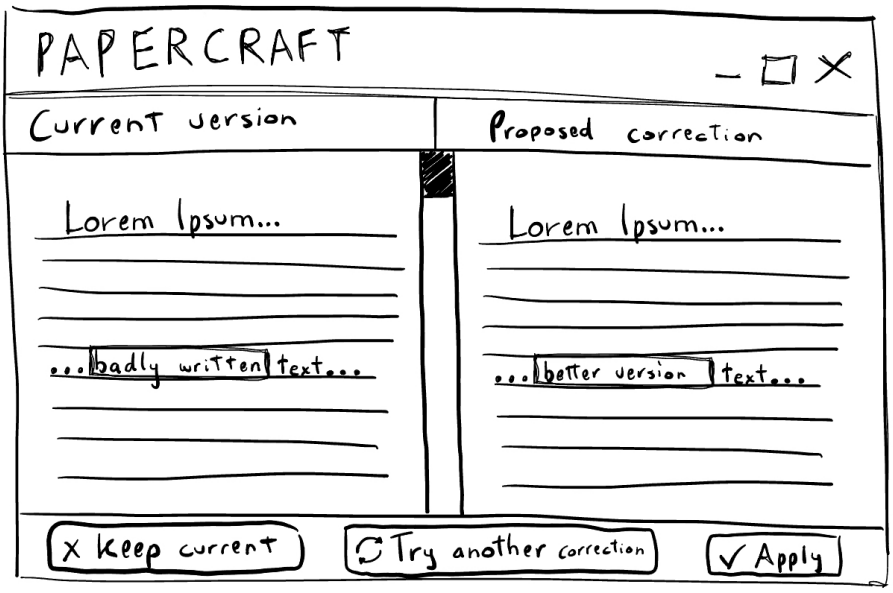


Figure 5.20: When the user asks for an automatic correction, the two versions of the text (corrected and not) are shown, and the user can perform three actions: accept or reject the correction, or generate another correction.

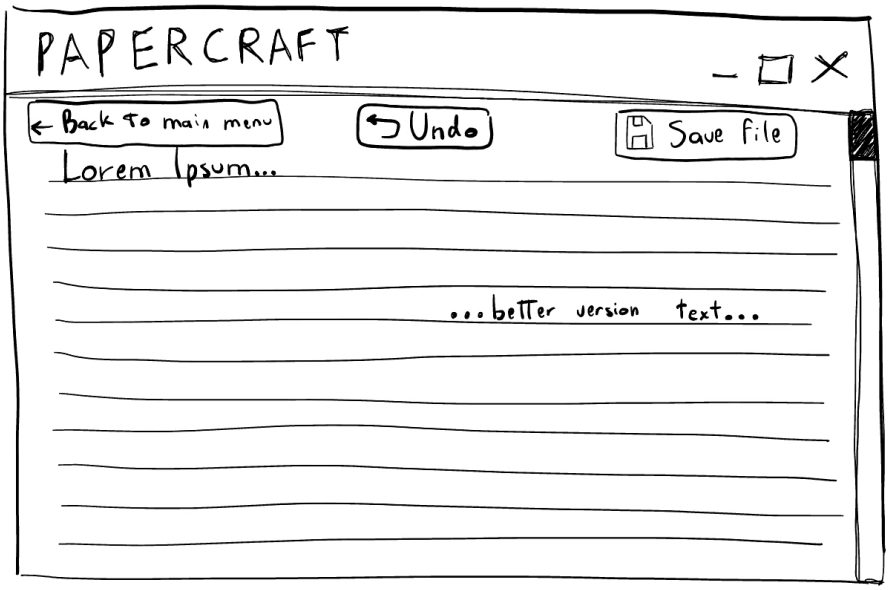


Figure 5.21: When the user has changed the text, an undo button shows.

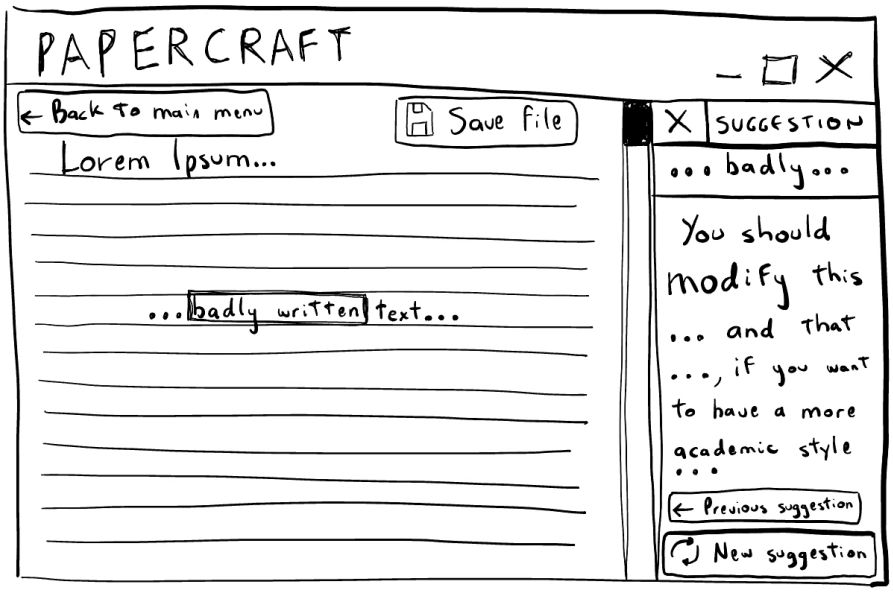


Figure 5.22: If the user generates a new suggestion (presses on "new suggestion"), a new button appears, that lets the user go back to the previous suggestion.

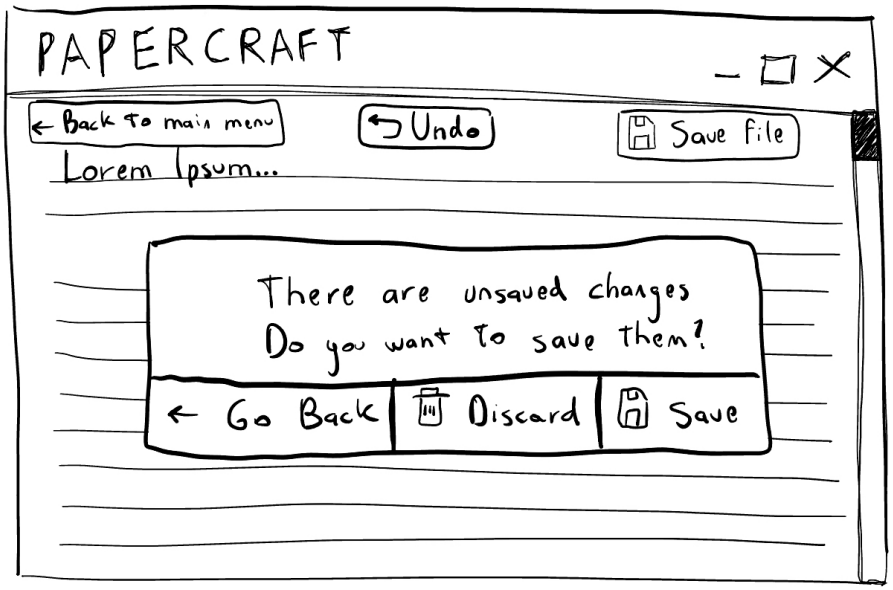


Figure 5.23: If the user wants to exit and there are changes not saved, he is prompted to decide what to do on them.

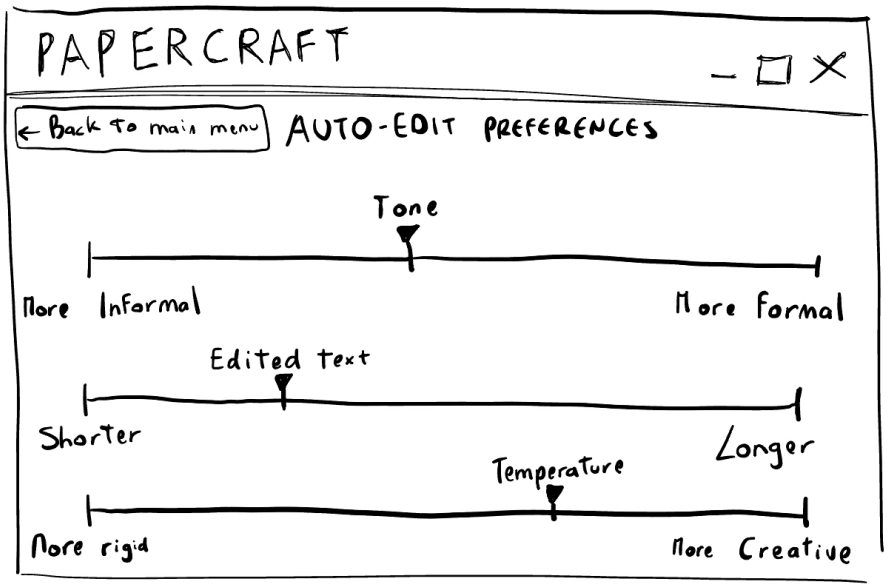


Figure 5.24: The settings screen. Here the user can customize the text that is generated by the AI.

- **(H1) Visibility of System Status**: The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.
- **(H2) Match between system and the real world**: The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
- **(H3) User control and freedom**: Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.
- **(H4) Consistency and standards**: Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.
- **(H5) Error prevention**: Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.
- **(H6) Recognition rather than recall**: Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the interface to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.
- **(H7) Flexibility and efficiency of use**: Accelerators — unseen by the novice user — may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.
- **(H8) Aesthetic and minimalist design**: Interfaces should not contain information which

is irrelevant or rarely needed. Every extra unit of information in an interface competes with the relevant units of information and diminishes their relative visibility.

- **(H9) Help users recognize, diagnose, and recover from errors**: Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.
- **(H10) Help and documentation**: Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.
- **(HN) Non-heuristic issue**

5.5 "ChatBot" Prototype - Heuristics

The heuristics related to the *Chatbot* prototype version of our project offered us a lot of valuable insights; all the followings are the result of 1 heuristic evaluation which was exhaustive enough, so there has not been the necessity to implement a merge operation.

1. **H10**: Help and Documentation.
   - **Where**: Start page.
   - **What**: The chat-bot bar reads "Write here", without giving any hint of what the user may or may not ask.
   - **Why**: The user may be restrained from using the bar at first, as it may experience it as a "black box" without providing any further knowledge of what is going to happen.
   - **Severity**: 3, as this lack of additional information on the spectrum of actions that the user is allowed to perform is recurrent in many pages of the prototype and can create confusion in the user.
2. **H6**: Recognition Rather Than Recall.
   - **Where**: Make Simpler Suggestion – Document, Abstract.
   - **What**: The corrected version of the abstract is shown here, but not the previous version provided by the user in the previous page.
   - **Why**: Since the old version is not displayed, the user is required to remember it in case he wants to "understand" the corrections put forth by the AI by comparing them with his solution.
   - **Severity**: 3, since in several pages a "new version" is proposed by the AI without providing a comparison with the previous version the user, making it hard for him to understand the metrics scales.
3. **H1**: Visibility of System Status.
   - **Where**: Make Simpler Suggestion – Document, metrics below Abstract.
   - **What**: It is not clear whether the metrics "Tone, Grammar, Simplicity" displayed below the AI corrected version refer to the corrected version or to the previous one submitted by the user.

- **Why**: The user does not understand what version the metrics refer to, and this ambiguity could confuse him/her, who could think they are referring to the currently displayed version (AI generated) instead of his/her own.
- **Severity**: 3, as this issue persists in the following tabs.

4. **H10**: Help and Documentation.
   - **Where**: Make Simpler Suggestion – Document, metrics below Abstract.
   - **What**: The metrics "Tone" and "Simplicity" are not self-explanatory and no explanation of their semantics is provided.
   - **Why**: Confusion about the rationale used by the AI to correct the paper arises as the user cannot further his/her knowledge about the metrics employed by the app to improve his paper.
   - **Severity**: 3, because the documentation is not accessible neither here nor in the 'Settings' page, making it impossible to understand the rationale even when the user goes looking for it.
5. **H4**: Consistency and Standards.
   - **Where**: Make Simpler Suggestion – Document page, metrics below Abstract, and Settings page.
   - **What**: In the Settings of the app, "simple" is used as a boundary label for the "Tone" setting, whereas, in the other page mentioned, "Simplicity" is treated as a standalone metric unrelated to the "Tone".
   - **Why**: The usage of two semantically similar words in these two different contexts creates confusion in the user, who does not understand how "Simplicity" is evaluated as a standalone metric if simple is defining the "Tone" metric already.
   - **Severity**: 2.
6. **H5**: Error Prevention.
   - **Where**: Sending Messages – Chat.
   - **What**: While "PaperCraft is thinking", the chat-box is not disabled, enabling the user to write additional information.
   - **Why**: The user can keep typing in the chat-box at the time when user control should be revoked, especially because it is not clear what he/she would be allowed to actually type (as explained in point 1) at such time and may trigger unwanted actions.
   - **Severity**: 4, as the following flow could become unpredictable.
7. **H9**: Help Users Recognize, Diagnose, and Recover from Errors.
   - **Where**: PaperCraft Answers to Grammar – Chat, chatbot answer.
   - **What**: The text answer provided by the chatbot does not include any instructions for the user on how to proceed with the improvement phase, as it only reads: "It seems to me this could be greatly improved. I highlighted for you some parts that could be replaced!".
   - **Why**: The missing flow instructions (e.g. "hover on text to make options pop up") create confusion in the user, who could start tapping around in hope to trigger a change.

- **Severity**: 3.

8. **H9**: Help Users Recognize, Diagnose, and Recover from Errors.
   - **Where**: Hover Grammar – Chat, grammar pop-up.
   - **What**: Button with corrected version suggested by the AI (i.e. "Explores") does not carry information on how to apply the change.
   - **Why**: The user could be confused as to how to apply the suggested correction, which would not happen if a "click to apply change" were displayed or the option was better highlighted.
   - **Severity**: 1.
9. **H1**: Visibility of System Status.
   - **Where**: Hover Undo – Chat, Grammar metric.
   - **What**: The Grammar metric '4/5' has been changed to '5/5' after the correction, but the change is not highlighted in any way (e.g. 5/5).
   - **Why**: The missing highlight on the changed score creates confusion in the user, who cannot appreciate what the effect of his action was when the tab refreshes and the result is displayed.
   - **Severity**: 2, as the issue is repeated in the following tabs.
10. **H1**: Visibility of System Status.
    - **Where**: Overtone – Chat page, Tone pop-up.
    - **What**: "Tone up!" per se does not carry any info on what kind of tone is going to be applied.
    - **Why**: The user cannot predict what tone is going to be applied, especially if he/she did not set the tone preferences himself/herself but is using the default ones; thus, just displaying "Tone up!" with no additional info with regards to what tone the app is currently using could confuse the user as to what change he/she is to expect next. Plus it is not clearly shown where one should go in order to manually adapt such tone preferences.
    - **Severity**: 3.
11. **H1**: Visibility of System Status.
    - **Where**: Tone Corrected – Chat page, metric bar.
    - **What**: The "Tone" metric '1/5' has been changed to '3/5' after correction, but the change from 1 to 3 is not easy to spot as it is not clearly highlighted.
    - **Why**: The missing highlight on the changed score may require a longer stare on the side of the user who would not immediately appreciate what the effect of his action was.
    - **Severity**: 2.
12. **H4**: Consistency and Standards.
    - **Where**: Settings.

- **What**: The box contains two different kinds of settings; in fact "Tone, Length, Creativity and Text size" refer to the AI generator (Chatbot flow & Document flow), while "Auto-save and Auto-modify" only refer to the Document flow.
- **Why**: The design is inconsistent with the underlying view the user is in, as it puts together two different kinds of settings which should instead be made contextual to the flow the user has followed so far. As it is, the same interface presents semantically uncorrelated options; for instance, if the settings are opened while in the chat-bot flow, the document settings are unnecessary since they are not correlated to the underlying view.
- **Severity**: 3, as the user could miss the meaning of part of this menu and consequently not be able to exploit it as intended.

13. **H10**: Help and Documentation.
    - **Where**: Settings page, metrics' bars.
    - **What**: The "Tone, Length and Creativity" bars' extremes are unclear, as the indications "Short/Long" and "Simple/Complex" are not contextualized with any default example to better explain the intended scoring scale.
    - **Why**: The user does not know how to tune the parameter bars, as their respective scale boundaries do not provide clear indications as "Short i.e. ¡ 50 chars" would.
    - **Severity**: 2, as the lack of default examples could make the user actions more random than intended.
14. **H10**: Help and Documentation.
    - **Where**: Settings page.
    - **What**: The "auto-save and auto-modify" buttons do not carry any hint on their functionality.
    - **Why**: As the buttons are not self-explanatory nor contextualized (as said in H12), these two buttons result in being unclear.
    - **Severity**: 3, since these options could never be used by the user who fails to understand them.

5.6 "Desktop Application" prototype - Heuristics

Now the heuristics related to the *Web Application* prototype version of our project are going to be detailed; similarly to the previous prototype, all the followings are the result of 1 heuristic evaluation, so there has not been the necessity to implement a merge operation.

1. **H7**: Flexibility and efficiency of use.
    - **Where**: "Suggestions" view.
    - **What**: Lack of a "suggestions history".
    - **Why**: The user might want to move forward in the suggestions history after going back, but he can only generate new ones.
    - **Severity**: 3

2. **H3**: User control and freedom.
    - **Where**: "Improved text" view.
    - **What**: Lack of a "suggestions history".
    - **Why**: The user might want to go back to a previous suggested version, but he can only generate new ones.
    - **Severity**: 3
3. **H4**: Consistency and standards.
    - **Where**: "Improved text" view after applying a correction.
    - **What**: Presence of an UNDO button.
    - **Why**: The UNDO button is present, in this case, to roll back to before the application of the change, but this possibility is absent if the user wants to roll back after applying a suggestion.
    - **Severity**: 2
4. **H10**: Help and documentation.
    - **Where**: "Preferences" view.
    - **What**: Lack of a graduated scale for the preferences.
    - **Why**: The user, lacking a numeric guide or a graduated scale to select the settings, could be confused in choosing a preference instead of another.
    - **Severity**: 2
5. **H8**: Aesthetic and minimalist design.
    - **Where**: "Preferences" view.
    - **What**: Lack of a subdivision of the sections.
    - **Why**: The positioning of the elements on the screen appears to be too fluid and not compact, without shape.
    - **Severity**: 3
6. **H10**: Help and documentation.
    - **Where**: "Preferences" view.
    - **What**: Lack of an info point.
    - **Why**: The user might want to get suggestions and clarifications about how the AI modifies his work with respect to the selected preferences, but there is no description about how these work.
    - **Severity**: 2

The subsequent violations, conversely, have been unanimously considered to be rejected, following a comprehensive discussion among ourselves, as we perceived them to lack correctness or substantive value.

1. **H5**: Error prevention.

   Reason for rejection: We consider that not including the 'save' functionality for empty files is not a violation, since it is never conventionally adopted to request so when a file is empty or has not been modified since the last save.

   - **Where**: View for "new text".
   - **What**: Lack of a notification if the file is empty.
   - **Why**: The user might want to save the empty file, but in this case there is no notification.
   - **Severity**: 2

2. **H9**: Help users recognize, diagnose, and recover from errors.

   Reason for rejection: We decided not to include this violation in the accepted ones because, apart from the Nielsen heuristic association, it presents itself as a duplication of the previous rejected violation.

   - **Where**: View for "new text".
   - **What**: Lack of a notification if the file is empty.
   - **Why**: The user might want to save the empty file, but in this case there is no notification.
   - **Severity**: 2

3. **H4**: Consistency and standards.

   Reason for rejection: The violation reported is not correct with respect to the prototype presented, as the indicated view actually presents, side to side, both the original version of the text and the suggested one by the AI.

   - **Where**: View with suggested corrections.
   - **What**: Impossibility to contemporarily visualize the original text and the suggested text.
   - **Why**:The user, in order to avoid having to remember the text suggested, might want to see the original text and the proposed modified text at the same time.
   - **Severity**: 2

4. **H3**: User control and freedom.

   Reason for rejection: We decided not to include this violation in the accepted ones because, despite being correctly noted, it is presented as a duplication of the accepted violation #2.

   - **Where**: View with improved text.
   - **What**: Lack of the possibility to roll back to a previous suggestion.
   - **Why**: The user might want to look into past suggestions but is not able to; he/she can only generate new ones.
   - **Severity**: 3

5.7 Selected prototype

When choosing which paper prototype to move forward with, we carefully evaluated the results of the heuristic evaluations done on the proposed designs. We aimed to find the prototype that had the most potential for making meaningful improvements as we progressed to the medium-fidelity stage.

It's important to mention that the heuristic evaluation of the "Desktop Application" prototype did not meet our expectations. We noticed some duplicated or imprecise heuristic violations. As seen in the list of heuristic violations for the "Desktop Application" prototype provided earlier, these aspects made it hard for us to get useful insights for improving the prototype. Additionally, we think the evaluation lacked clarity in its findings, which led to confusion and complicated the refinement of our prototype. Despite the limitations caused by the issues mentioned above, we carefully reviewed the received evaluation to find potential areas of improvement.

On the other end, the heuristic evaluation of the "Chatbot" prototype was particularly useful and well conducted, providing us with valuable insights and pointing out key areas where improvements could be made. This evaluation helped us realize the usability issues with this prototype, as it included explanations for the severity of the problems, giving us further details on how serious the violations were.

The main reason we chose the **Chatbot** prototype is because we thought it had a more detailed design and its features could offer users a richer and more varied experience. Plus, we could take greater advantage of the evaluation done on this prototype, as mentioned earlier.

Medium-fidelity prototype

At this point, we had the necessity to improve the low-fidelity **Chatbot** prototype we selected.

To start working on the medium-fidelity prototype, we had another brainstorming session to look at the helpful reported violations we got from our evaluator we used as a guide. After the analysis, it was clear that the two most significant screens which had to be refined were the screens dedicated to the *Chat* mode and the *Document* mode. This was mainly because the user would mostly focus his/her interactions around these two screens of the application. However, since most of our application is centered around these two screens, the medium-fidelity prototype covered most of the lo-fi prototype itself.

We decided to employ the versatile design tool *Figma* which offers a professional environment for UI design.

As we worked, we made sure to correct all the mistakes pointed out in the heuristic evaluation by using the following strategies applied in the Medium-fidelity prototype:

- *Violation 1*: this was resolved by introducing a tutorial screen when the application is opened for the first time, while a change in the placeholder text for the AI input box has been also applied to supply the user with a better indication of what the AI is going to expect.
- *Violation 2*: the violation has been resolved by keeping a graphical sign on the sentences that have been changed, so that users can over hover again on them and see what was the original version, allowing also to undo the correction automatically.
- *Violation 3*: the issue has been corrected by clearly specifying that the metrics refer to the currently visualized version in all necessary interfaces.
- *Violation 4*: to provide users with clarifications about the metrics, a specific info point has been inserted next to them, so that, when they hover on it, an overlay box with further explanations is displayed.
- *Violation 6*: to correct this problem, the chat's aesthetics have been modified in all necessary interfaces in order to clarify that the insertion box is disabled while crucial computational operations are being carried out.
- *Violation 7*: to solve this violation, the AI response has been updated so as to give users the proper hint about what they are supposed to do to receive the support they seek.
- *Violation 10*: in this specific case, a description of how the AI is going to improve the highlighted text is provided in the overlay box, as well as a reminder for users that they can change the configuration of the tone adopted by the AI in the 'Settings' section.
- *Violation 12*: not dealt with, as it regards a page not included in the mid-fi prototype. Refer to the Hi-Fi prototype.
- *Violation 14*: not dealt with, as it regards a page not included in the mid-fi prototype. Refer to the Hi-Fi prototype.

- *Violations 5, 8, 9, 11, 13*: these violations were indicated to be the least important, with a severity value of 1 or 2, so they were the last to be considered, so to give more priority to the most crucial ones. It was sufficient to introduce some graphical enhancements and written indications with respect to the lo-fi prototype to get the adjustment we were suggested (except for violation #13, which regards a page not included in the mid-fi prototype).

High-fidelity prototype

7.1 Tools

Since we have chosen to build the ChatBot as a Web Application, we created it using the React framework for JavaScript. The decision to use React was driven by the fact that we all have already learned to use it in the previous Web Applications course. For the styling of the app, we used Bootstrap, the popular CSS framework, for the same reason. The application was developed using Node.js and the npm package manager.

The application had to have data able to persist if the user refreshed the page. In order to make this happen, we built a back-end server in charge of keeping the user's data. For that purpose, we used Express.js, the popular framework for NodeJs.

We didn't use the real AI for our app, because there were not any existing services that could fit our peculiar needs.

We also used the *Ngrok* service to let the users test our application while we controlled the back-end server on a different PC (more on this later).

The full code of the prototype is available at this link.

7.2 Significant Screens

7.2.1 Chat

This is the main screen of the application. When the user visits the website he can see an empty chat. He can submit a message using a textbox, where he can request some suggestions directly to the AI or write some text that has to be improved. This screen is important for us because it is the main way for the user to interact with the AI, either by chatting with it or by writing some text to be corrected.

7.2.2 Document

The document screen opens when a user wants to improve a paper he has already written or a document of some kind. From this screen the user can edit a section of the document, either manually or automatically using the AI. The application also provides an overall suggestion about the section the user is examining. We think that this is an important feature of our application. For a researcher, editing directly his papers with the help of AI could be an advantage.

7.2.3 Popover

The popover is a tiny piece of UI that pops up when the user clicks on a section of the text that has to be corrected. Despite being small, it is very powerful and it offers a lot of important features. It opens near the piece of text that has to be corrected, so it is always clear to the user which section of the text is going to be modified. It shows a suggestion of the AI regarding that particular piece of text. It enables the user to generate corrections, using the AI, to substitute the bad piece of

text. For each correction, the user can tune some parameters that influence the characteristics of a certain correction.

7.2.4 Documentation

This screen is a simple text that the user can read and contains a description of the application. It explains the meaning of the settings the user can change, the sections of the sidebar, and the statistics that the application provides to the text analyzed.

7.3 Limitations

7.3.1 Pre-stored data

The application starts always with some pre-filled data, that represents the previous interactions of the user with the application. The presence of this data in the application is not strictly necessary, since the user can in any case start a new chat or upload a new document. The data is needed to simulate a real usage of the application, where all the interactions (previous chats, documents) get saved in a database.

7.3.2 Hard coded features

Since the application does not use the AI functionalities, the messages that the user can send, the pieces of text that he wants to be evaluated, and the documents that he can submit are limited and pre-determined. However, for some features of the application, we managed not to depend on hard-coded data.

We used the technique of the Wizard-of-Oz to generate some AI corrections depending on the users' actions, because otherwise it would have been impossible to cover all the possible combinations of settings, and texts to be corrected. We think it is worth mentioning the prompt engineering work we focused on to enable chatGPT to generate the corrections. More specifically we adopted the technique mentioned in [3], in order to train chatGPT through a few-shots prompt. We tried to limit as little as possible the users' actions. We didn't need to implement an interface for the Wizard as we used the debugger of *Visual Studio Code* which easily let us change the response of the server. The Wizard only responds to the user when they ask for a new correction, according to the settings chosen by the user, this way we covered all the combinations of the settings.

The reason why we couldn't rely on some existing services is that the features that our application provides are peculiar: the AI, in our vision, is able to distinguish between a simple conversation and the submission of text that has to be improved. Moreover, the text to be improved, after being properly recognized, has to be split in correct and not correct sub-pieces. The AI should provide our application some properly formatted data, that tells about which parts of the text should be highlighted or not and for each highlighted part it should generate a suggestion. Finally, each "bad" piece of text should be categorized in one of the three types of error: Grammar, Clarity, or Tone. Each of these errors should have an indicator that shows the severity of the error.

7.3.3 Other limitations

The prototype does not have a users database. This means that the application cannot be used by multiple users at the same time with different data. Also security features, like the login system,

are not implemented. The documents cannot be deleted from the database once uploaded. This feature is not important for the completion of our tasks so we did not implemented it.

Usability testing

8.1 Preparation and run

The ultimate step of our project consisted of conducting a set of **usability tests**; the goal was to show off a stable version of our hi-fi prototype to some final users, so to get a deep and concrete feedback on how well we had managed to convey the functionalities we have worked on until this final stage of the project.

In order to achieve so, we managed to conduct **4** usability tests by recruiting the correspondent amount of scientific researchers who showed interest in our application; it is imperative to underline that the researchers who enlisted for the sessions were not related to the ones we had had the opportunity to interview in the *Needfinding* section of the project, as it was important to analyze points of view that were completely detached from our work.

8.1.1 Methodology

The methodology we chose for our usability tests has been the **Think Aloud** methodology; we kindly asked the participant to describe what they were thinking while performing the tasks, therefore stating what they were doing, why and what they thought was happening.

This allowed us to obtain useful insights to improve our prototype as it is shown in the *Results* section; however, we know about the possible drawbacks of this methodology, such as the fact that this could bring up subjective opinions. Also because of our methodology choice, we decided not to take into consideration the **Time-On-Task** metric, as it would have surely been flawed and not realistic.

In order to simulate the application's behaviour, we decided to lean on the Wizard-of-Oz technique, this way we were able to simulate the behaviour of the LLM according to the preferences of the user.

Since the AI features are only partially covered by the **Wizard-of-Oz** technique (other features are hard coded), before each test we informed the tester that, whenever he had to fill in some kind of text, he had to pick the text from a list of paragraphs provided by us. Moreover, whenever the user wanted to upload a document, he had to choose a document provided by us.

8.1.2 Equipment

The equipment we used consisted of two laptops, one of them used as the client on which the application was running; the server was running on the other laptop and a tunnel with *Ngrok* was employed in order to establish a connection between the client and the server to enable us to exploit the Wizard-of-Oz technique.

8.1.3 Material

As a preliminary condition for this assignment, we have prepared all the material we thought to be necessary to conduct the tests in the most controlled and profitable way; the documentation has been organized as follows and can be consulted by means of the following link:

- *PaperCraft Usability Test - Script*: a document explicitly describing the flow of the test session from the point of view of the facilitators; it has been realized as a roughly word-per-word presentation of what the facilitators are supposed to say, along with some interludes of how they were supposed to manage all the other pieces of documentation during the session. The usefulness of this script lies in the necessity to interact with all the evaluators in the same way, avoiding the eventuality to give some of them different information from others or present the overall structure of the test with different approaches, which could lead to diverse biases in the general experience.
- *PaperCraft Usability Test - Consent form*: a document representing an informed consent to be presented to the evaluators in the early moments of the test sessions.
- *PaperCraft Usability Test - Pre-test questions*: a document reporting a list of questions to be posed to the evaluators prior to the beginning of the actual test of the application; the purpose was to get a general idea of the background and of the opinions of the testers, in order to have a clearer mind about what could have been expected and how to effectively interpret their actions and behaviors in eventual afterwards reflections.
- *PaperCraft Usability Test - Tasks and scenarios*: a document, particularly useful for the participants, presenting each task that a user should be able to achieve while using the application, along with a brief scenario whose purpose was to help them immerse in the hypothetical situation of a researcher using the actual application.
- *PaperCraft Usability Test - SUS survey*: a document presenting the System Usability Scale [2] questionnaire, to be compiled by the participant after the entire test; this is composed of predefined standard questions that gave us a quick but still reliable measurement of the usability of the system according to the tester's perception.
- *PaperCraft Usability Test - Post-test questions*: a document reporting a list of questions to be posed to the evaluators after the end of the test; the purpose was to dive deeper into the users' experience and have, with respect to the SUS survey, a more informal and detailed but still structured feedback of how the tester had perceived the overall prototype and what were its main strengths and weaknesses in a final user's eyes.
- *PaperCraft Usability Test - Quantitative metrics*: a document summarizing the quantitative statistics gathered during the usability test; this helped us get numerical insights into various aspects of the application's performance and user interactions. By quantifying metrics such as task completion rates and error rates, it offered a structured and objective assessment of the prototype's usability. These quantitative metrics complemented the qualitative feedback gathered through pre-test and post-test questions, providing a comprehensive understanding of the application's strengths and areas for improvement from both a quantitative and qualitative perspective.

8.1.4 Tasks

In order to test our prototype, we decided to further refine our tasks; this way we would have been able to investigate more key features of our hi-fi prototype:

- **Simple**
  - *Request a suggestion to receive info about quality*: Considered successful after the user asks the chatbot for a suggestion by sending a message in the textarea.

- *Upload a locally stored document to refactor a pre-written paper*: Considered successful after the user uploads a document and visualizes the sections.
- *Restore the original text to start over in the refactoring phase*: Considered successful after the user restores the original version of a segment they asked and applied a suggestion for.

- **Moderate**
  - *Edit the section of a document manually to change what will be analyzed by the application*: Considered successful after the user manually applies and confirms some preferred modifications in the section of a document.
  - *Correct grammatical errors in a document to improve its quality (Moderate)*: Considered successful after the user successfully applies any correction suggested by the AI on the popover.

- **Complex**
  - *Improve the formality of a text to adapt it to the specific context*: Considered successful if the user successfully selects the right settings in order to obtain a text suitable to academic standards.

8.1.5 Tests

All the teammates were present and we all took turns in filling the roles of facilitators and observers for the tested participants. The testing was conducted in the available classrooms of the **Politecnico University** (Tester 1, 2 and 3) and in a more aseptic hall of the **DAUIN** building (Tester 4).

Figure 8.1: In order from left to right, Tester 1, 2, 3 and 4

Tester 1

The first test was conducted with *Lorenzo Bertetto* as the facilitator, *Daniele De Rossi* as the observer and *Giulia Di Fede* as the Wizard of Oz.

The research field of the first tester is **Reliability of Embedded Systems**. The time they spend on writing during this period spans to several hours each day. However, they consider themselves as a poor writer and are open to using technological tools for assistance. They highly value editing suggestions and think it is important to adjust the tone in their articles. They have used a chatbot before and wish for suggestions on additional articles for reference, coherence within their article, and control over their writing process.

Tester 2

The second test was conducted with *Davide Colaiacomo* as the facilitator, *Giulia Di Fede* as the Wizard of Oz and *Lorenzo Bertetto* as the observer.

The tester's research interests are about **Natural Language Processing** and **multi-modal learning**. The tester typically writes around 6 articles per year, depending on their workload. They have used chatbots before but have always adapted and refined the output; for this reason, they find chatbots powered by LLMs very useful for grammar checking, paraphrasing and summarizing. We consider this tester to have been extremely valuable to our usability testing as their experience on NLP has given us important insights on the usefulness and the possibilities of expanding our prototype.

Tester 3

The third test was conducted with *Giulia Di Fede* as the facilitator, *Daniele De Rossi* as the Wizard of Oz and *Davide Colaiacomo* as the observer.

The user's research interests focus on **data science**. They have used chatbots like chatGPT before in the process of writing their article. The user seemed skeptical about using a chatbot (low confidence) but thinks it could be useful in creativity or providing alternative perspectives.

Tester 4

The fourth test was conducted with *Giulia Di Fede* as the facilitator and *Daniele De Rossi* as the Wizard of Oz and the observer.

The user's research interests focus on **sensor fusion** and **machine learning**. They write approximately one article every four months and have recently begun using chatbots for paraphrasing and to avoid self-plagiarism. They are comfortable with using AI but stress the importance of checking its results. This user highlights the need to obtain also a more accessible paraphrasing of text, something we have discovered earlier in needfinding, because their audience includes people who aren't experts in their field.

8.2 Results

Task	Successful Task Completion
T1	100%
T2	100%
T3	100%
T4	100%
T5	100%
T6	100%

Table 8.1: Here it shows the successful completion for the designed tasks. Every tester has accomplished all tasks successfully, earning a perfect score based on our testing criteria. However, it's crucial to recognize that despite the excellent score, it doesn't necessarily indicate flawless usability for our prototype.

Tester	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Score
1	4	3	3	4	4	4	4	3	4	4	92.5
2	4	3	3	4	3	2	4	3	4	3	85
3	2	2	2	3	3	4	2	3	3	4	70
4	3	2	2	3	3	1	3	3	2	2	60
Average	3.25	2.5	2.5	3.5	3.25	2.75	3.25	3	3.25	3.25	76.87

Table 8.2: Here it shows the relative values obtained with System Usability Scale (SUS) Questionnaire to measure the perceived usability of our system. We can notice that most of our tests obtained a SUS score over average (68) with the exception of the fourth tester.

8.2.1 List of potential changes

During the four tests, different issues emerged; some raised because the testers had had problems while using our application, while other issues were reported directly by the testers after doing the tests. Some of these were found by us while observing the tester using the application. A subset of the emerged faults were fixed almost immediately because they required few simple adjustments; others were not recoverable, because they were caused by the AI behavior, which was not directly controlled by us (during the tests, the Wizard of Oz generated the corrections using **chatGPT**, so that was not directly connected to our application). When the testers reported actively the issues, they often proposed also some solutions; these solutions were frequently picked to solve the problems, as we thought of them to be extremely valuable, while other times the advices were not directly adopted because we didn't consider them as suitable, but still conceptually useful.

There are also some non trivial changes that we would like to implement into our application; as anticipated before, some were suggested by the users while others came up through some brainstorming between us. First, we are aware that a limitation of our application is in the manual editing of the document. Users must first edit the document section, then they have to submit the modifications, and only after that they can receive a feedback from the AI. What we would like to hypothetically see is a more fluid behavior, by which the user receives suggestions while he is writing; this is actually a behavior similar to code editors or existing document editors, that highlight grammar errors by the time they are committed.

Another enhancement, in this case requested by some testers, is the possibility to summarize a piece of text or even the entire document. When writing research papers, often the researcher has to write the abstract and this feature could be helpful for doing that task. Another case is when the paper has to be no longer than a certain number of pages in order to be submitted to a certain conference. This feature, at the moment, is only provided by the interaction with the AI. A further improvement could be a direct integration in the interface of the application. For example, the user could drag the mouse cursor to select the piece of text he wants to summarize in the document, and a context menu could appear providing a possible summary of it.

Another feature that has been requested is the capability of generating content. At the moment, this feature can be handled only by the interaction with the AI. A more useful way of doing that could be the generation of text based on the context (for example the currently opened document), or better, rephrasing an existing piece of text chosen by the user. This feature requires further investigations for the actual implementation.

Pain-point	Source	Status	Notes
Grammatical errors should not require pressing the Generate new correction button.	Suggested by tester post-test	Solved	N/A
Statistics shouldn't be represented by just numbers but should be more clear	Found by observer.	Solved by substituting numbers with labels.	N/A
List all corrections instead of using arrows to change versions.	Suggested by tester post-test	Not solved because it would require putting the list of corrections in a separate panel, and thus the correlation between correction and highlighted text would be lost.	N/A
Sections should be visually linked to the selected document.	Found by the observer.	Solved by arranging the sections as sub-menus of the documents (hierarchically)	N/A
The apply correction button should not disappear when the applied correction is equal to the correction selected.	Found by the observer.	Solved by setting the button to "disabled" instead of not visible, and adding a tool-tip that suggests the reason of the disabled state.	N/A
The order of the buttons should be set accordingly to the flow of actions: tweak the settings, generate correction, apply correction, restore correction.	Suggested by the tester post-test	Partially solved: since the flow of the application isn't exactly Generate correction - Apply correction - restore correction, we couldn't find a way to address this issue. The user can perform these actions in multiple orders, these actions (except for the settings, that have to be adjusted before generating a new correction) are conceptually at the same level. One thing that we did was to hide the settings view while the user is switching from one correction to the other.	N/A
The tutorial window is not visible enough, as it was often ignored.	Tester's difficulty during testing.	Solved by moving it from inside the chat to above the entire application.	When the test was performed, the DPI was low (it is possible that the text, being smaller, was less visible than usual).
The underlying behavior of the AI regarding the statistics should be explained more in detail	Suggested by a tester post-test	Not solved: the way the score is obtained is part of the AI architecture, which is, in our scenario, hard-coded. Of course, if we were able to integrate the real AI into our application we would have improved our documentation explaining also how the score is obtained.	The tester who raised this question is an expert of LLMs and Machine Learning, so probably what he was asking for is a more technical explanation about how the score is calculated by the internal algorithm.

Table 8.3: List of issues generated during the test activity

Conclusions

9.1 What we learnt

Regarding our experience in this course, we can affirm that throughout the semester we gained significant insights into the process of building a user-centered application. We learned quite some important things about the challenges of this field, spanning from need-finding to testing for our application. This course gave us the chance to not only become aware of some important Human-Computer Interaction theories, but also to become more agile in the field thanks to the development of this prototype.

We are particularly grateful to have learnt about:

- The extreme importance of need-finding in the whole process of building a seamlessly interactive application.
- The need for tools which are particularly defined to aid the human-AI collaboration. In fact, especially relating to our lab theme, it seemed to us that the specific Human-AI interaction branch of HCI is still very new and leaves many open possibilities for exploration. We noticed the need for explainability, control and freedom over Large Language Models.
- The dynamics of usability testing in which our project culminated into.
- The use of professional design tools like *Figma*.

9.2 Group feedback

During the semester we took advantage of each laboratory session, trying to gather as much feedback as possible and integrating all our ideas. Each one of us tried to contribute at their best.

We did our best to hold meetings in order to organize our work and confront our opinions to come up with a common idea and agree on a feasible solution.

We tried to split our work to meet all of our needs, so that working in a group would create a positive experience.

In the end, we believe that all our skills together culminated into a rewarding experience.

Appendix

Consent for Participation in Interview Research

I volunteer to participate in a research project conducted by Lorenzo Bertetto, Davide Colaiacomo, Daniele De Rossi and Giulia Di Fede, from Politecnico di Torino. I understand that the project is designed to gather information about “Researchers in the fields of Computer Science who are in the process of writing an article, find more information about related articles of interest and verify and elaborate the selected information.”

1. My participation in this project is voluntary. I understand that I will not be paid for my participation. I may withdraw and discontinue participation at any time without penalty. If I decline to participate or withdraw from the study, no one will be told as the interviews will be conducted anonymously and therefore my name won’t be registered.
2. Participation involves being interviewed by students of Politecnico di Torino currently enrolled in the course of Human Computer Interaction [02JSKOV]. The interview will last approximately 20-30 minutes. Notes will be written during the interview. An audio tape of the interview and subsequent dialogue will be made. If I don't want to be taped, I will not be able to participate in the study.
3. I understand that the student will not identify me by name in any reports using information obtained from this interview, and that my confidentiality as a participant in this study will remain secure. Subsequent uses of records and data will be subject to standard data use policies which protect the anonymity of individuals and institutions.
4. Faculty and administrators from my campus will not be present at the interview, but will have access to the notes of the interview.
5. I have read and understand the explanation provided to me. I have had all my questions answered to my satisfaction, and I voluntarily agree to participate in this study.
6. I have been given a copy of this consent form.

Date _____ Signature _____

Signatures of the Investigators

_____

_____

_____

_____

Figure 10.1: Consent Form used for Needfinding interviews.

Bibliography

[1] Jakob Nielsen. "Enhancing the explanatory power of usability heuristics". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '94. Boston, Massachusetts, USA: Association for Computing Machinery, 1994, pp. 152–158. ISBN: 0897916506. DOI: 10.1145/191666.191729. URL: https://doi.org/10.1145/191666.191729.

[2] John Brooke. "SUS: A quick and dirty usability scale". In: *Usability Eval. Ind.* 189 (Nov. 1995).

[3] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL].