OOP Project Report – Group 59

Borislav Marinov, Edward Oh Jang Hun, Wiktor Cupiał, Jaouad Hidayat, Eduard Faraon, Eduardo Hernández

ABSTRACT

This report deals with the results from a conducted heuristic evaluation on a prototype of a GUI created on JavaFX.

1 INTRODUCTION

The objective of this report is to present the results of a heuristic evaluation conducted on a prototype of a GUI created on JavaFX for the Talio application. The Talio application is a project management tool that allows users to create and manage boards, lists, and cards. The application has four main pages: the homepage, the overview page, the board details page, and the card details page.

The homepage is where users establish a connection with the server. After clicking the 'Connect' button, the user is directed to the overview page. On this page, all existing boards are present, and there is a search bar that is still in the process of implementation. Users can create a new board by clicking the 'New Board' button, which takes them to the 'Board Details' page. On this page, users can see the specific details of a board, which includes a collection of lists with different cards inside of them.

Users have the option to go back to the overview page, customize and delete the board via the 'Settings' button that opens a smaller screen where options for background and font colour are presented. They can also enter a new list, delete an existing list, or add a new card inside of a list. Cards are added in a stack-like manner inside of a list. On each card, there is a 'Details' or 'Edit' button that allows the user to enter the card and see its title, description, and list of sub tasks.

On the card details page, users can add a new sub task via the 'Plus' button next to the sub tasks label and delete a sub task via the 'Delete' button next to each task. This page also has three additional buttons that allow for saving an edit to the card, canceling an edit, and deleting the whole card from the list. The presented video went through all of the implemented functionality, allowing a full view of the features.

To evaluate the design, a group of 6 experts were recruited. These experts were usability professionals, designers, developers, or other individuals who have a good understanding of the principles of usability. They were provided with a video of a working version of the interface and an interactive mockup to get a better understanding of the application and its functionalities.

The experts used 10 heuristics rules to evaluate the design, which included aspects such as navigation, layout, consistency, feedback, and error prevention. They graded each issue they found on a severity scale from 1-5, with 5 being the top priority for the team to fix. They also provided recommendations for how the usability issues can be fixed, which were actionable and specific.

In the following sections, we will present the results of the heuristic evaluation, describe how we will improve the application based on the results, and show the final GUI design.

2 METHODS

To do our evaluation we recruited 6 experts. They are experts that work in the field and they were familiar with the Heuristic Usability Evaluation and they have done it before. They have also researched different designs for an application with a similar purpose, meaning they already have a different perspective, as to what such an application requires.

In this report, we will describe in detail the procedure of conducting a heuristic usability evaluation, including the instructions for the experts, what they will be seeing, what they need to do step by step, and the heuristics they will be using.

The first step in conducting a heuristic usability evaluation is to select a group of experts who are knowledgeable about usability and have experience in evaluating designs. These experts can be usability professionals, designers, developers, or other individuals who have a good understanding of the principles of usability. In our case, a fellow group of students working on the same application was selected, as they have a generally good idea of the project and its goals.

2.1 Design evaluation process

Beforehand, the group of experts received a document containing the guidelines for this evaluation. Inside this document the team has placed clear instructions on how the review should be conducted. This includes using the 10 heuristics rules, following a certain feedback format for each issue found and grading this issue on a severity from 1-5 with 5 being the top priority for the team to fix.

After that the heuristic evaluators need to become familiar with the product or system they are evaluating. This may involve using the product or system, reading documentation, or talking to the developers. In our case a detailed presentation video of the product was provided alongside with an interactive mockup that the experts could use to get a better understanding of the application and its functionalities. In addition, step by step instructions have been included to further assist the evaluators.

The next thing to do is to evaluate the design against the heuristics: The experts will evaluate the design against a set of heuristics or guidelines. The heuristics may vary depending on the project or design being evaluated, but they typically cover aspects such as navigation, layout, consistency, feedback, and error prevention.

As a follow-up the evaluators will prioritize the usability issues based on their severity and impact on the user experience. They may use a scoring system or a ranking system to prioritize the issues.

Lastly, the experts will provide recommendations for how the usability issues can be fixed. The suggestions should be actionable and specific, and should include details on how the fixes can be implemented.

2.2 Heuristics used

The heuristics used in a heuristic usability evaluation can vary depending on the project or design being evaluated. However, for the goals of this course we have chosen 10 rules to follow. These include:

- Visibility of system status: This means that the system should always provide feedback to users within a reasonable time frame, so they are informed about what is going on.
- Match between system and the real world: A system should use language, concepts, and phrases that are familiar to users, rather than system-oriented terms. It should also follow real-world conventions, presenting information in a logical and natural order.
- User control and freedom: Users may choose system functions by mistake, so it's essential to provide an emergency exit that allows them to leave an unwanted state without going through a lengthy dialogue. It's also crucial to support undo and redo functions.
- Consistency and standards: Users should not have to wonder whether different words, situations, or actions mean the same thing. The system should follow platform conventions and use consistent language and visuals.
- Error prevention: The best error messages are those that don't need to be shown because the design has been carefully crafted to prevent errors from happening. However, if errors do occur, the system should provide clear messages that indicate the problem and offer solutions.
- Recognition rather than recall: Users shouldn't have to remember information from one part of the dialogue to another. The system should make objects, actions, and options visible and provide instructions when appropriate.
- Flexibility and efficiency of use: The system should cater to both inexperienced and experienced users, with features that can speed up the interaction for experts. Users should be able to tailor frequent actions and personalize the system.
- Aesthetic and minimalist design: Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.
- 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.
- 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.

In summary, we asked the experts to individually review mocks of our application and find as many problems as possible. We asked them to rate their severity in scale (1-5) and categorize them with respect to the previously mentioned heuristics.

3 RESULTS

For simplicity sake and display of results, we have calculated the average severity rating of each heuristic's problems. We have decided to show the final results in a table (Table 1) containing reduced data from the feedback document, the average and peak severity for each problem in the heuristics so we can clearly see what heuristic is in the most severe state overall. We also decided to create a column for the highest severity as this will help to distinguish heuristics that have a high deviation in the problems found, which allows us to determine outliers and more serious problems in each category. This approach also allows for quick demonstration into which heuristics have the most troublesome issues for the current state of the application and where the most improvements would be needed. For example, a severity rating of 1 and 2 suggests that it is not crucial to address this issue immediately and can be left out of this summary section, while a severity of more than 3 is definitely something to be quickly addressed and resolved.

Usability Heuristic	Average	Peak
Visibility of system status	3	4
Match between system and the real world	3	3
User control and freedom	3	5
Consistency and standards	3	4
Error prevention	3	3
Recognition rather than recall	2	2
Flexibility and efficiency of use	4	5
Aesthetic and minimalist design	3	3
Help users diagnose and recover from errors	2	2
Help and documentation	1	1

Table 1: Representation of the upwards rounded average severity for problems and the highest severity problem per heuristic.

The complete feedback document has underlined a range of problems with the heuristics of our prototype. Prioritizing which of these problems we tackle is a very important part of the post-evaluation process. As we do not have unlimited time, we need to rank them against each other by evaluation which ones are most fundamental for our application, the factors deciding this being the severity rating provided, the relevance we give this problem and the basic requirements for the app. The most important problems we have determined with help of the experts are the following:

- User confirmation is not required for destructive actions.
- Some user actions such as joining a server do not give the possibility being fixed if an error occurs .
- Some frequent user actions are hidden behind multiple, unnecessary clicks.

- Inconsistency was found in naming conventions with a Card and Task within a list as well as use of text or images in buttons.
- Board ordering in the overview is not intuitive as it is ordered by the time of creation.
- Missing functionality allowing users to delete cards and subtasks.
- A lack of a disconnect from the server button causing inconvenience for the user.
- Lack of consistency of the design, as components disappear and appear in certain conditions
- Impossible to delete a sub task
- Lack of scrolling for when components become too many to be displace on a single page

There were more highlighted issues that concerned themselves with minor changes in the interface that are not critical to an adequate user experience, which is why they were left out of the above list. The exact way we were able to separate between these issues and the more important ones is that we compared them to the minimum requirements that we were provided with when given this task and the severity that experts allocated to each problem. The more they were connected to basic functionalities and the higher the severity, the bigger the possibility for a problem to be determined as critical and included in this results section.

4 CONCLUSIONS AND IMPROVEMENTS

After conducting a heuristic evaluation on a prototype of a GUI created on JavaFX, we have identified several existing problems in the interface design that is currently in use for our Talio application. The evaluation was conducted by six experts in the field of usability who were provided with a video of a working version of the interface, and were asked to evaluate the design against a set of heuristics or guidelines.

Based on the results of the evaluation, we have identified several usability issues that need to be addressed to improve the user experience. These issues include problems with consistency, feedback, error prevention, and navigation.

To improve our application, we plan to make several changes to the GUI design. First, we will address the issues related to efficiency by ensuring that there will be an option for the user to delete sub-tasks and scroll through the panels as part of the "Flexibility and efficiency of use" heuristic. We will also work on improving feedback and error prevention by making sure that the user is provided with clear and concise messages when something goes wrong like showing an error box or preventing them from executing the action. This version should be better as the user will be aware that what they did is not allowed. This will further fulfill the "Error prevention" and "Help users recognize, diagnose, and recover from errors" heuristics.

In terms of navigation, we will work on making it easier for the user to move between different pages of the application by providing clear and intuitive navigation buttons. This would allow the application to feel more consistent and smooth, therefore improving the overall user experience with this application and satisfying the "Recognition rather than recall" heuristic. We will also make sure that the search bar is fully implemented and functional to enable easy access to different boards.

From Table 1. in the results section, it is clear to see that the most problematic heuristic is the flexibility and efficiency of use. Most of the problems posed by the experts for this category can be resolved with the addition of a single button. A deletion button will be added to the component of a sub task, which will then allow the user to delete a sub task at any point, making it easier to arrange their action plans. One unique issue we discovered was the lack of a scroll bar, which should be added in order to allow for scrolling and accessing all different boards, cards, or lists.

Finally, we will work on improving the overall look and feel of the application by making use of a more visually appealing design with more clear color schemes and typography.

Overall, we believe that these changes will lead to a significant improvement in the user experience of our Talio application.

Conclusion is now over 400 words.