OOP Project Report - Group 64

Ioana Forfota, Emilia Ketterer, Tudor Măgirescu, Lazar Polovina, Dimitar Smenovski, Ziliang Zhang

ABSTRACT

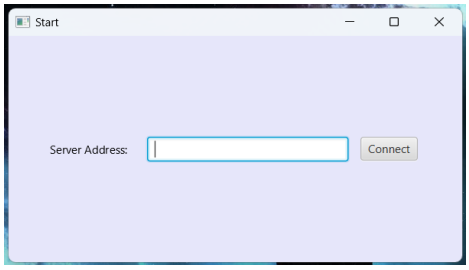
Over the short course of this project, our team has attempted to develop an application allowing a user to better organize themselves. A heuristic usability evaluation is "a usability engineering method for finding the usability problems in a user interface design so that they can be attended to as part of an iterative design process" [1]. Conducting such an evaluation on our application is vital in the early phase of the project, as the results allow improvements to be made to the application, developing the prototype. Through obtaining feedback from experts, our team was able to find multiple areas of improvement, ensuring a better experience for the client when using our final graphical user interface.

1 INTRODUCTION

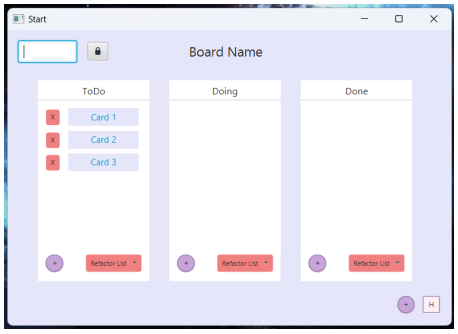
As stated, the aim of this project is to develop an application allowing a user to better organize themselves, specifically by keeping track of thoughts and to do's in the form of cards. During the planning process we created multiple mock ups of the graphical user interface of this application. While creating these we attempted to focus on minimal designs, clearly displaying the main features of the application and communicating their purpose through stylistic choices. We also wanted to clearly split the different scenes of the application by changing their default color, ensuring that there would be a separation between the main scenes and more specific features. To test whether we achieved all of these goals and find ways to further improve our application, our team completed this heuristic usability evaluation.

2 PROTOTYPE

When running the prototype of the minimal application, the user first sees the following scene:



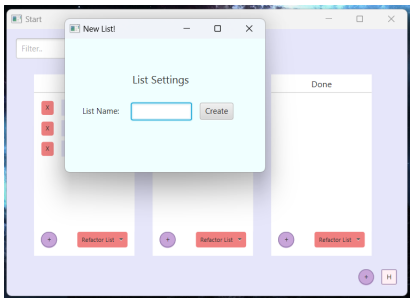
In this scene, they can input the server address they wish to connect to. If a valid address is input, the user can click the 'Connect' button and will be connected to the specified server. If this server address does not exist, the user will see the message 'invalid' in the textbox. Once a valid address is input, the user sees the following scene:



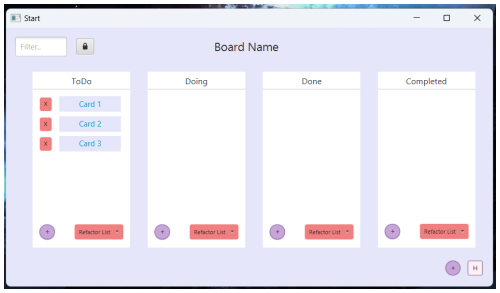
This main scene displays a board with three lists (ToDo, Doing, Done), of which ToDo contains three cards (Card 1, Card 2, Card 3). The user now has multiple different ways to interact with the board.

Creating a new list:

Clicking on the purple circular button in the bottom right of the board causes the user to see a new additional scene:

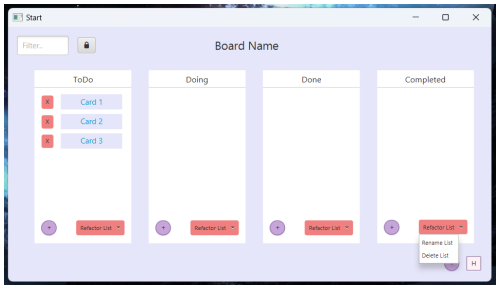


In this ‘New List!’ scene, the user can specify the name of the new list they want to create in the textbox. Once the ‘Create’ button is clicked, the new list is created and appended to the rest of the lists. For example, if the user were to type ‘Completed’ into the textbox and click ‘Create’, the board would change to the following:

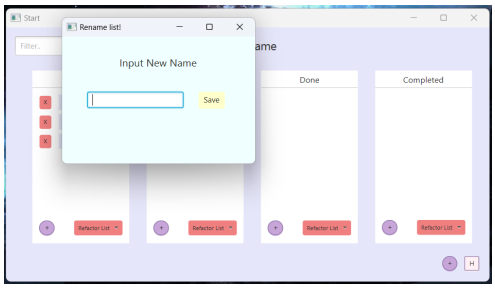


Refactoring a list:

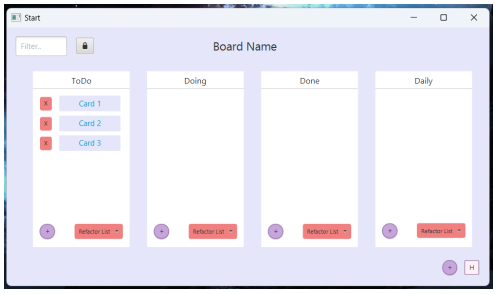
In addition to adding new lists, the already existing lists can be renamed and deleted using the ‘Refactor List’ button in the bottom right of each list. If the user wanted to refactor the newly added list they would see the following options:



Clicking ‘Rename List’ opens the following new additional scene:

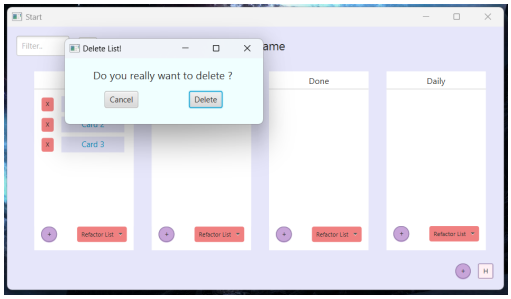


In this scene, the user can input the new name for the specific list. For example, if the user wanted to rename the last list to ‘Daily’, they could type this and click ‘Save’. This would result in them seeing the following scene:

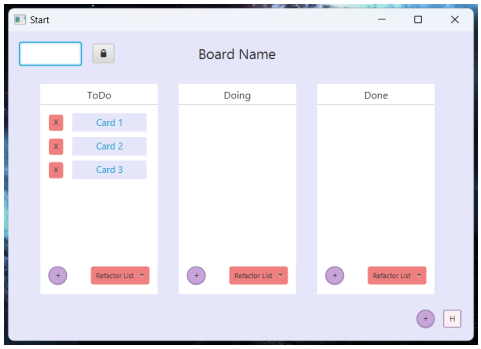


From here, the user has the same options as before. If they now realized that the ‘Daily’ list does not have much use for the board, they could click on ‘Refactor List’ again and then click ‘Delete List’. This would cause the user to

see the following additional scene:

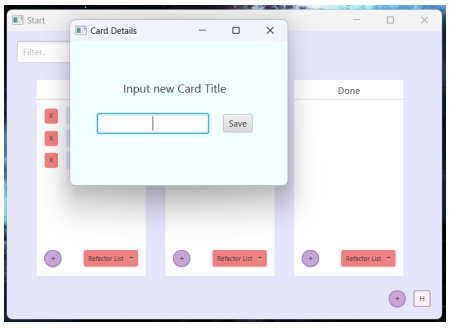


Clicking cancel would mean no changes happen to the board. However, if the user clicks ‘Delete’, the board reverts to the following state:

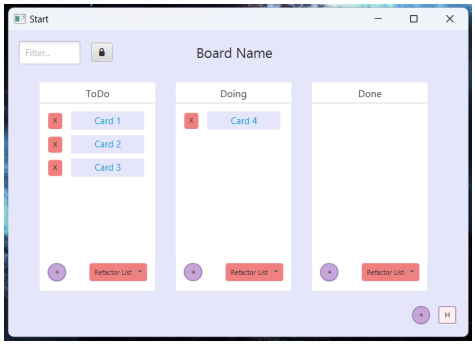


Adding a card:

In any of the existing lists, a new card can be added. This action is list-specific, and requires the user to click on the circular purple button in the bottom left of the list they wish to add a card to. For example, if the user wishes to add a new card to the ‘Doing’ list, they could click the button in the bottom left of this list, resulting in the following scene:

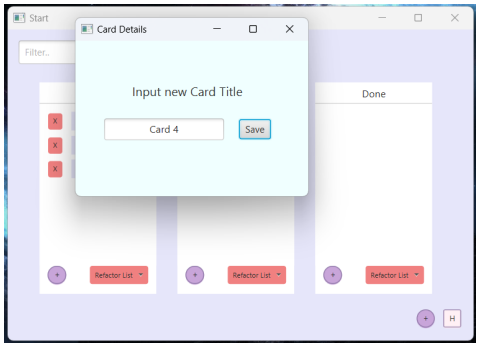


In this scene, the user can input the name for a new card. If they type ‘Card 4’ and hit the save button, the user would see the following main scene with the new card:

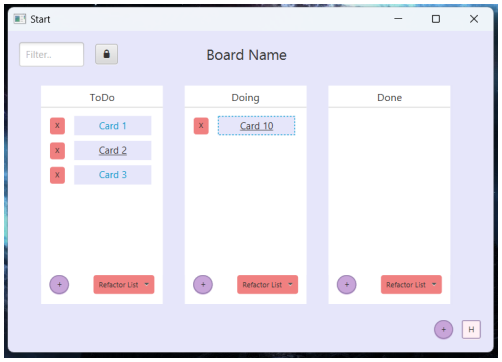


Renaming a card:

If the user wanted to rename a card, they could simply click on the card they wish to rename. For example, if the user wanted to rename the new card they created (‘Card 4’) they would see the following scene:

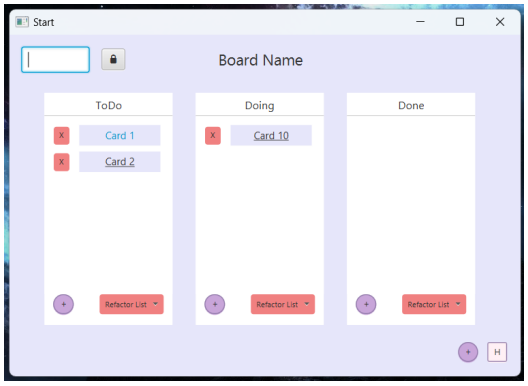


If they changed the text inside the text field to ‘Card 10’ and clicked ‘Save’, this new name would be displayed, meaning the board would now look like the following scene.



Deleting a card:

If the user wanted to delete a card, they could do so by clicking the red button with an ‘X’ next to the respective card. For example, if the user wanted to delete ‘Card 3’, they could simply click the red button with the ‘X’ next to it, changing the board to the following:



3 METHODS

To complete this heuristic usability evaluation, we chose another team of six people as our evaluators. Since they are also a team creating a similar application to ours with the same goals, we hope that they will be well informed on what features the application should have. This should allow them to judge the implementation of these features. We gave these experts access to our prototype, as included above, and asked them to go through it alone, noting down anything they noticed about the design specifically related to the following heuristics.

1) Visibility of system status
2) Match between system and real world
3) User control and freedom
4) Consistency and standards
5) Error prevention
6) Recognition rather than recall
7) Flexibility and efficiency of use
8) Aesthetic and minimalist design
9) Help users recognize, diagnose and recover from errors
10) Help and documentation

After noting down a minimum of three issues, we asked them to discuss among themselves (as a group) about these issues, compiling them into one file that can then be sent to us. We also asked for these final issues to be written using the following formatting:

1) Description of the problem
2) Possible cause of the problem
3) Possible solution to the problem

This formatting should allow us to easily follow our experts' thoughts and analyze the results we obtained on how usable our interface is.

4 RESULTS

We received quite a long list of problems from our team of experts once they looked over our prototype for the minimal application. To begin processing these results, we first split the problems into two categories, a list addressing issues we had already planned on improving in our design when going beyond the minimal application, and a list of problems we had not previously considered. We decided not to include the first list of problems in our analysis of the feedback since we had already considered them. To summarize these problems shortly:

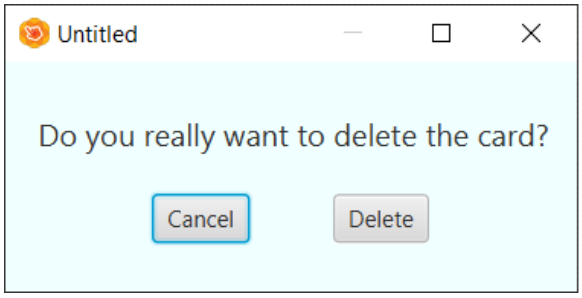
1) “There is no disconnect button, preventing a user from leaving a server.” This button will be added in sprint 2.
2) “There is no explanation for the buttons on the board (i.e. “H”, “lock”, “filter”).” We will add a documentation under the “H” button - which we will rename to “help” for clarity - specifying the actions linked to each button.
3) “Fast paced work-flow is not possible because of the lack of keyboard shortcuts.” We plan on adding these to the application once more pressing features have been implemented.

After filtering out these issues, we were left with 9 problems relating to our minimal application that we had not previously considered. We organized them by priority, voted on by all team members based on what we deemed most important for a client, and went through possible improvements related to each problem. The problems are listed with their respective improvements in the next section.

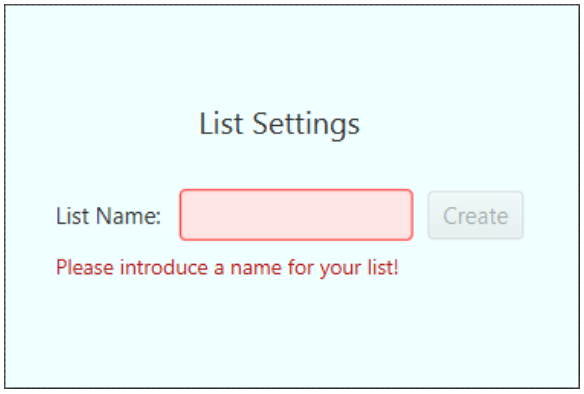
5 IMPROVEMENTS

Based on this feedback, there are some changes that we would like to make to the application, improving the general experience for the user.

The first problem, regarding having a warning when deleting a card, is a crucial oversight on our behalf. This should be implemented to ensure the user does not delete a card without a simple way to recover from this error. This becomes especially important when more information can be added to a card, making it difficult to simply recreate it. Therefore, even though it might slow down the work-flow of the application, we deem this a very important problem that we will implement. For this, we will add the following scene to our application:

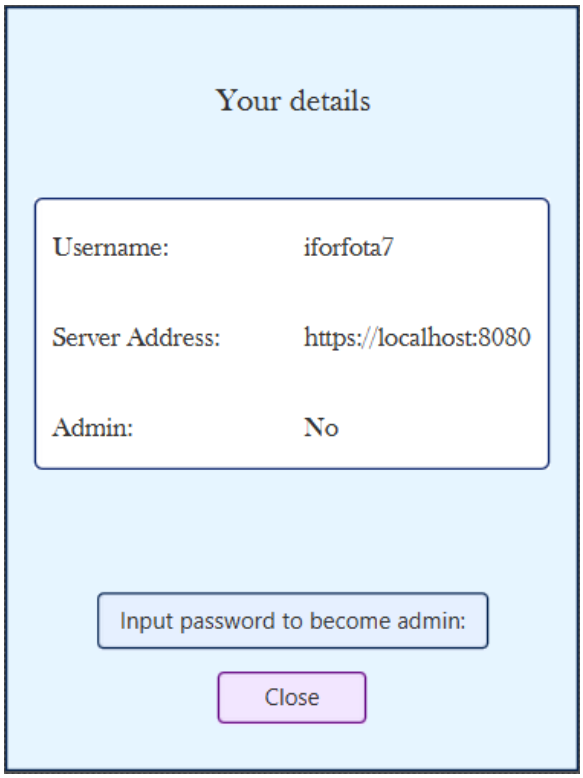


Second on the list of problems is the current ability of a user to create a list or card without a name. As pointed out by the experts, it does not make sense for a user to be able to create such a list or card, as it would serve no purpose. A simple way to solve this problem would be to prevent the user from being able to click the “save” button when creating or renaming a new card or list if the textbox for the name is empty. In this case, the user could be informed by a warning message that appears below the text box. Through this simple change displayed below a lot of future errors and confusion could be avoided.

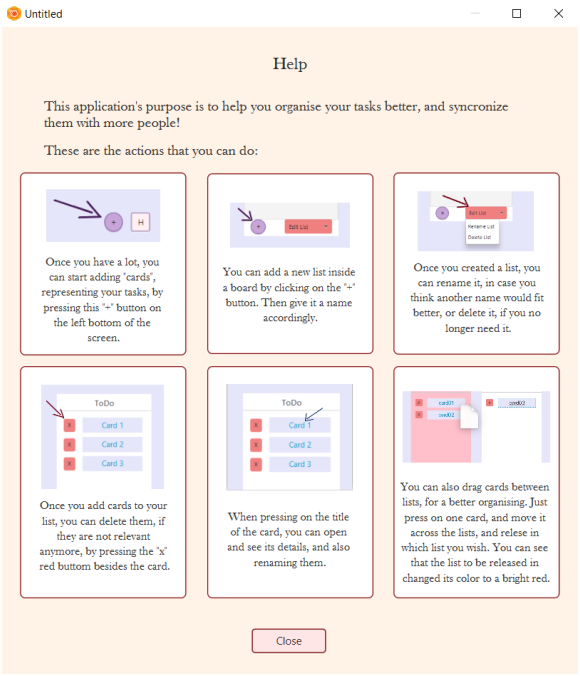


Currently, the user cannot see what server they are connected to. This has to be remembered by the user while they are running the application. Having the server address displayed somewhere on the board, would spare the user the difficulty of remembering it, making it more pleasant to use the application. To maintain a minimalist design, it could be beneficial to add a small

button with an information icon in the corner of the board, which once clicked opens a new scene with the server information. Additionally, this scene could also display the username, whether the user is an admin, and present the option to become an admin.



Regarding the fourth problem, being the unclear naming of the add list button, we would like to consider multiple options. As stated, the current button is not clear in its labeling, leading to confusion for the user. However, we would like to maintain a minimal design and therefore not change the button to have a name such as “add list”, since this would take away from the current aesthetic. Another solution would be to explain the use of the button in the “help” scene, which we already plan on implementing and can be accessed directly through a button named “help” (currently “H” in the prototype). The help scene might look as follows.



Fifth on the list is the problem that the word “refactoring” used in lists might be ambiguous and too technical for a general user. This is a very good consideration we missed, and we will therefore change the button to “edit list” instead, making it less technical and clearer.

The sixth problem, regarding the detection and prevention of potential errors such as duplicate names of a card, or list, is one we had not considered. This was mainly because, on the database side, these duplicates would have no effect since each list and card has its own generated unique id. The experts are right that this could lead to confusion for the user, since they might accidentally add duplicates and then no longer be able to tell them apart. However, we think this is a minor inconvenience, and not even necessarily an error on the users behalf, so we will not implement any changes regarding this issue.

As the experts pointed out, there is no clear indication that tells the user that they have been successfully connected to a server except for the

board overview scene appearing. We thought that the board overview becoming accessible would be enough of an indicator to the users that they have been successfully connected, and we will likely not include a more direct message since working on other features appears more pressing to us.

The eighth problem listed states that there is no undo feature for deleting a card or list. This is true, however we did not deem this necessary since there is a warning upon deletion that ensures users can not accidentally delete cards or lists. Therefore, we will likely not make any changes to the application based on this problem.

The last problem listed by the experts was that the user has no indication whether a list has been successfully stored. While we acknowledge that there is currently no specific feature to inform the user that a list has been stored, we believe that this is quite intuitive for the user since the list appears on the scene once it has been created. We will therefore not make any changes to address this issue in the near future.

6 CONCLUSION

By getting feedback from experts and obtaining a list of problems with our graphical user interface, we have managed to analyze our application in more detail. While going through the list of problems, our team was able to find solutions to challenges we hadn't even previously considered, ensuring we could present our users with the best possible interface. We have come to the conclusion that our application needs a new scene confirming the deletion of a card, a system to check that no lists/cards are created without a name, a more general information scene with the server address, a more detailed help scene, and a more clearly labeled "edit" button for lists. By implementing these changes (mocks shown in section 5), we can ensure that our graphical user interface will more closely adhere to the listed heuristics, and consequently provide the user with a better experience when using our application.

REFERENCES

[1] Jakob Nielsen. 1 Nov 1994. How to Conduct a Heuristic Evaluation. In *NN/g Nielsen Norman Group*.