Usability Inspection Report Template
Based on Usability Aspect Report (UAR) Template from Brad A. Myers and Bonnie John
http://www.cs.cmu.edu/~bam/uicourse/UARTemplate.doc

Complete this form *for each* problem that you identify. An empty form, suitable for use in your actual report, in on the next page.

#: Unique id #	**Problem/Good:** Indicate if problem or good aspect
Name: Succinct but descriptive name for problem or good aspect	
Relevant heuristic: State the most relevant heuristic	
Evidence of issue: Indicate where the issue is within the user interface. In addition to interface facts, pictures are almost always necessary and usually faster to produce than words alone.	
Detailed explanation: **Maximum 100 words** A detailed explanation of how the relevant heuristic is violated (for problems) or met (for good aspects). If making assumptions about user behavior (e.g. what the user will or will not be familiar with) include evidence to support your assumptions.	
Severity or Benefit (minor, major, critical): Use a scale of 2 (minor), 3 (major) or 4 (critical) **Justification:** Justify your numerical severity rating with the factors of frequency, impact, and persistence. Also include how you assessed their weighting. **Maximum 60 words each** Describe your weighting of severity factors **Maximum 60 words**	

#: 1	Problem/Good: Problem 1

Name:
Cannot retrieve the target file in the search bar if user enter only the partial name of the note

Relevant heuristic:
Recognition rather than recall

Evidence of issue:
The system cannot perform name retrieval by partial name of a note provided by the user.
As can be seen from figure 1, if users enter the full name of the note, they can retrieve the results, but figure 2 shows that if users only enter part of the name of the note, they cannot retrieve any notes that contain this field.

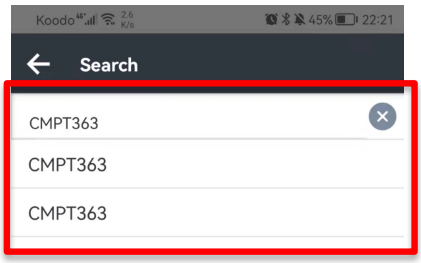


Figure 1: Joplin search page, full name of note can show results.

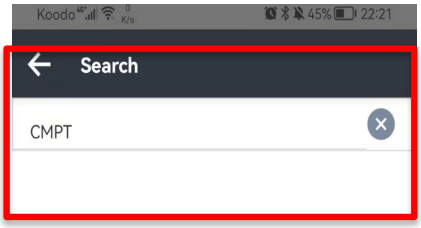


Figure 2: Joplin search page, partial name of note cannot show any result.

Detailed explanation:
The notebook contains a note named “CMPT363". If the user cannot remember the full name of a note and intends to use a memory fragment for the note name as the search term (E.g. use “CMPT” to search). Joplin does not show any results since it cannot be retrieved by part of the note name or letters. The only way to search is to enter the full name of the note. This increases the information that users must remember, that is, increases the cognitive workload of users, so it violates the usability heuristic of “Recognition rather than recall”.

Severity or Benefit (minor, major, critical):
4 (Critical)

Justification:
Frequency
This question is triggered relatively frequently. For new users, the frequency of search is not very high, because the number of notes can only be searched one by one. But with a longer push, the user must use the search bar to find more and more notes. From the perspective of long-term product development, this is a very common problem.

Impact:
This is an insurmountable problem, which greatly affects the user experience because the user must completely remember the name of the note to perform the search function. It can also be very frustrating for users that they can't retrieve anything every time.

Persistence:
This is not a one-time problem. Although the user discovered the problem, it is not possible for the user to remember the exact name of every note. This problem will haunt users repeatedly and users will refuse to use the feature again

Weighting:
This problem has a high impact and persistence. Although it is not very frequent for users who just use the software, it will become a difficult problem in the future. This search function is a Level 4 issue that must be fixed before the product is released

#: 2	**Problem/Good:** **Problem 2**

Name:
Note the name is repeated without prompting

Relevant heuristic:
Error prevention

Evidence of issue:
When users create a note with the same name, no hint is given that there is already software with the same name, or (1) will be added to the same name files like notability apps. This can lead to confusion when the user later looks for the file and needs to click on the note to confirm that the content is in the note.

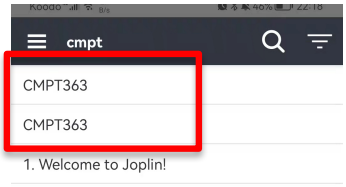


Figure 1: Joplin note page. Creating the note with the same name that already existed

Detailed explanation:
When users use the notes Explorer function of Joplin, their search box does not provide the associated search terms, which means that users must remember the name of the note they want to find, because Joplin cannot prompt them for the associated note. This increases the information that users must remember, that is, increases the cognitive workload os users, so it violated the heuristic of recognition rather than recall.

Severity or Benefit (minor, major, critical):
2(minor)

Justification:
Frequency:
This is because users sometimes do not remember that a note with the same name has been created. For example, users often create a note called “draft”, but they may use "draft" again as a temporary document. Users will inadvertently create notes with the same name.

Impact:
This is a very difficult issue to overcome because the user will be very confused when two files with the same name are retrieved, because there is no prompt at the time of creation. So much so that they have to click on the note to make sure.

Persistence:
This is a one-time problem. When users realize that files have the same name while retrieving them, they change the name of the note to distinguish them.
Weighting:
This problem has a high frequency, a high impact but not persistence. Because, the problem of having the same name as a note can lead to confusion, with no indication that a file with the same name has been created and time wasted trying to verify the contents of the note. But you can fix the problem in time by changing the note name. Therefore, the problem has two levels of severity.

#: 3	**Problem/Good:** **Problem 3**

Name:
System does not give feedback when no result is retrieved

Relevant heuristic:
Visibility of system status

Evidence of issue:

When the user cannot retrieve any results, the system does not give any hints to show the system status

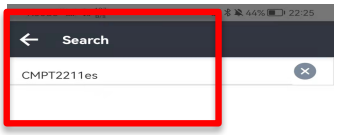


Figure 1: Joplin search page, the red square shows no any result when user enters.

Detailed explanation:
After the user enters a search term and starts a search, if the system does not find any results, the system will only show a blank at the bottom of the search bar. The user does not know what is happening, the user may assume that the search cannot be performed because of a software problem, or the results cannot be retrieved because of a network problem, or even begin to doubt that the search has even begun. The system does not clearly communicate system status to the user, so it violates the usability heuristic of “Visibility of system status”.

Severity or Benefit (minor, major, critical):
4(Critical)
Justification:

Frequency
This is a very common problem because every time users use the search bar function they encounter this problem. For first-time users, he takes time to figure out since the system does not give any feedback.

Impact:
This is a more difficult problem to overcome, because users will enter a panic phase with software that is completely unresponsive. They will start to reflect on what is wrong with their operation and make multiple attempts, which will undoubtedly consume the user's time. Even users who know about this problem will habitually wait a few more seconds, which undoubtedly causes a great impact.

Persistence:
This is not a one-time problem. Although users find this problem, users still wait a few more seconds after searching. Since this problem will cause users to habitually wait every time they encounter it, this is not a one-time problem.
Weighting:
This problem has caused a serious impact on user experience and users will continue to encounter this problem in the future. This can cause users to stop using the search function and even lose interest in using this software. Problems that confuse users are disastrous problems. So this is a Level 4 issue that needs to be addressed before the product is released.

#: 4	**Problem/Good:** **Problem 4**
Name: The help document is not easy to find the function that user need.	
Relevant heuristic: Help and documentation	
Evidence of issue: When users use new features of the software, they need to go back to the help document to learn. However, the help document does not directly show the functions that users want to learn. Instead, users need to read the help document by themselves. It is not easy to use, because users do not know the position of the function to be consulted in the help document, and there is no direct search or link for users to reach the location of the target function.	

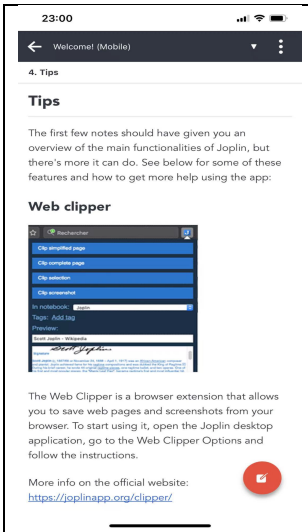


Figure 1: Users need to browse through the documentation of the features they want to find

Detailed explanation:
Instead of providing a direct link to a list of features, the help document requires users to browse and find it themselves. When users encounter problems in using a certain function, they need to consult the help document and then go back to the Note to use it. This makes it inconvenient for users to use. It would be more convenient to provide a help option when using this function. Or a function option button in the help document that makes it easier for users to use. So it violates the usability heuristic of “Help and documentation”

Severity or Benefit (minor, major, critical):
3(major)

Justification:
Frequency:
The frequency of this problem is relatively high, users will be able to use help files when they use Joplin with new features. So the frequency is going to be high.
Impact:
This is not a difficult problem to overcome because users must try to use some functions and they need to know how to use some functions
Persistence:
This is not a one-time problem, because users will use the help documentation each time when they want to know how to use a certain function, and the time will be wasted.
Weighting:
This problem has high frequency, and not a one-time problem, but not difficult to overcome. Some users can try the function to use, but not all users can do like that. So, they need to open the help documentation each time to find the function to learn how to use some functions. Therefore, the severity of the problem is three.

#: 5	**Problem/Good:** **Problem 5**

Name:
There is no undo function when copying multiple files

Relevant heuristic:
User control and freedom

Evidence of issue:
No undo operation is provided when the user chooses to copy one or multiple files. Below image shows what happens after my unexpected operation, with each file accidentally copied three times

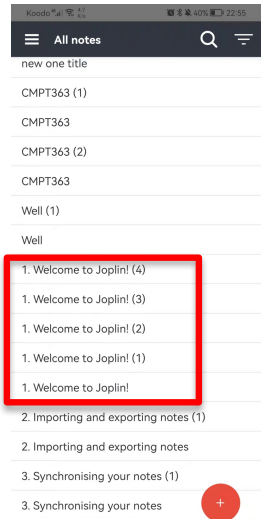


Figure 1: The red square shows all copied files after using Copy operation.

Detailed explanation:
If a user selects one or more files, the system copies the files with number differentiation. However, if the user wants to undo that “copy” operation, find that there is no way to recover that. The only way to do this is if the user chooses all discard files and deletes them. When a user mistakenly performs a copy operation, there is no clear way to undo the operation, so it violates the usability heuristic of “User control and freedom”.

Severity or Benefit (minor, major, critical):
2 (Minor)

Justification:
Frequency
The frequency of this problem is not very high, only when the user uses the copy file function and needs to undo the copy operation. For some users they will rarely undo this action

Impact:
This is a difficult problem to overcome. Once the user performs an improper copy operation for various reasons, they need to be "revoked" of that operation. The only way is to delete each copied file one by one. This will greatly increase the user's usage time and will lead to the user's disgust with the software. This will also have a serious impact on the reputation of the software.

Persistence:
This is a one-time issue, and once users discover it, they are careful to avoid improper copy operations. Once the inappropriate operation has been avoided, there is no need to undo the operation
Weighting:
This problem has a serious impact, and it takes a lot of time to fix the operation once the user encounters it. However, this problem is uncommon and once it happens, users will subconsciously avoid it to happen again. So it is a Level 2 problem that needs to be fixed with low priority.

#: 6	Problem/Good: Problem 6

Name:
After filter is used, the content of sort by is not displayed on the interface.

Relevant heuristic:
Recognition rather than recall

Evidence of issue:
After selecting the filter, and back to the notebook interface. Sort by what is not shown on the interface. The user can only click the filter to view the filter again.

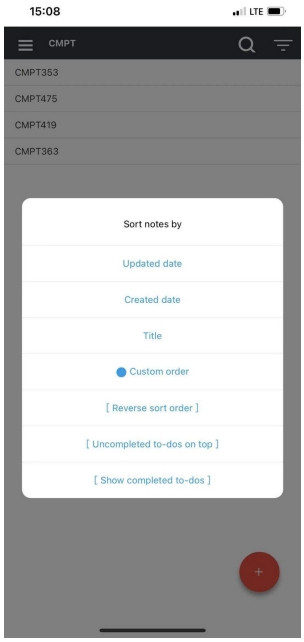


Figure 1: Joplin using filters to select sort notes by what.

Figure 2 : After selecting the option of the filters, there is no display sort by what.

Detailed explanation:
When the user uses the Joplin's filter, none of the boxes displays the contents of sort by, forcing the user to remember the filter, that is to violate Recognition rather than recall. Joplin's Filter function should tell the user the contents of the Filter on the notebook page after it is used. So it violates the usability heuristic of "Recognition rather than recall".

Severity or Benefit (minor, major, critical):
2(minor)

Justification:

Frequency:
The frequency of this problem is relatively high, users will frequently use the filter to find the notes, for example, if they forget the note's name, and they need to find the notes according to the created date or updated date. etc. So, there should be a box to show sort by what after selecting the filter.

Impact:
This is not a difficult problem to overcome because there are not many options contained in filters. So, users can overcome and remember what they selected.

Persistence:
This is not a one-time problem, because users will not remember filters each time.

Weighting:
This problem has high frequency, and not a one-time problem, but not difficult to overcome. Some users can try to remember the filters after using, but not all users can remember it each time. So, if they can't remember the filers, they need to open the filter again to check what was sort by. Therefore, the severity of the problem is three.

#: 7	Problem/Good: Problem 7

Name:
The operation of moving files is not consistent

Relevant heuristic:
Consistency and standards

Evidence of issue:
As shown below, if the user wants to move their notes to another notebook in Figure 1, there will be a text prompt in the inverted triangle area at the top, while in Figure 2 there will be no prompt if you want to move your notes to another notebook.

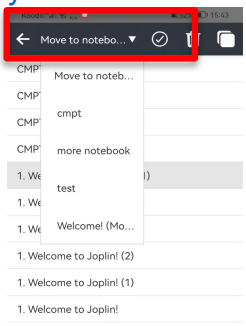


Figure 1: Joplin notes page, have text at the top

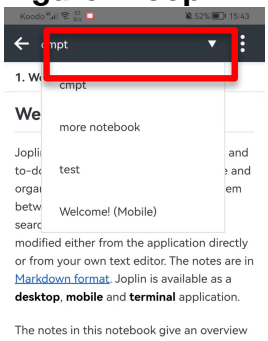


Figure 2: Joplin note detail page, no text at the top.

Detailed explanation:
If the user selects a note in a notebook interface and long-presses the note, "move to notebook" will be clearly displayed in the gray horizontal bar above the software. The current action is clearly displayed and the drop-down menu shows the name of the notebook to move to. However, there is only an inverted triangle at the top of the note's detail page. If the user clicks, it will only display the drop-down menu and no specific operation instructions. The same operation is not consistent, which violates the Usability heuristic of "Consistency and standards".

Severity or Benefit (minor, major, critical):
2(Minor)
Justification:
Frequency
This problem is common and occurs every time users want to move notes from within a note to another notebook. For users who have the habit of sorting out, this problem is also often encountered
Impact:
This is an issue of relatively little consequence. Especially for novice users, who can easily cause confusion after doing this operation. The new user chooses not to use the action due to failure to display any obvious prompt. Failure to achieve system consistency always results in forcing users to learn new things, thereby causing cognitive load on users.
Persistence:
This is a one-time problem, and when the user discovers the existence of this problem and understands what the action really means, the user is forced to learn the action and equate it with the marked move action (in note page). Users will know how to deal with that problem after they are forced to learn.
Weighting:
This is a common problem for users especially if they are using the software for the first time. For new users it will also have a serious impact on them. Forcing users to learn new actions is a bad thing, and maintaining consistency will increase the frequency of users. Since it's a one-time problem, users will quickly learn how to deal with it, so it is a Level 2 problem that needs to be fixed with low priority.

#: 8	**Problem/Good:** **Problem 8**

Name:
There is no recycle bin to store the notes which are deleted by users

Relevant heuristic:
Error prevention

Evidence of issue:
After users delete notes, there is no recycle bin to store the notes to give the user a chance to restore them.

Figure 1: There is no Recycle bin for deleted notes.

Detailed explanation:
When the user needs to retrieve deleted Notes, the Recycle bin is not available to restore notes, that violates the Error prevention. As a result, the user cannot find the deleted notes and may regret deleting them. Sometimes there is something very important in your notes, but after you delete it there is no way to get it back. So it is violates the Usability heuristic of "Error prevention".

Severity or Benefit (minor, major, critical):
3 (major)

Justification:
Frequency:
The frequency of this problem is relatively high. Users tend to reorganize notes so often that they delete notes that they don't think are useful at the time, but they can't restore notes after deleting them.
Impact:
This is a difficult problem to overcome because users cannot predict whether they will need notes in the future. And, without a Recycle bin, the notes are deleted forever.
Persistence:
This is a one-time problem, because the user will back up notes or will no longer delete notes easily.
Weighting:
This problem has high frequency, and is difficult to overcome, but a one time problem. This is because every time a user needs to clean up and delete unused notes, they are unsure whether they will be needed in the future. There is no way to restore notes in the Recycle bin after they have been deleted. This can cause users to feel very anxious and helpless because Notes are gone. But the user will decide not to delete the notes easily or back up. Therefore, the severity of the problem is three.

#: 9	**Problem/Good:** **Problem 9**

Name:
The setting tag page has a large space in the middle

Relevant heuristic:
Aesthetic and minimalist design

Evidence of issue:
In the interface of setting tags, there is a large space between typing the new tag box and the “OK and Cancel” button.

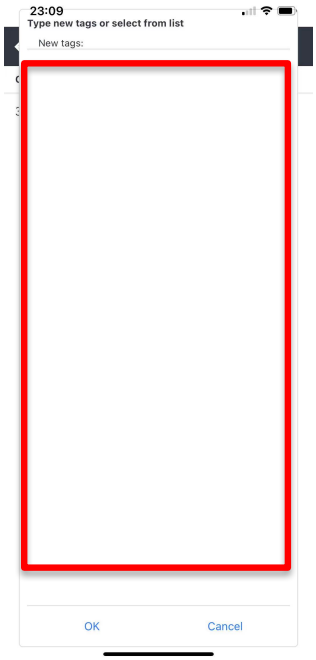


Figure 1: Joplin setting tags page.

Detailed explanation:
The interface of setting tags has a large space, which violates the UI design and Visual elements in Aesthetic and minimalist design. In this way, the input box for the new tag and the button below will be very obscure. Users are not used to looking at both ends first. This leads to confusion when entering the setting tags.

Severity or Benefit (minor, major, critical):
2(minor)

Justification:
Frequency:
The frequency of this problem is relatively high. Because users often set a note to a new tag, and the setting tag interface always displays a large blank in the screen.

Impact:
This is not a difficult problem to overcome because users will find the typing box and “OK”and “Cancel” button when they look at both ends.

Persistence:
This is a one-time problem, because the user would realize the typing box and the button at both ends.
Weighting:
This problem has high frequency, but is not difficult to overcome, and a one-time problem. This is because users can overcome this problem after one-time use. But it is a common problem, and makes users uncomfortable to set a new tag. Therefore, the severity of the problem is two.

#: 10	Problem/Good: Problem 10

Name:
No function of searching notebook

Relevant heuristic:
Flexibility and efficiency of use

Evidence of issue:
As shown in the figure below, all notebooks are displayed in the sliding window on the left, but there is no function to search notebooks.

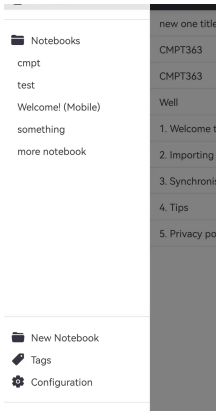


Figure 1: Joplin left sliding window, shows all notebooks.

Detailed explanation:
Users can search for the note through their name with the “magnifying glass” button on the top right. After clicking the three-layer button in the upper left corner, the left mezzanine will be expanded. The names of all notebooks are displayed in it, but the user cannot search for notebooks on this page. All the notebook names will show here and users must find it name by name if they need. So it violates the Usability heuristic of "Consistency and standards". So it violates the Usability heuristic of " Flexibility and efficiency of use".

Severity or Benefit (minor, major, critical):
2(Minor)

Justification:

Frequency
The frequency of this problem is not very high, the user must have many notebooks to choose to use the search engine and encounter this problem.

Impact:
This problem is easy for users to overcome and difficult for most users. Users can search by notebook name, of course, if there are too many classified notebooks, it will take users a lot of time to do this.

Persistence:
This is a one-time problem that users can overcome in other ways once they understand the problem

Weighting:
Overall this problem is not a serious problem, the user is only triggered when using search notebooks and can use another way to overcome this problem, but in extreme cases, if the user really has many notebooks, Forcing the user to use a one-by-one search method is very unreasonable and consumes a lot of time for the user. So the problem is that you can give a lower priority to solving the problem. (Level 2)

#: 11	Problem/Good: Problem 11

Name:
Editing mode cannot be pushed out by swiping down the screen or click “down “button

Relevant heuristic:
Consistency and standards.

Evidence of issue:
When the user has finished typing, there is no way to swipe down the screen or “done” button, to push out edit mode.

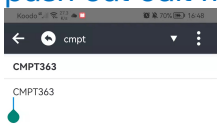


Figure1: Joplin note editing page, user cannot swipe down the screen to finish typing, or “done” button.

Detailed explanation:
When the user has finished typing, there is no way to swipe down the screen or “done” button, to push out edit mode. When mobile users are finished typing and editing, they usually swipe down the screen or “done” button to complete the editing. Notability, for example, allows editing mode to be launched by swiping down after typing on the keyboard. New Joplin users won't be used to hitting the back button to exit editing. It violates Consistency and standards.

Severity or Benefit (minor, major, critical):
3(major)

Justification:
Frequency:
The frequency of this problem is relatively high. Because users often need to type the notes. They often need to hit the back button to finish editing when using the Joplin.
Impact:
This is not a difficult problem to overcome because users need to get used to clicking the back button in the upper left to finish editing.
Persistence:
This is not a one-time problem, because users need to get used to hitting the back key in the upper left to exit the edit, rather than getting used to it all at once.
Weighting:
This problem has high frequency, and not a one-time problem, but it is not a difficult problem to overcome. Users need to get used to using the back key in the upper left to exit the edit. It can be very uncomfortable and at a high frequency to use, but it can be easily overcome. Therefore, the severity of the problem is three.

#: 12	Problem/Good: Problem 12

Name:
When searching for some particular letters, it cannot display a note containing these letters.

Relevant heuristic:
Consistency and standards

Evidence of issue:
When a user searches for some letters, the system does not display notes containing these letters. As figures shown below, Figure 1 shows the notes that can be searched for containing the letter "A" while Figure 2 shows that using the letter "C" without any results

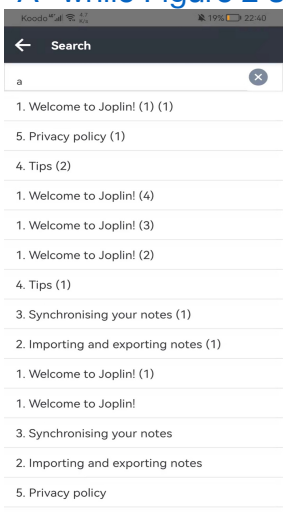


Figure 1: Joplin search page, when searching letter ‘a’

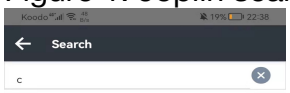


Figure 2: Joplin search page, when searching letter ‘c’

Detailed explanation:
After the testing, when the user uses a single letter: ‘a’ ‘b’ ‘d’ ‘e’ ‘f’ ‘i’ ‘s’ ‘v’ ‘x’ as the search term for searching operation, the system can retrieve all the notes containing these letters. If the user uses a single letter ‘c’ ‘g’ ‘h’ ‘j’ ‘k’ ‘l’ ‘m’ ‘n’ ‘o’ ‘p’ ‘q’ ‘r’ ‘t’ ‘u’ ‘w’ ‘y’ ‘z’ as the search term for searching operation, the system cannot display any results notes containing these letters. This question shows it not in compliance with industry regulations, so it violates the Usability heuristic of "Consistency and standards".

Severity or Benefit (minor, major, critical):
2(Minor)

Justification:

Frequency
This problem occurs very frequently, whenever users use the search function and want to search by searching note keywords.

Impact:
This problem can have an insurmountable impact on users. If the user finds that part of the letter is searchable, the user will be confused. When users repeatedly encounter such problems when using search, users will give up searching notes by partial letters. It also affects the reputation of the product.

Persistence:
This is not a one-time problem, and when the user finds out that the problem exists, they can't have the patience to find out which letters can still be searched and memorized. It is more likely that the user will avoid using the feature again or that the user will be repeatedly plagued by the problem.

Weighting:
Overall, this problem is devastating, it will always exist in the user's scope of use and trouble the user many times. There is no other way for users to overcome this problem. Forcing users to memorize some special letters is very unfeasible, it will not only increase the cognitive load of users but also reduce brand reputation. At the same time, it needs to follow the conventions of the industry and set all letters to be able to search the content of the notes, so I think this is a level 4 issue.

#:	**Problem/Good:**
13	**Good 1**

Name:
Distinguishes between read-only and edit modes

Relevant heuristic:
Error prevention

Evidence of issue:
Read-only can prevent users from accidentally changing the content, and improve the accuracy of users.

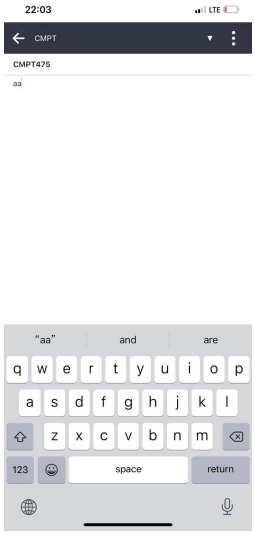


Figure 1: Joplin edit mode page

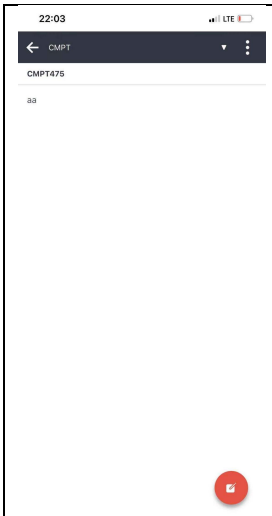


Figure 2: Joplin read-only mode

Detailed explanation:
Distinguishes between read-only and edit, which improves the accuracy of user editing and prevents errors. Avoid accidentally changing the text. Furthermore, it allows users to review the edited content. So it violates the Usability heuristic of "Error prevention ".

Severity or Benefit (minor, major, critical):
4(critical)

Justification:
Frequency:
Every time users want to use Joplin to recording notes, they must use read-only mode to read their notes and prevent editing. When users download the software, they often benefit from the design

Impact:
Read-only mode fundamentally solves the problem of users editing notes by mistake. It protects users from unnecessary errors and saves users a lot of time.

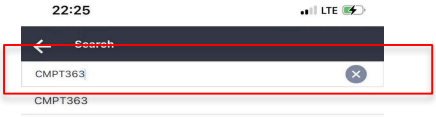
Persistence:
Read-only mode will always benefit the user, and will always help the user edit notes incorrectly.

Weighting:
Read-only mode benefits the user all the time, always helping to prevent the user from modifying the notes while reviewing them. Therefore, it is 4 critical benefits design.

#: 14	**Problem/Good:** Good 2
Name: The search box keeps a record of the last search	
Relevant heuristic: Recognition rather than recall	

Evidence of issue:

The search box keeps a record of the last search. After searching the record will be kept. User can see the record when go back to the searching interface.



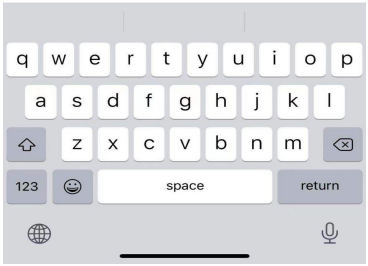


figure 1: Joplin search page. The search box keeps a record of the last search

Detailed explanation:

For example, after searching the note “CMPT363”, and I return to the searching interface, there is a record of “CMPT363” kept in the searching bar. It helps users not have to recall the last search, and save time for users to recall. So it violates the Usability heuristic of "Recognition rather than recall ".

Severity or Benefit (minor, major, critical):

4(critical)

Justification:

Frequency:

Users often use a search function to find notes, and the record is kept for users to next search. The benefits of this design will continue to benefit users

Impact:

Help users do not have to remember the last search, for users to save time. Overcome user recall difficulties.

Persistence:

Record of searching will always benefit the user, in this way, users don’t need to remember the last search.

Weighting:

The ability to record a user's last search saves the user from having to recall notes from their last search. To a certain extent, it helps users reduce the recall time. Users can always benefit from their last search, no need to recall it. Therefore, it is 4 critical.

#:	Problem/Good:
15	Good 3

Name:
There is an obvious confirmation option when deleting notes

Relevant heuristic:
Error prevention

Evidence of issue:
When the user deletes the note, a confirmation dialog box will pop up

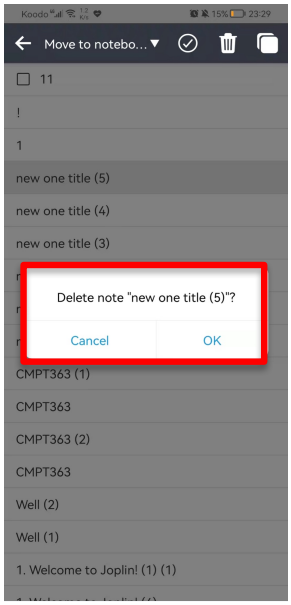


figure 1: Joplin delete note page

Detailed explanation:
When a user attempts to delete a note, the system will pop up a prompt box and ask the user to confirm the operation and the operated object. The user can select the cancel option on the left or confirm option on the right for subsequent operations. So it use the Usability heuristic of "Error prevention".

Severity or Benefit (minor, major, critical):
4(critical)

Justification:
Frequency
This design is often beneficial to the user, as it appears every time the user wants to delete a note and asks the user to confirm that option.
Impact:
The benefits of this design can have a meaningful impact on users, it can prevent users from accidentally deleting files that they do not want to delete. This avoids the time and energy consumed by users when recovering deleted files.
Persistence:
This is not a one-time benefit, even if the user is very careful to operate, there will be error deletion, users can always benefit from this design.
Weighting:
Overall this is a very good design and will prevent users from making mistakes when deleting files. This design prevents high cost mistakes, so I think it is a good design for Level 4