Design/Expert Review

I_SITE

Written by: Erik Bennerhed

Introduction

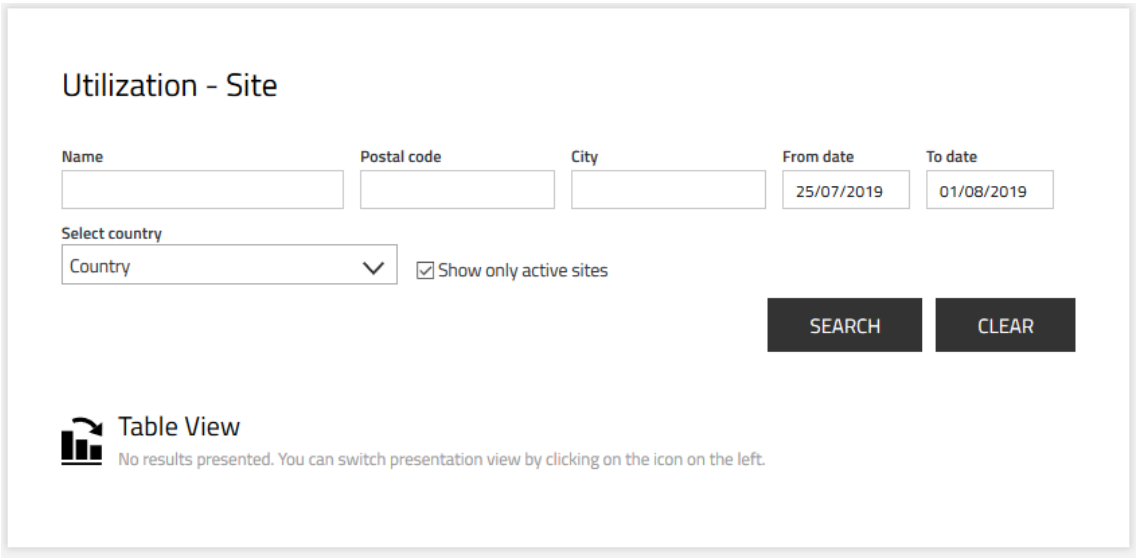
This is a Design/Expert Review performed by the summer intern Erik Bennerhed the summer of 2019. It attempts to connect literature and practices in the industry to I_Site, and how one could view certain parts of the website in the lens of current design practices. As a general approach, Nielsen's (1994) 10 Usability Heuristics as well as the book "About Face" written by Cooper, Reimann, Cronin and Noessel (2014) have been used as the main sources of this review. Out of a review of many parts of the website, 10 themes have been selected for further analysis and discussion of possible improvements. The themes are not ordered in terms of importance, severity or any other way.

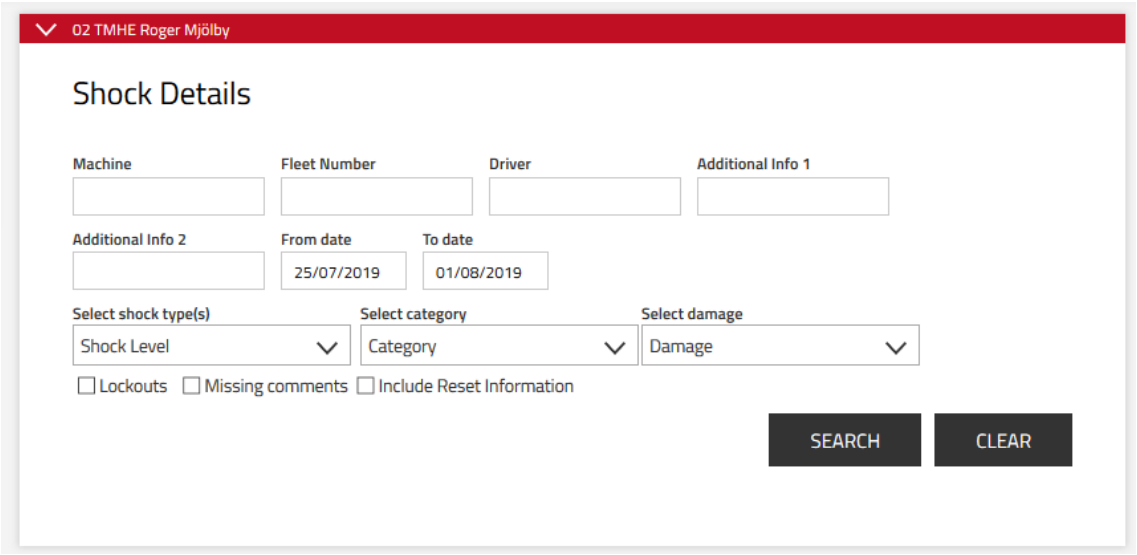
Contents

Recognition rather than recall

Showing users things they can recognize improves usability over needing to recall items from scratch because the extra context helps users retrieve information from memory – Budui (2014)

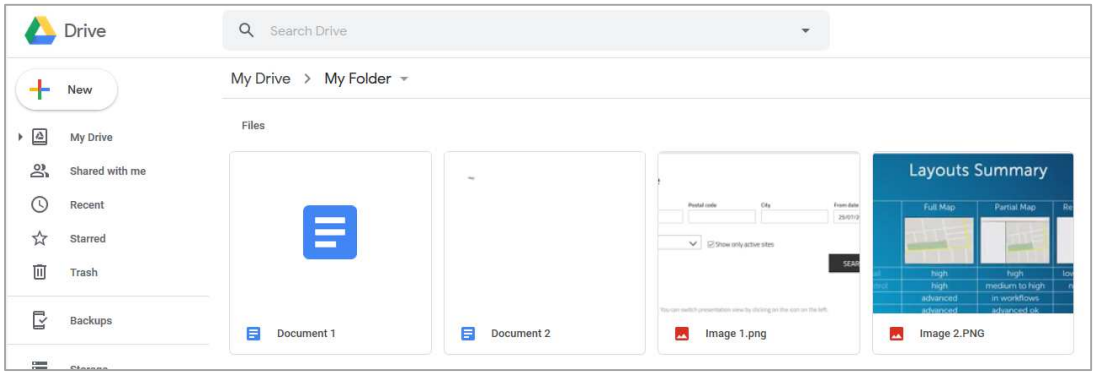
Most of I_Site's functions are introduced with interfaces breaking this heuristic. It presents the user search boxes requiring recall, which puts burden on users, especially new users. Especially, since it can be hard to know what really is being searched. Another issue this creates, together with similar page designs visually across the site, is that users may become confused what they are supposed to search since many pages look similar and that's the only thing they can <nrecognize. In the end, this increases likelihood of ending up feeling confused (navigational trauma). Beneath are two examples.

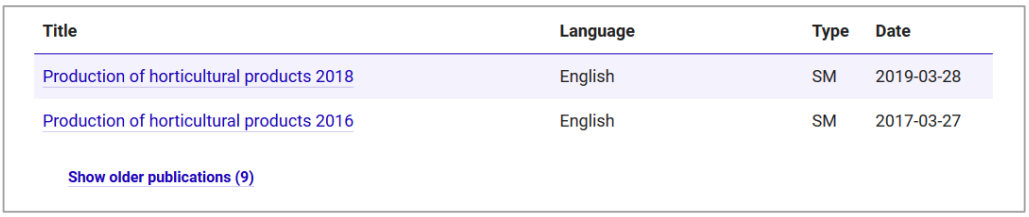




Especially hard is it for the user to deal with "Additional info 1" and "Additional info 2" – this is practically impossible to deal with for new users as they would not only have to recall the item that would be needed to be included in "additional info", but also decide which of the two ones should be filled in. This adds unnecessary complexity to something that doesn't need to be complex.

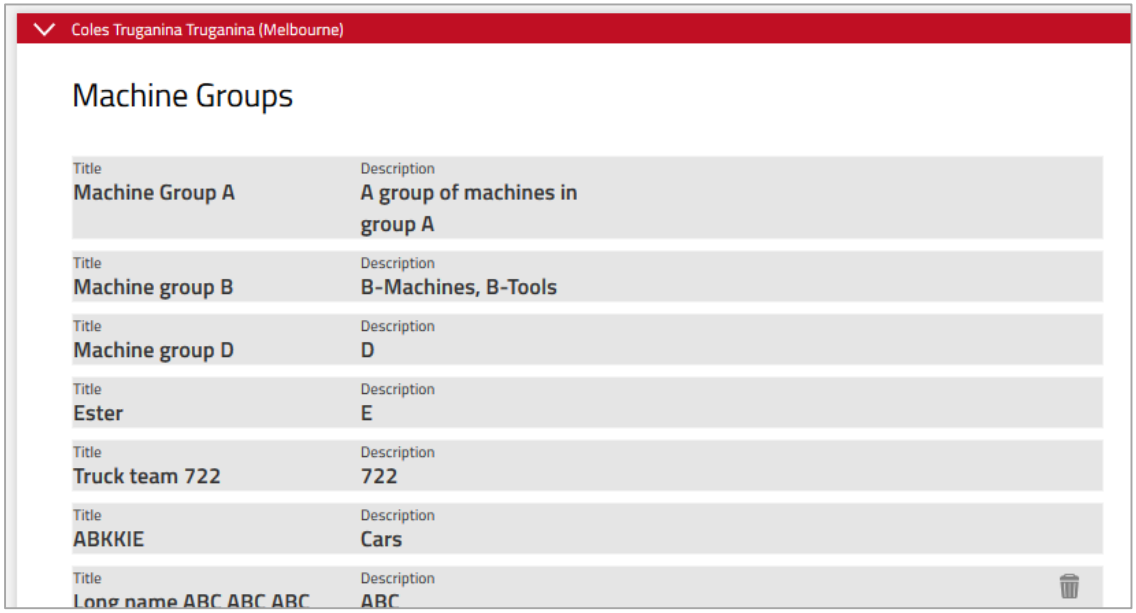
Some sites that have utilized a recognition > recall can be seen underneath. They all utilize a pattern where they present information for you to act on, rather than the user having to request it. On Google Drive, users can see the files uploaded and pick from those without having to recall the exact names of the files per se. Similarly, most sites provide easy access to their services by simply naming them explicitly for the user to see, rather than having the user request their products from memory.





SCB lets users browse their categories with a "drill-down" pattern, where each drill down a) notifies the user about the context he or she is in, and b) takes the user closer to his or her goal.

Actually, there are some examples on I_Site which could be said to follow this pattern. Under Manage Site -> Machine Groups, for instance, users are presented a list of existing machine groups. This way, the user does not have to recall the name of a machine group to find it, but can instead just browse the list of existing groups – and filter them by site if that would be needed.



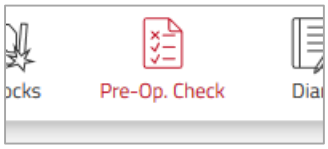
Be careful with jargon and abbreviations

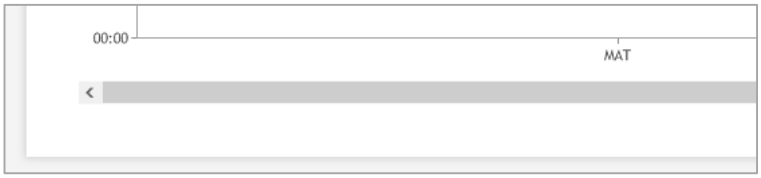
Digital products are also often obscure, hiding meaning, intentions, and actions from users. Applications often express themselves in incomprehensible jargon that cannot be fathomed by normal users (“What is your SSID?”) and are sometimes incomprehensible even to experts (“Please specify IRQ.”). – Cooper et al. (2014)

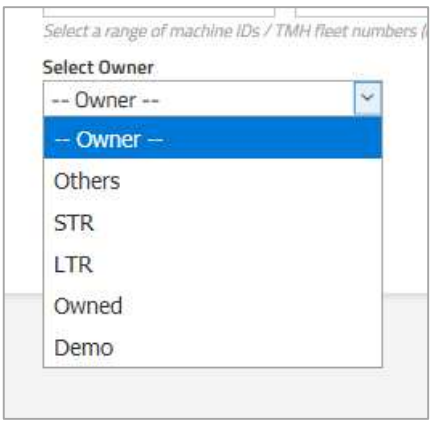
We should stay away from jargon, because our menu’s users won’t yet be acquainted with it. – Cooper et al. (2014)

While not by itself “bad” or necessarily wrong, jargon and abbreviations pose a risk of hiding content to those not familiar with the language. One should make sure that the user(s) of the system really understands the language and abbreviations if one decide to include them in the software. On I_Site, there are a number of especially abbreviations that at least for me (The reviewer) was hard to understand. However, one should not that I’m a total novice in the area.

Beneath are some examples.







Sometimes, one may be in a situation where an abbreviation is needed or may be argued for, in those cases, at least offering the opportunity to investigate the abbreviation should be offered. This could many times easily be accounted for by allowing a hover-state tooltip that spells out the whole abbreviation for the user. Flightradar24 lets the user hover over the name of an airport code to see the destination fully spelled out. This way, users who are not familiar with the “MUC” code can easily learn where it is. (See beneath)

Standards

Think about standards as grammar in a language. They are the corner stones for communication and are for us to make sense of the things we present. In many ways, language and design are similar, and just like how you would make sure to often abide grammatical principles (To make sure your interlocutor understands), you should try to follow design standards as it will be easier for the other part (the user) to recognize what is going on. Standards…

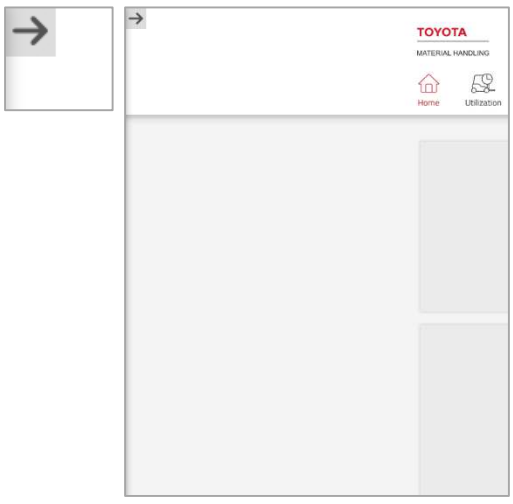
> …Improves users' ability to quickly learn interfaces and enhances their productivity by raising throughput and reducing errors. – Cooper et al (2014)

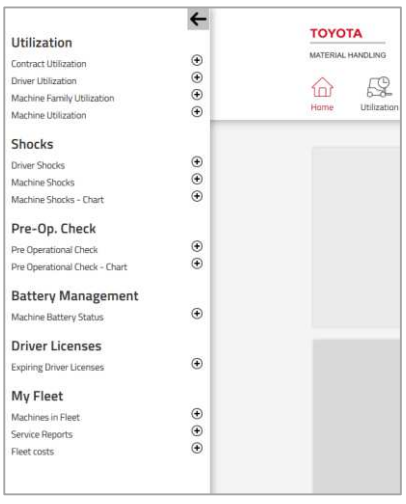
It may not even be a bad idea to not abide by platform standards, but it is very risky if you do it without much thought. Many times, there are already existing UI patterns that could cover what you are trying to achieve. Using these more common patterns would make learning easier for new users. These more common UI patterns have often been tested more, so they are likely to often fulfil basic usability needs, even though they may still have some issues as well. Therefore, observing how others have solved similar situations (Or following design standards) can often be helpful to make your product predictable and easy to adjust to. You can violate design guidelines if you come up with a significantly better alternative – which has been tested and validated using formal methods. Most of the times, you would do better and save time by just staying with standard practices.

> "Obey standards unless there is a truly superior alternative."
> – Cooper et al. (2014) Design principle 17.5

With that said,

On I_Site, there are some unconventional components that took a while to get used to, or even to notice at first. Two of the more easy ones are examples of icon usage which may confuse some people.



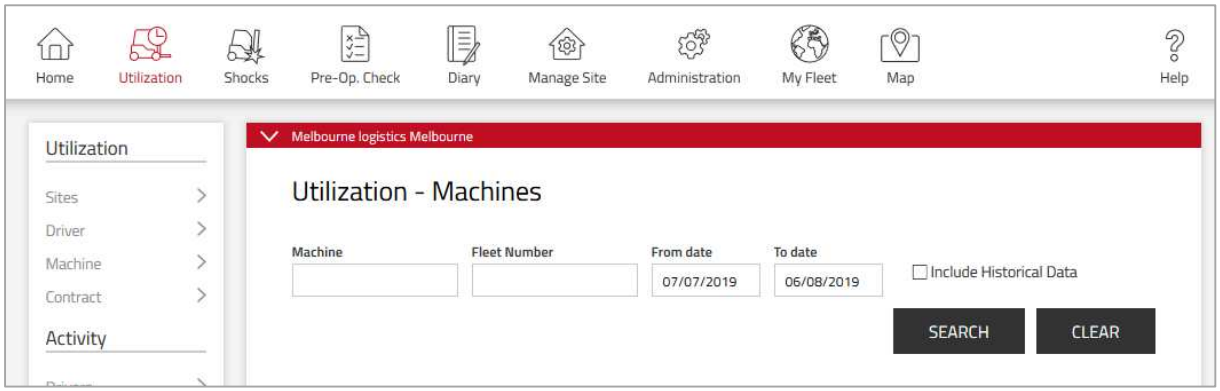


The arrow icon to expand the hamburger menu has a dedicated icon among platforms and look like three horizontal lines on a row. See Gmail.com and Koss.com for example.

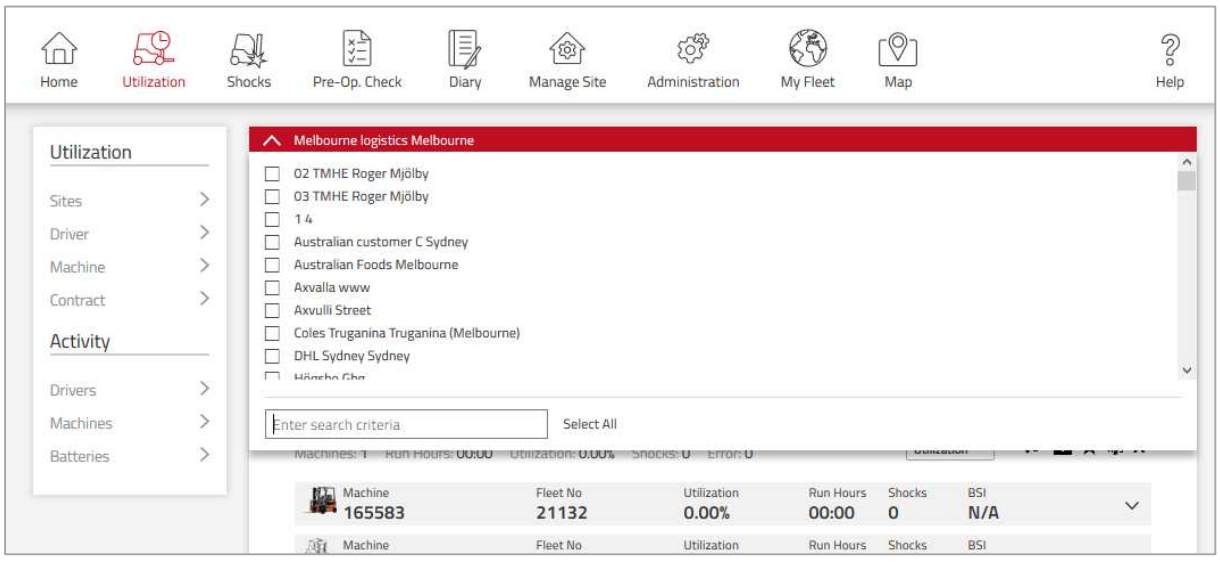
Another icon which may be discussed is the use of a +-icon for add. It is used in the hamburger menu to speak of, and adds a target component to the home page dashboard. This may not be very clear for the user as this is not very verbose, and may especially in this case be confused for being expand icons. It appears to be quite common to use plus icons in accordion UI components (Friedman, 2017).

Simply a text-button such as "Add to dashboard" could alleviate such a confusion, or a redesign of the overall structure of the site (Requires more work).

Another unconventional solution is the site selection menu found on many places on the website. See below.

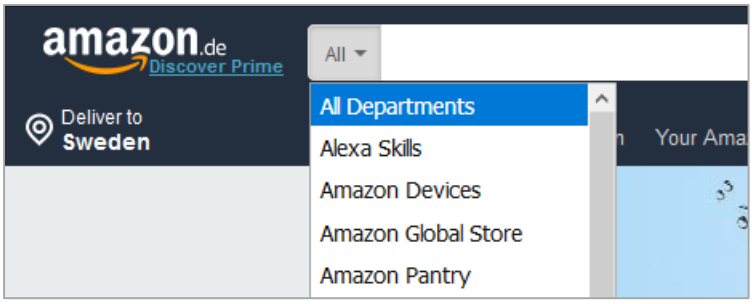


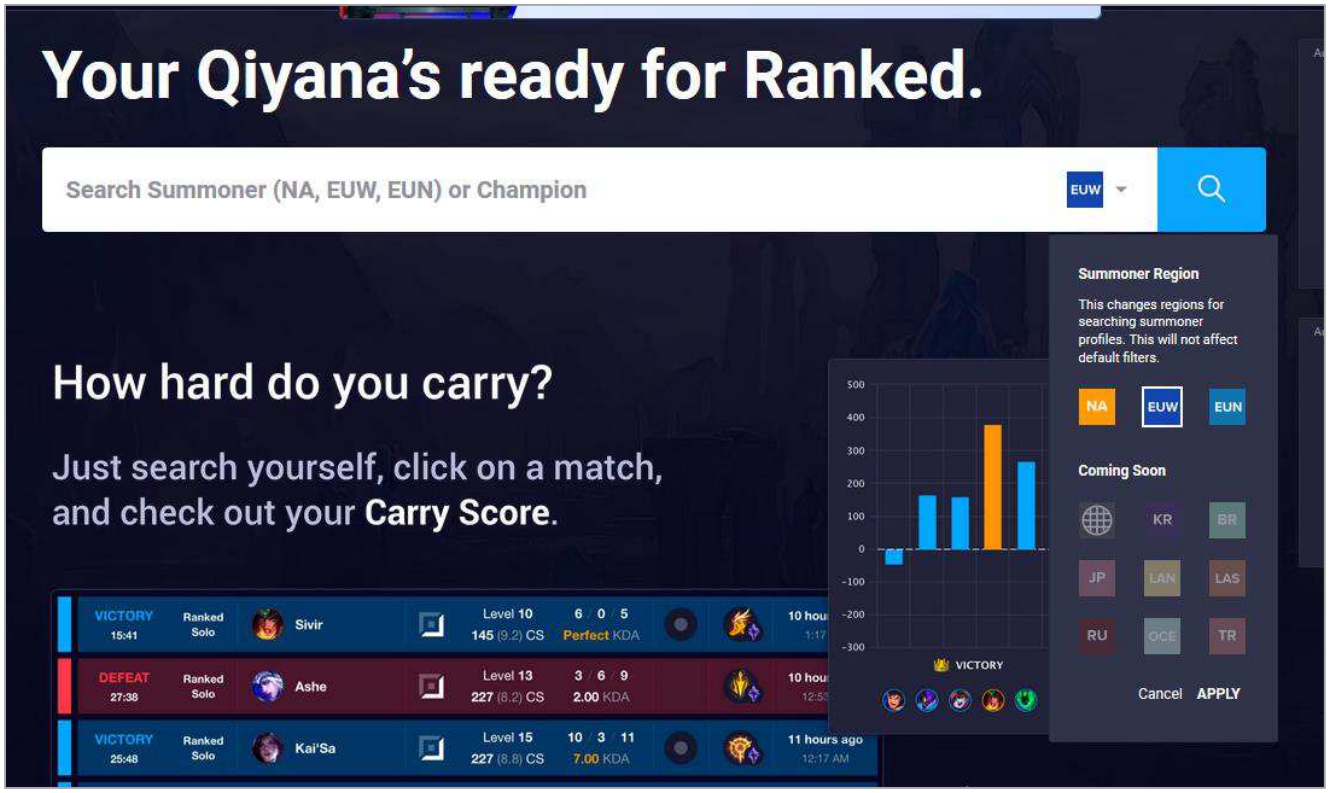
While the component is easy to find (It is red after all), it may not not stand out as being a drop-down where you can filter sites (See below). My initial thought was that it was some kind of subheader, or even just a divider. Two reasons for this may be that it is unusually broad for being a drop-down, and it has no pliancy/affordance that tells the user it is clickable. The only hint that might give a clue is the use of the arrow to the left.



So, how could one do this differently? Well, judging the overall structure of the site as well as the intended goal of making it possible for the user to sort the result to include only specified sites – perhaps something more unconventional was needed. The general disposition of controls and data could probably have been very different, which would have completely changed the core structure of the site. So, adding a somewhat more uncommon component could perhaps be argued favorably in this case, but in doing this, one should be very careful with one's execution and test it properly.

Some sites such as Amazon and U.GG (See below) have combined search entries with combo icon buttons or menus where you can filter your search. While what they are doing is not quite the same as what I_Site attempts to do, one could use these examples as inspiration of how I_Site could work. It does not mean a similar replica would work on I_Site, perhaps it will even be the contrary – however it remains clear that a good solution to this matter would need proper and thorough ideation and evaluation to be sure that things go well together.

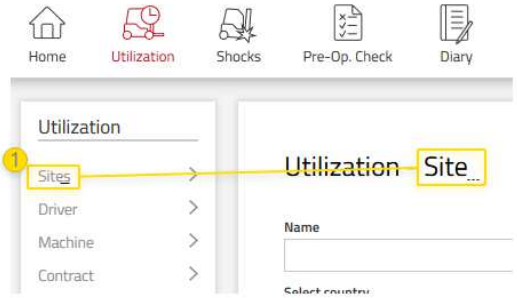


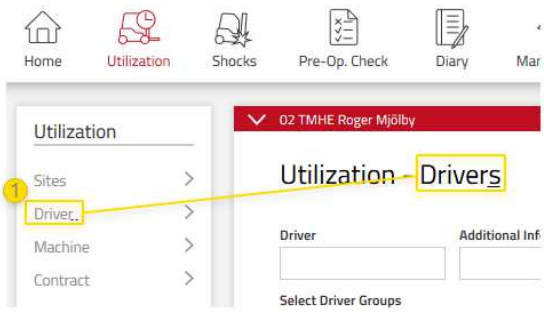


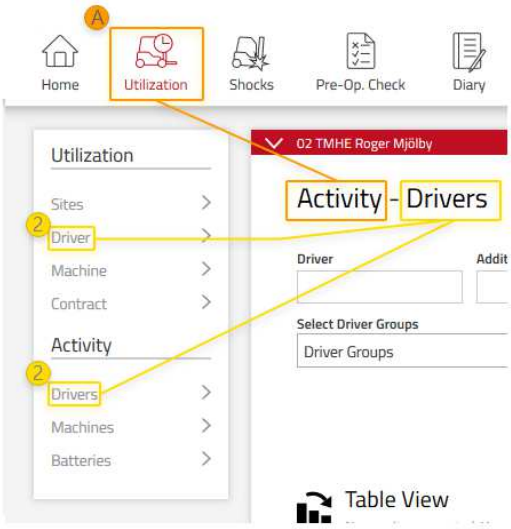
Consistency

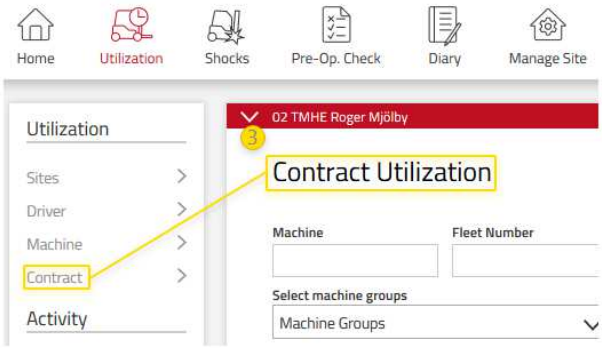
This may not even be applied to design only but to pretty much everything. If you want to be clear, you need to be consistent. Even small deviations can cause confusion. So, while it may feel like a nitty-picky detail, it is actually very important to check for inconsistencies.

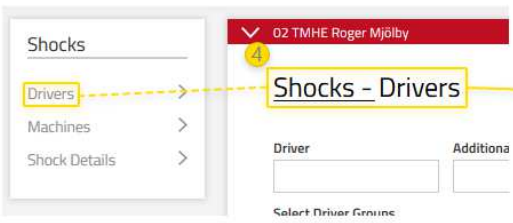
There are a couple of cases on I_Site where things such as descriptions, headers and other elements that correspond to each other have slightly different headings. It also appears that the headings of each section/page of I_Site does not follow a consistent pattern. This may not be a game-breaking issue by itself, but is enough to cause friction for the user (Minor details in the program that eventually could cause irritation). Examples and explanations beneath.

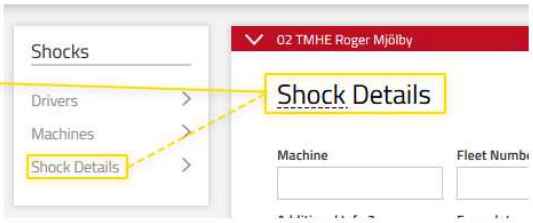












In the case of (1), it is a rather minor detail that would by itself not really pose a big risk of being a usability problem. However, coupled with (2) duplicate item names under the side panel (Such as “Drivers” under both Utilization and Activity) and an absence of a position mark in the menu

(For more, See *Visibility of system status*), it can easily become confusing for the user, especially since the pages (Utilization – Drivers & Activity – Drivers) look quite similar visually. This situation is partly caused by (A) the hierarchy, which puts "Activity" in the side menu under the same level as "Utilization", although the king header of that section has the name "Utilization" which creates a hierarchical conflict. (For more, see *Hierarchy & Structure*). You can also find instances of "Drivers" and ""Machines" under Shocks, Pre-Op. Check, Manage Site, Adminisrtration and My Fleet.

(3) and (4) shows other examples where the page heading seems to follow an inconsistent manner.

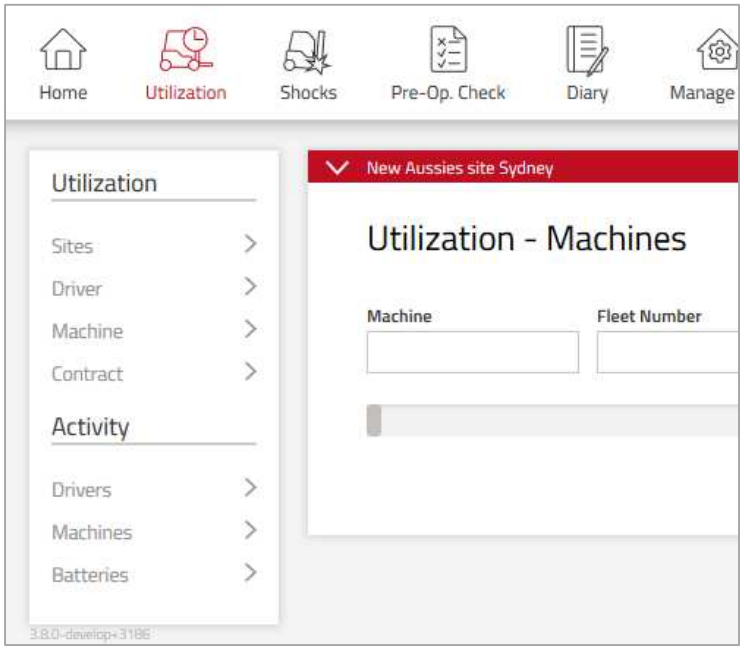
Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time. – Nielsen (1994), Heuristic 1

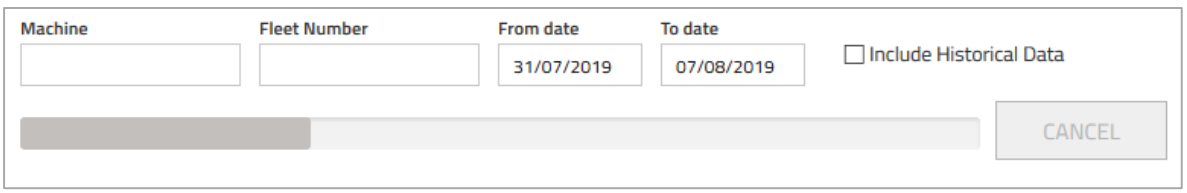
Communicating the current state allows users to feel in control of the system, take appropriate actions to reach their goal, and ultimately trust the brand – Harley (2018)

Users can get stressed if they feel they are not in control, and loss of control can be for many reasons – one of them is the visibility of a system's status. In other words, a user wants to understand what is going on, preferably at all times. This way, he or she can follow why the system acts as it does. It mustn't only be about what is going on actively, but also where you are in the system.

One of these situations, where the system does not fully notify the user where she or he on the website, is on the second-level hierarchy which is the left menu (See below). The site does not mark on the menu which page is currently selected. It does, however, mark which tab is selected above. This could easily be dealt with by e.g. coloring the chosen section with another color than the ordinary one.



Depending on where you go on the website, the system may sometimes start "loading" something, which is cool because it shows it is working with something. However, it would have been even better if it showed *why* it is working with something right now. For example, it could say "Fetching data" or "Generating graph" instead of being blank. This is especially important if it comes automatically as is done in Utilization – Driver, as a blank loading bar may raise as many questions as it is trying to solve. But also for situations where the state is more actively generated (As in clicking a button), it could be a good idea to properly spell out what is being done.



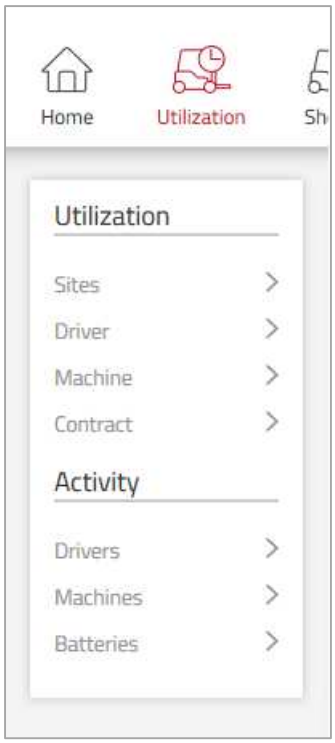
Hierarchy & Structure

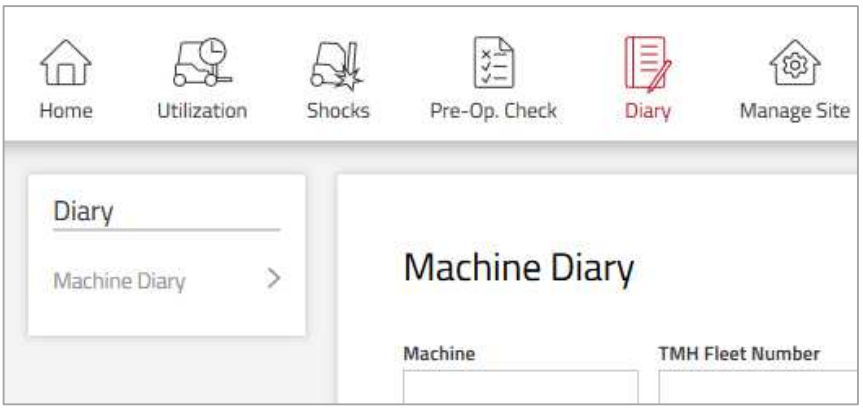
Hierarchy is a core component of anything substantial. It can be anything from how things are ordered and what is above what, but also what comes after what and when it comes in the order.

"Less is more" is a common saying, indicating that minimalism can give you "more" meaningful good than unnecessary complexity. If one takes a look at I_Site, several items appear by their own numerous times over different pages. Many times, the users may still end up in "Machine Configuration" which is a sort of "final destination" for many interaction paths. More content requires more processing power. More processing power is demanding. For example, since several paths lead to the same door, learning may be prolonged as one would need to process and make sense of all the paths in relation to each other.

Perhaps, the underlying issue comes from the overall structure of the site as a whole. When the same items appear as a subsection several times, it may be a hint that another structure would be more fitting.

One hierarchical problem can be found under the "Utilization" tab where utilization is both the root of the tab but also one of the two subsections (Utilization & Activity). This gives conflicting hierarchical signals to the user, which may be confusing. Another example can be seen to the right beneath, where the tab only has one subsection with one sub-sub-section. In this case, the menu could be omitted as it just adds visual clutter which could take away focus from the main task.

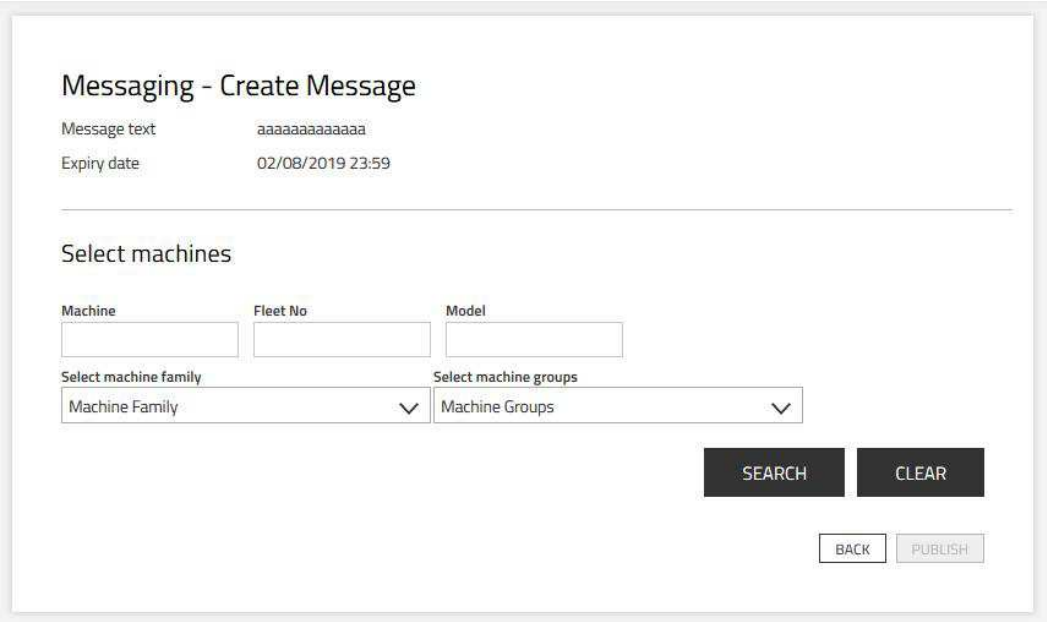




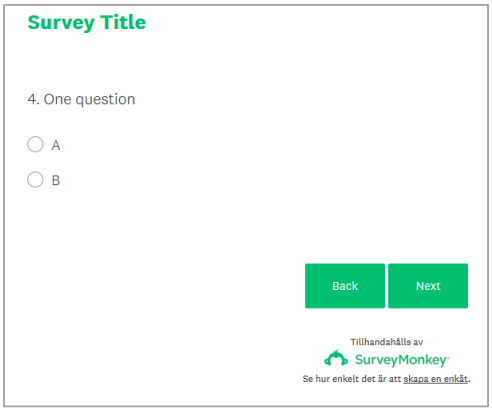
Emergency exits

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 and extended dialogue. Support undo and redo. – Nielsen (1994), Heuristic 3.

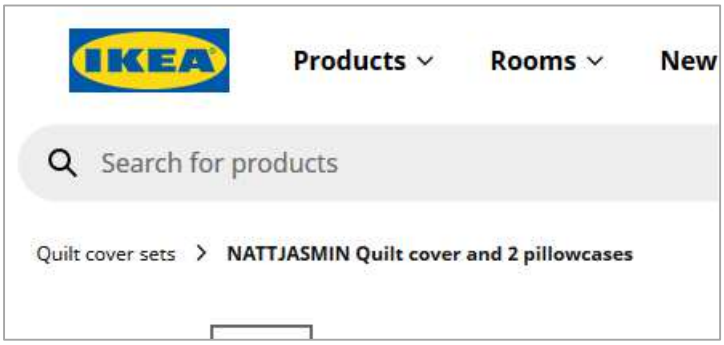
Though there are some examples with escapes/undoes, there are times when the users may be stuck without a clear escape, especially between pages. Sure, one may use the built-in back function in the browser, and according to some, that might be fine. But a visually appearing back function/escape does more than just taking back the user to the previous position, it also works as an assurance that you are on track, and a trace to lead you back if you would want to/ if you get lost. One of the few examples of a "Back" can be found under "Create Message" after having entered your message text. At the bottom-right of the component, there is a "Back" button, however it is both much smaller and less prominent than the two black buttons above, making it prone to being missed.

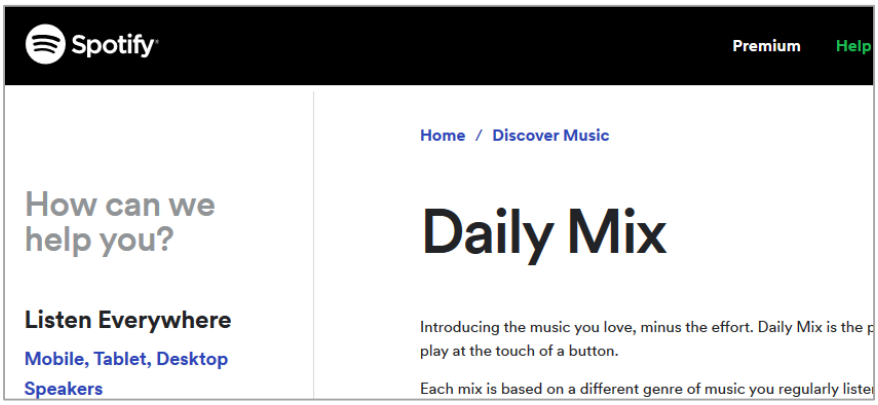


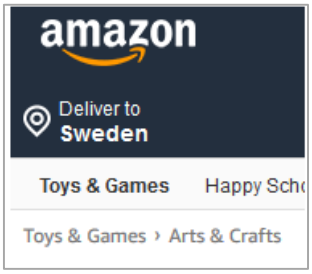
While it is good to have a back here, it is very anonymous in its appearance and lose much of its prominence due to the both bigger and more visually dominant buttons above. Many web services have clear "Back" and "Next" (If it's needed) buttons, often discernable by a margin or by a divider.



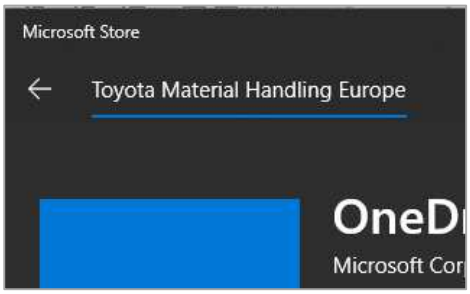
However, when browsing between many different functions or pages, many websites utilize breadcrumbs which lets users see the path they are in. These breadcrumbs often are clickable, so they can easily be used to navigate backwards. Breadcrumbs are very cost-efficient for what they give, as they take little space but offer much to the user.

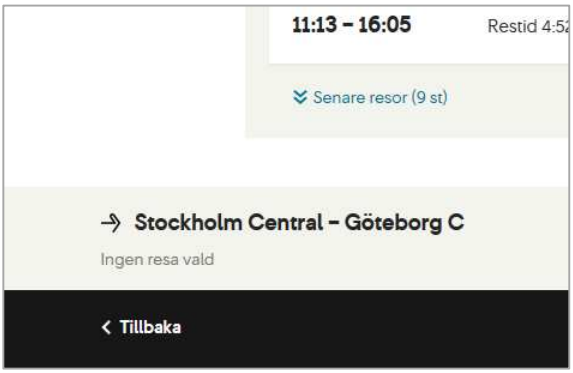






Another solution, which seems to be more used on more transient/simple apps, is to have a Back functionality at the edge of the application. This pattern is seemingly uncommon among web pages. Beneath is an example on Microsoft Store where an arrow can be used to go to the previous state. One example which is *similar* but not the same is on SJ's webpage where they have added a "Back" on the web page's footer when the user is ordering a ticket. This one exists only in the order-a-ticket mode. In a sense, ordering a ticked is a type of small task, requiring more transient interface standards. Still it could be used as inspiration in a future development of I_Site, where depending on the task, different escape controls could be used.





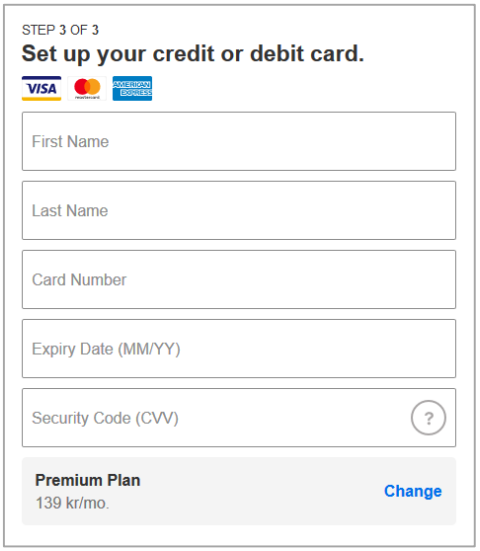
Input hints

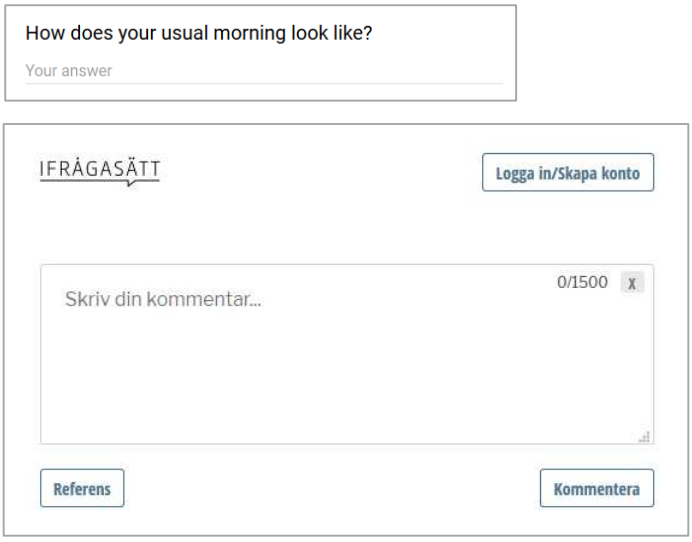
An input hint helps users establish what information should be entered into an input field (Esser, 2018). It helps them quicker set the context. Contrary to popular belief, entering data to text boxes can be quite a demanding task, as the user has to repeatedly recall information. Similarly to *recognition rather than recall*, giving some material to the users they can attach their thoughts to may speed up the process of setting context.

Right now on I_Site, there are no input hints on entry boxes, which is not fatal but a little help on the go is always appreciated.

For example, we have the term "driver" being used repeatedly on the site. A new user may be unsure whether or not this means "driver" as in someone driving a vehicle, or a "driver" as in a device or part of a device. If one want to speed things up – adding a hinting text could speed up the process as it helps set the contex. Important note here is that the hint should disappear as soon as the user starts typing something – as keeping the hint could make the user confused as it could appear that the text would be included in the search.

Input hints are very useful as a "primer" for many other cases as long as they are not too dominant in presense. It does not have to be an ambigious situation as above. Input hints are commonly found on many websites, sometimes it may be all that is needed (i.e. no header is needed, only the input hint).





Avoid blank slates

"Avoid blank slates" is one of Cooper et al.'s (2014) points in making "harmonious interactions". In a nutshell, it means that rather than "assuming nothing" and leaving something empty, it is better to make an estimated guess in what the user might want.

This is a term that does not seem to be used very broadly within UI design, but the message can still be interesting as a general term. Figuratively, a blank slate is equal to a fully cleaned kitchen where all kitchen appliances have been stoved away. This is nice and can often be preferred. However, imagine if a kitchen-system knew you were going to make lasanga in the evening, and could prepare all necessary tools so that they are ready to be used when you come home after work.

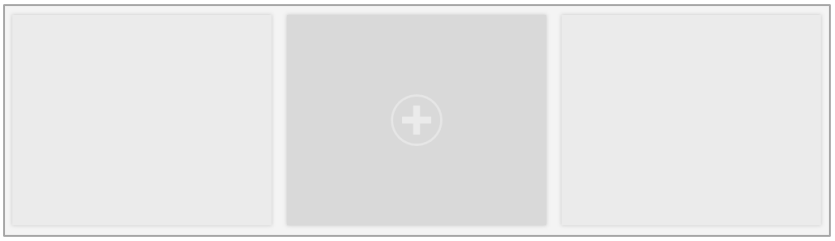
Likewise, the idea of a digital system could follow a similar pattern. If you could make a system be smart and prepare things that the user is likely to prefer, it can significantly make things easier to deal with. It is easy to be afraid and assume nothing of the user, but many times normal users, and especially new users, would feel much more comfortable if the system makes an approximate guess what it thinks is right and then manipulate that to make it exactly right – rather than leaving everything empty. The effect is that the system may help the user to quicker reach his or her goal – or be in a state which would be the same as in doing nothing.

Fully predicting what the user might want doesn't even have to be very hard, altough it can sometimes be. There are some easy examples that could be used for making based guessed of what the users might want. One is to save the users' last input and present that the next time the user enters the application. It makes sense to think that what one user did one time is what the user would want the next time she or he uses the application. This, of course, may not always be true but it may just as well happen to be. Other ways are possible of which some fit some scenarios better than others.

"For most users, a blank slate is a difficult starting point. It's much easier to begin where someone else has left off." – Cooper et al. (2014)

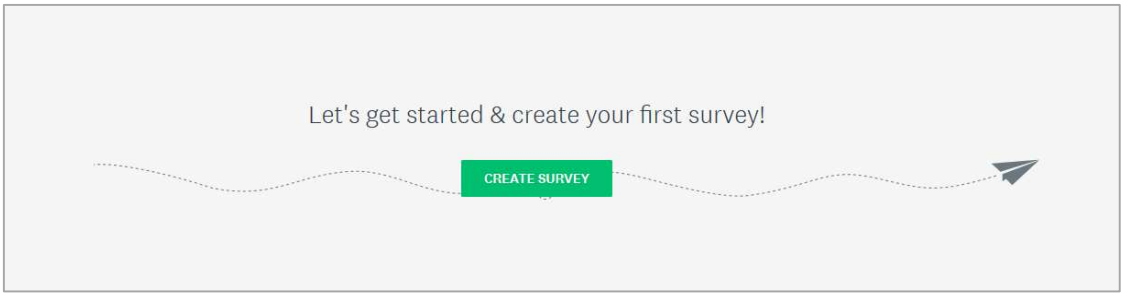
I would like to elaborate a bit more on this expression coined by Cooper. Cooper mainly discuss this term in relation to entry controllers, but I think one could use the saying on a more general basis. Mostly, what it is intended to say is that something that is empty requires more effort to fill in than when there already is something there to work with. This, however, does not only have to be entry controllers – but could also be more general things such as content areas. Leaving an area intended for interaction blank without any hints, makes it harder for users to pick it up. One such example can be found on the home page on a first login on I_Site. As can be seen under, though the user is presented a grid, its content is empty and does not really imply anything.

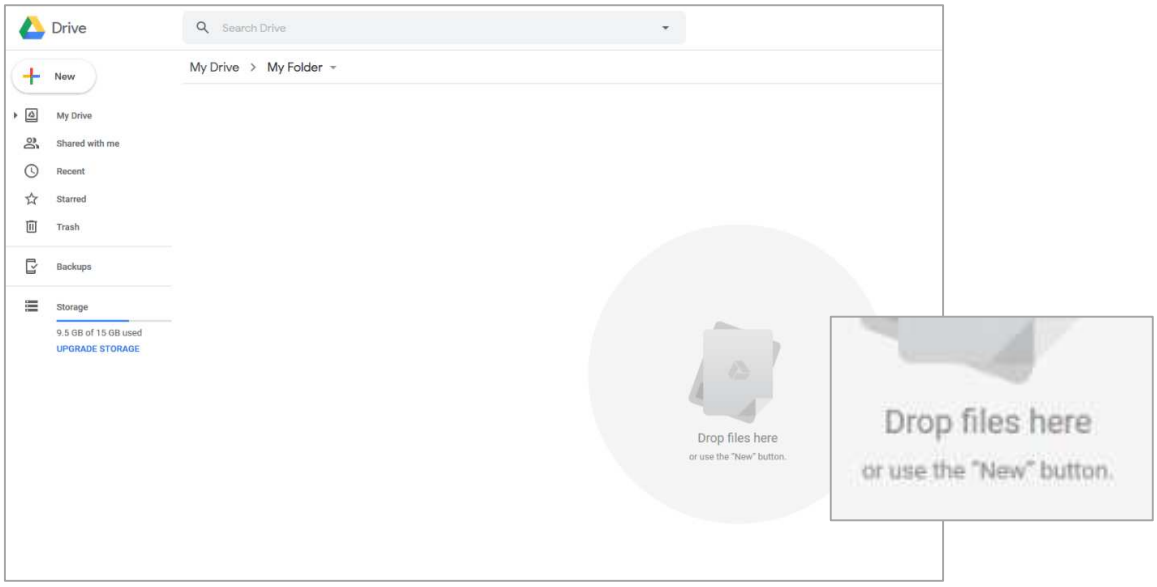
While filling in this dashboard with pre-defined boxes may help laying out a disposition, it does not brief the user what it is and what it is intended to be used for. All in all, they are just placeholders for something _blank_. Even if one would hover over the area, it is hard to estimate what it really means as nothing is projected ahead of what it *could* be.



With this in mind, it would be easy to take a look at Cooper's term about blank slates and try to fill the dashboard with something that ought to be relevant. But the special thing with I_Site is that it cannot really fill a dashboard with relevant content as the nature of the site is quite individual and based on what each and every user decides to implement and monitor. Similarly, sites such as Google Drive cannot put in fake files, and Surveymonkey.com cannot really create dummy surveys. This means that you cannot really "fill" the slates with actual content, instead you would have to come up with something else.

Rather than leaving this intended area completely blank, Google Drive and Surveymonkey (See below) have resorted to something that could be compared to an *input hint*, though one might call it call-to-action or something else. Namely, they utilize the area with a component that either directly calls for action, which by its presence, also tells the user what this area is intended for, or utilize the area to subtly hint how it could be used. The exact contours of the solutions may differ, but the purpose of them are the same: to introduce the purpose of the content area by inviting the user to take part in manipulating its content. I_Site could use this chance on the home page, and implement a similar pattern for new users rather than blank placeholder boxes. The sites may not be exactly the same, but one could still use them as inspiration for I_Site.



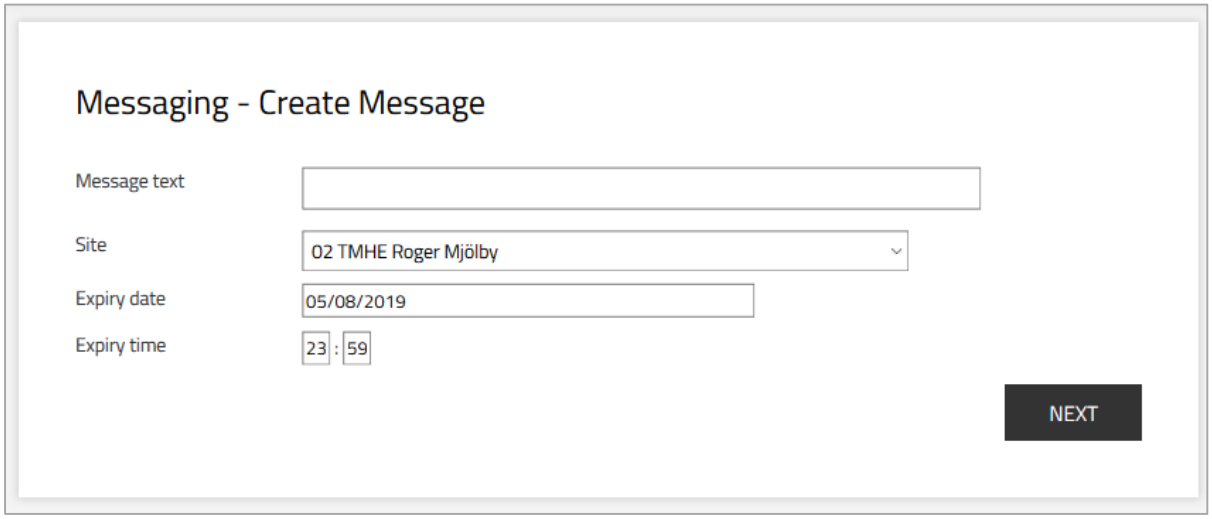


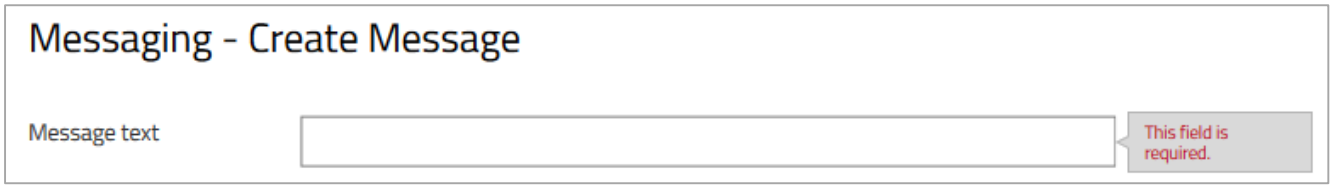
Error prevention

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.

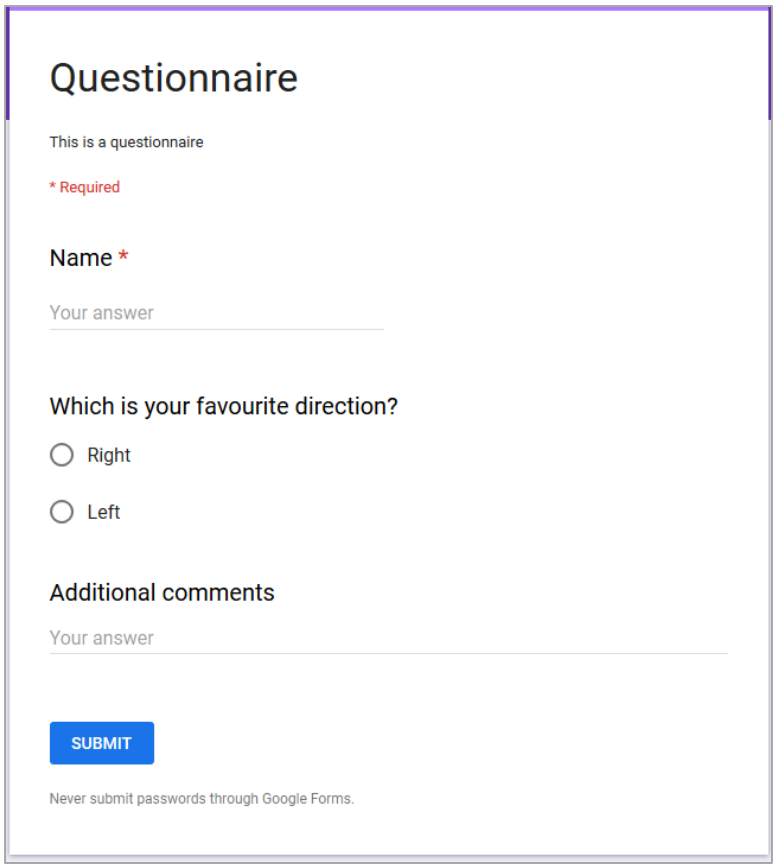
– Nielsen (1994), Heuristic 5.

Making a mistake is both stressful and breaks the flow. On I_Site, there are two examples where an error message is sent on situations which could have been prevented. Both of them are related to the topic of data entry. The first one is to that of required entry boxes, where and input is required to go to the next phase. Right now, this requirement is not hinted at the entry of the page, rather the user has to make the mistake of not giving any input before he or she is notified. For examples, see under.

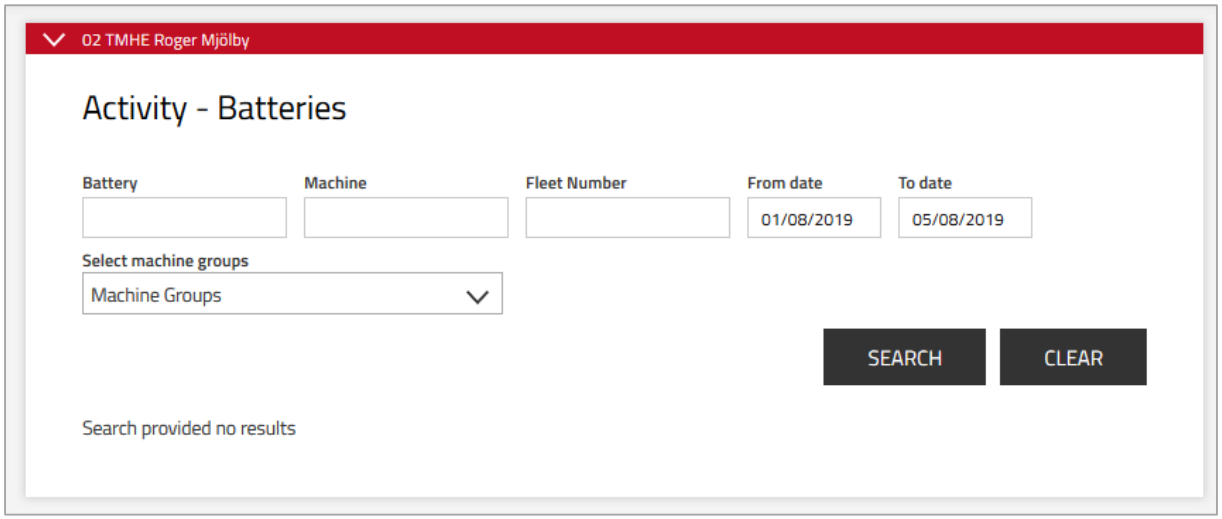


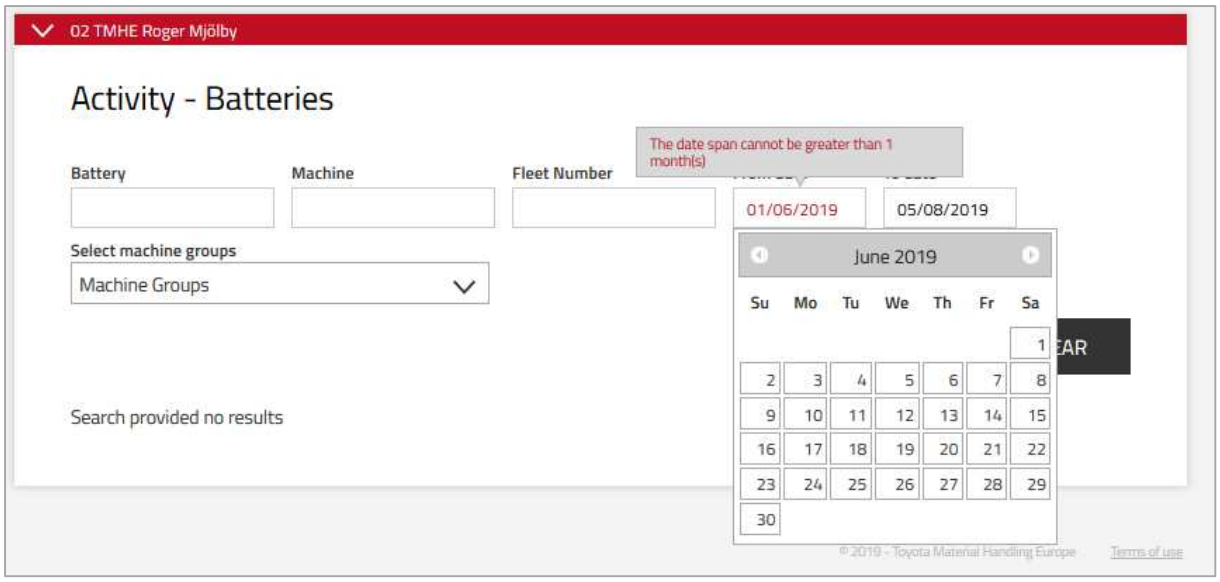


This is a minor obstacle, as it is easily overcome and one could argue it was a mistake by the user at first place. Still, it adds some frustration to the user which may overtime result in negative overall experiences. Better would be to prevent this from happening at all. A common way to hint required input in forms like these is to use a "* Required" tag where a star-tag (*) is put above the form, and stars are added to each required component to indicate that they are needed. See under for an example on Google Forms.



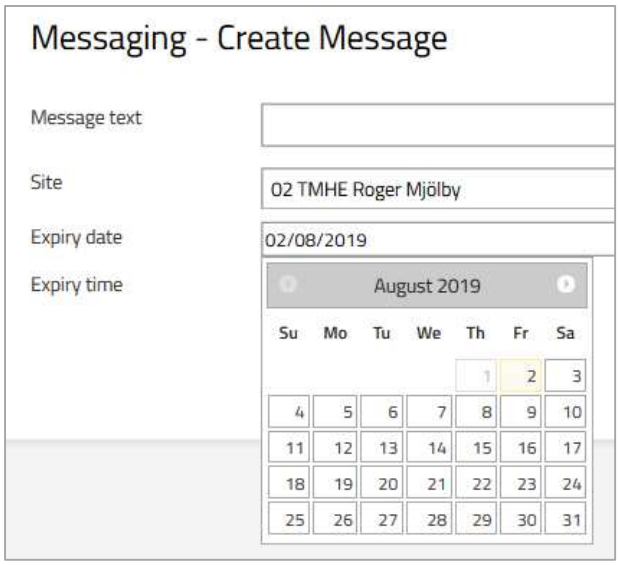
Another issue can be found under "Activity - Batteries" where you cannot have a date span over 1 month. Again, this requirement is not hinted, so it is possible that a user attempts a longer date span than what is possible. See the two states under.





As described above: this is a situation that may not be fatal to the activity undergoing, as the user may simply re-arrange the dates so it fits the requirements. However, this is again something that would add significantly less friction if the state of having done a "mistake" could be prohibited to begin with.

This time, it is a bit more complex as we are dealing with a more special problem. In a way, a solution already exist. One partial solution to this problem already seem to have an implementation on I_Site, which is when you are creating a message. Here, you cannot pick an expiry date before the present day (See under). This way, users cannot make the "mistake" of picking an invalid date, as the date is not possible to pick.



A similar solution, but where you make the threshold instead be -30 days from the present, could have been a good solution for what we are dealing with. However, that would make it impossible to pick an increment of 30 days that is say, 8 months ago. There may be many ways to deal with this. One possibility could be to simply let users pick any date they wish on the "from date", then change the "to date" to 30 days after that date – and allow for a ±30 days span on "to date". This is however only a spontaneous idea, so it would have to be elaborated a bit more to see if it makes sense.

References

Budui, R. (2014, July 6) Memory Recognition and Recall in User Interfaces. Retrieved from https://www.nngroup.com/articles/recognition-and-recall/

Cooper, A., Reimann, R., Cronin, D., Noessel, C. (2014). *About Face*. Indianapolis, IN: John Wiley & Sons, Inc.

Esser, P. (2018) Support Users with Small Clues in the Input Hints Design Pattern. Retrieved from https://www.interaction-design.org/literature/article/support-users-with-small-clues-in-the-input-hints-design-pattern

Friedman, V. (2017, June 21) Designing the Perfect Accordion. Retrieved from https://www.smashingmagazine.com/2017/06/designing-perfect-accordion-checklist/

Harley, A. (2018, June 3) Visibility of System Status (Usability Heuristic #1). Retrieved from https://www.nngroup.com/articles/visibility-system-status/

Nielsen, J. (1994, April 24) 10 Usability Heuristics for User Interface Design. Retrieved from https://www.nngroup.com/articles/ten-usability-heuristics/