Guidelines for productivity tools and data entry UIs

ABOUT THESE GUIDELINES

If you can answer **yes** to each of the items in the checklists, it is likely that what you made supports the work users do.

Many of the items covered in these guidelines are covered in books on interaction design. The best one is Microinteractions by Dan Saffer, published by O'Reilly.

1. **Keep the user in one place to complete the task:** Minimize mouse movements, enable keyboard navigation

Keep the amount of movement that users have to make to enter or check data comfortable and tight. When users have to move the mouse, this slows them down. Implement tabbing from field to field. Enable navigation using only the keyboard.

CAN USERS:

- ☐ *Use the interface efficiently, with minimal movement of their hands?*
- ☐ *Work the interface without using the mouse at all?*
- ☐ *Navigate without expanding or collapsing sections?*

2. Support a Save–Next automatic workflow

Don’t make clerks go back to a task queue to get work. When they have disposed of one item, bring up the next.

CAN USERS:

- ☐ *Keep working from task to task to task without having to go back to a queue to get work?*
- ☐ *See that the work they have completed has moved forward?*
- ☐ *See how much more work they have in front of them?*

3. Automate as much as possible

Use the power of the computer to assist users. Derive outcomes and changes in state from work already done, either by the system or by the user.

For example, if all of the items in a list have been checked off, we can safely say that task is complete. Don't make users explicitly click a checkbox to say they did it.

Parse anything possible so users don't have to focus on data formatting.

CAN USERS:

- ☐ *Flow through their work without having to tell the system what they've done?*

4. Minimize button clicks and confirmations

Every time the user has to leave the data entry fields, it slows them down, so implement caching and auto-saving whenever possible.

Allow for batching up records to be confirmed and committed. Run edit checks on each field to ensure that the number of digits are correct, and if appropriate, that the format is also correct.

CAN USERS:

- ☐ *Focus on data entry rather than what the app is doing?*
- ☐ *See that their changes have been saved by the system without modals or popups?*

5. Minimize contextual noise: include only necessary content and fields

Tools should be spare and clean. Anything extra is distracting and slows users down. Learn from users what the minimum information is that they need to do this particular job, and only build that functionality. Remove anything that does not help them do the work.

HAVE YOU:

- ☐ *Minimized borders, shading, gradients, and shadows?*
- ☐ *Avoided headers?*
- ☐ *Avoided fancy animations that don't convey meaning?*
- ☐ *Removed, avoided, or automated table paging?*
- ☐ *Asked a real user exactly what fields she needs to do the task?*
- ☐ *Labeled fields and buttons in language the end-user uses to talk about the task?*

6. Avoid table layouts whenever possible

Don’t just drop the database into the user interface. A table is not always the appropriate way to represent the data or to arrange the interaction. When appropriate, create visualizations that use the power of the computer to show users what they need to know at a glance.

Learn what the user needs to do the best job possible on the task he’s doing. Automate table navigation options to appear when needed (more than 10 records) and not before.

CAN USERS:

- ☐ *Tell at a glance what the data is telling them?*
- ☐ *Quickly see and easily understand the data?*

7. Show only the data needed, in human form and language

Just because it's in the database doesn't mean you need to put it in the UI. Learn the task the user is trying to do and ask what might be extra and what might be missing. Don't just automatically display all of the data in the related data model in the user's view.

Use plain language labels and names that describe the data in a way that the user won't have to guess or make inferences about what you're showing her.

CAN USERS:

- ☐ *Tell what is most important?*
- ☐ *Quickly see and easily understand the data?*
- ☐ *Review data without having to ignore anything?*
- ☐ *Find all the data for a task in close proximity?*

8. Automate the work, not the form

The point of digital transformation is not to put forms online, but to help users do their best work and make the best decisions to ensure that the right people get the right benefit at the right time.

Avoid duplicating the paper form in the user interface, especially when the computer can do some of the work. Exploit the power of computing to collect data and validate data and states along the way. For example, if an applicant's case fails the approval criteria, and you can tell based on interactions the user has already taken in the system, recommend continuation or denial, with the option to override the automatic recommendation.

HAVE YOU:

- ☐ *Understood the users' needs?*
- ☐ *Learned the context for the story you're developing?*
- ☐ *Saved the user from tedious work (like calculating date ranges)?*

9. When in doubt, get feedback from users in usability study sessions

DO THIS ON MOCKUPS EARLY IN THE SPRINT:

- ☐ Get users in a screen sharing session where you can also talk to them on the phone.
- ☐ Give a little background, including a sentence or two about your intent.
- ☐ Let them look around a bit, and ask questions. (Note what the questions are, because those are probably things you need to clarify.)
- ☐ Say, “We need your help designing this part of the user interface, so give me your honest feedback. You’re not going to hurt my feelings. I want to make this work well for you.”
- ☐ Ask, “For the task of X, how well does this help you do that task?” (Where X is something like verifying identity, or updating contact information.) You could even ask, “On a scale of 1 (which would be crappy) to 5 (which would be awesome), how well does this help you?” If they say something below 5, ask them what it would take to get the UI to the next higher number. If they say 5, ask them what makes it a 5. (Take notes.)
- ☐ Ask, “What’s missing?” (Make a list)
- ☐ Ask, “Is there anything here that you don’t need to do that task – that we can take away?” (Make a list)
- ☐ Wrap up: “What else?”
- ☐ Thank them profusely, tell them how helpful this was.

10. Show what's clickable

Though we want as little decoration as possible, we do want to make it easy for users to see what is clickable and what is not.

It is not important to underline links in this user interface, but it is important to show that column headers can sort column data. It is important that buttons look like buttons. These signals are called "affordances."

CAN USERS:

- ☐ *Tell what is clickable without waving the mouse around and waiting to get the finger?*
- ☐ *See that a clickable thing has been clicked or not?*

11. Show changes in state

If the something has been worked on, show that by changing how it looks and behaves.

If it is completed, move it out of the workspace. If the state of the data changes within an interaction in the user interface, make it clear to the user that the state has changed.

CAN USERS:

- ☐ *Quickly and easily tell what state a task or a piece of data is in?*
- ☐ *See that a clickable thing has been clicked or not?*
- ☐ *Easily see where a case is in the workflow?*

12. Support quick and easy error recovery

Allow for undoing or correcting before committing the data to the database.

Show error messages in-line. Instrument the error messages so we can track recurring issues and remedy them.

CAN USERS:

- ☐ *Correct mistakes easily?*
- ☐ *Recover from making errors quickly and easily?*

13. Show changes in state

The point of tabs is to logically chunk content into sections that the user doesn't need to see all at once. Each should be parallel in meaning to the other. There should be no need to create tabs in a UI that supports a specific user task.

HAVE YOU:

☐ *Created a UI free of tabs?*

14. Minimize modals and popups

If you feel like you need to create a modal or a popup, reconsider. All information and error messages should be in-line, near where the issue is triggered.

Use save-on-blur when possible. Use 1- click save when not possible. Consider putting auto-save on a timer, when appropriate.

HAVE YOU:

- ☐ *Designed all messages in-line?*
- ☐ *Minimized explicit actions users must take to save data?*

FINAL DESIGN TIPS

- Don’t use heavy styling and complex functionality if you don’t need it.
- Try to make the UI look and feel light to support the flow of the work.
- Reuse what makes sense from earlier releases for optimal functionality and product design, but don’t force it. Create a user interface that supports users’ needs.