Design Guidelines Example

These guidelines are not hard rules. They are best practices and guardrails we use for consistency. Exceptions should be rare, thoughtfully considered, and justified in documentation.

Panels, Menus, Overlays

- Overlays triggered by a hard interaction (click) must have an explicit close control.
- Overlays triggered with a soft interaction (hover) do not need a close control.
- The control that opens a surface should also close it.
- Overlays should remain visually attached to their trigger, except modals and dialogs.
- Floating UI should indicate origins with motion or positioning.
- A collapsed panel must leave behind a visible reopen affordance.
- Panels should preserve state when temporarily hidden.
- Avoid opening multiple competing overlay layers.
- Menus should close on external interaction unless persistence is required.
- Temporary overlays should not permanently alter layout.
- Persistent panels should resize layout predictably.
- Flyout panels should use standardized controls and placement.
- Modal dialogs should block only high-risk or high-focus tasks.
- Modals should be reserved for simpler actions or dialogs.
- Modals should avoid internal scrolling when possible.

Motion & Spatial Design

- Motion should originate from the interacted element.
- Objects should not unexpectedly change position during interaction.
- Animation should preserve spatial continuity.
- Layout shifts during interaction should be minimized.
- Expanding containers should grow from their trigger edge.
- Objects entering should ease out, objects leaving should ease in.
- Motion duration should reflect interaction weight.
- Background content should remain visually stable during foreground interactions.
- Drag interactions should expose clear affordances.

Navigation, Layout, & Hierarchy

- Navigation should not change location between views.
- Visual hierarchy should support rapid scanning and orientation.
- Layout structure should remain stable during task execution.
- Primary actions should occupy consistent locations.

ⓒ 2026 Tom Greever - tomgreever.com/resources - from the book Articulating Design Decisions

- Secondary actions should not visually compete with primary actions.
- Toolbars should not contain navigation.
- Navigation should not contain manipulation tools.
- Global navigation and local navigation should remain visually distinct.
- Contextual navigation should appear adjacent to the content it affects.
- Settings pages should use left-panel navigation, no breadcrumb.

Accessibility & Usability

- Interfaces should remain usable without relying on color alone.
- Interactive elements must provide visible focus states.
- Keyboard interaction patterns should mirror pointer interactions where possible.
- Hover interactions should not gate access to essential functionality.
- Text should remain legible across supported sizes and zoom levels.

Information Architecture

- Related information should be grouped together.
- Parent-child relationships should be inferred by the layout.
- Information should appear where it is most likely to be needed.
- Objects should have a single canonical location.
- Organizational structures should prioritize retrieval over categorization purity.

Tool vs Container Behavior

- Tools manipulate objects or content.
- Menus reveal destinations, navigation, or containers.
- Icons that open containers should behave consistently.
- Toggleable tools must visibly indicate active state.
- Temporary tools should revert predictably after use.
- Persistent tools should maintain visible state.
- Viewer controls should remain separate from application chrome.
    - Viewer horizontal toolbars imply content manipulation.
    - Vertical sidebars imply navigation or structural context.

State & Visibility

- Users should never lose awareness of system state.
- Interfaces should support interruption and resumption of tasks.
- Active, selected, disabled, loading, and error states must be visually distinct.
- Changes in state should always be visually communicated.
- Unsaved changes must persist visibly until resolved.
- Empty states should provide a next action.

ⓒ 2026 Tom Greever - tomgreever.com/resources - from the book Articulating Design Decisions

- Background processes should expose progress.
- Notifications should not obscure primary workflows.
- Temporary status messages should dismiss predictably.
- Loading and empty states should be appropriate for the context and content.
- Hidden functionality should remain discoverable.

Data Density & Complexity

- Tables should prioritize scanning over decoration.
- Important columns should remain visible during horizontal scroll.
- Row actions should remain discoverable without hover dependency.
- Sorting and filtering should not reset unrelated user state.
- Dense interfaces require stronger hierarchy, not more spacing.
- Inline editing should not adjust or reflow any content surrounding it.
- Bulk actions should activate only when relevant.
- When data can be outdated, clearly communicate the last date updated.
- Selection state should remain stable across pagination and filtering.
- Something about visually connecting long horizontal rows of data (lines / hover)
- Zero results and empty states should be treated differently.
- Truncation should preserve the most important information.
- Metadata should remain visually subordinate to primary content.
- Data freshness should be communicated when relevance depends on time.
- Comparisons should be visually supported when comparison is the primary task.

Interaction Safety & Recovery

- Prefer undo over confirmation.
- Destructive actions should require intentional interaction.
- Dangerous actions should be visually differentiated.
- System feedback should appear immediately after action.
- Actions where delays are unavoidable should communicate a pending state.
- Failed actions should preserve user input whenever possible.
- Recovery paths should be adjacent to errors.
- Long-running actions should expose cancellation when possible.

Progressive Disclosure

- Start with the minimum necessary controls.
- Advanced functionality should not compete with primary workflows.
- Frequently used actions should require fewer interactions.
- Don't hide required inputs behind collapsible content.
- Where possible, include hints of what is being hidden.
- Optional configuration should remain secondary.

ⓒ 2026 Tom Greever - tomgreever.com/resources - from the book Articulating Design Decisions

- Interfaces should optimize for the common path first.
- Complexity should appear only when contextually relevant.

Consistency & Systems

- Similar interactions should behave identically across the product.
- Shared components should preserve interaction patterns.
- Keyboard behavior should remain consistent globally.
- Scroll behavior should not vary between similar surfaces.
- Resizing behavior should follow predictable constraints.
- Similar interaction patterns should preserve consistent controls, positioning, and motion.
- The same icon should not perform different actions in different contexts.
- Interaction patterns should be reusable before introducing new ones.

Content & Copy

- Error states should explain both the problem and the recovery path.
- Content structure should remain understandable independent of styling.
- Content should use active voice.
- Labels should reflect user language rather than system language.
- Similar content should be organized consistently.

Color, Attention, & Emphasis

- Color should communicate meaning, not decoration.
- Attention should be directed intentionally and sparingly.
- Multiple elements should not compete for primary attention.
- Selection, focus, and active states should remain visually distinct.
- Red should be reserved for destructive actions, errors, and warnings.
- Persistent attention indicators should require persistent relevance.
- Visual emphasis should reflect task priority.
- Interfaces should not rely on color alone to communicate state.
- Motion and color should not compete for attention simultaneously.
- High-emphasis treatments should be used consistently across the product.

Feedback Scope

- Loading states should appear at the level being affected.
- Feedback should avoid blocking unrelated work.
- Feedback should match the scope of the affected object.
    - One section is expanded → only that section changes state.
    - A single card is loading → only that card shows a loading state.
    - A row is selected → only the row appears selected.

ⓒ 2026 Tom Greever - tomgreever.com/resources - from the book Articulating Design Decisions

    - One field has an error → the field shows an error state.

User Control & Automation

- Automatic behavior should remain predictable.
- Automation should reduce effort without reducing control.
- System-generated actions should remain reversible when possible.
- Interfaces should not take actions on behalf of users without clear benefit.

ⓒ 2026 Tom Greever - tomgreever.com/resources - from the book Articulating Design Decisions