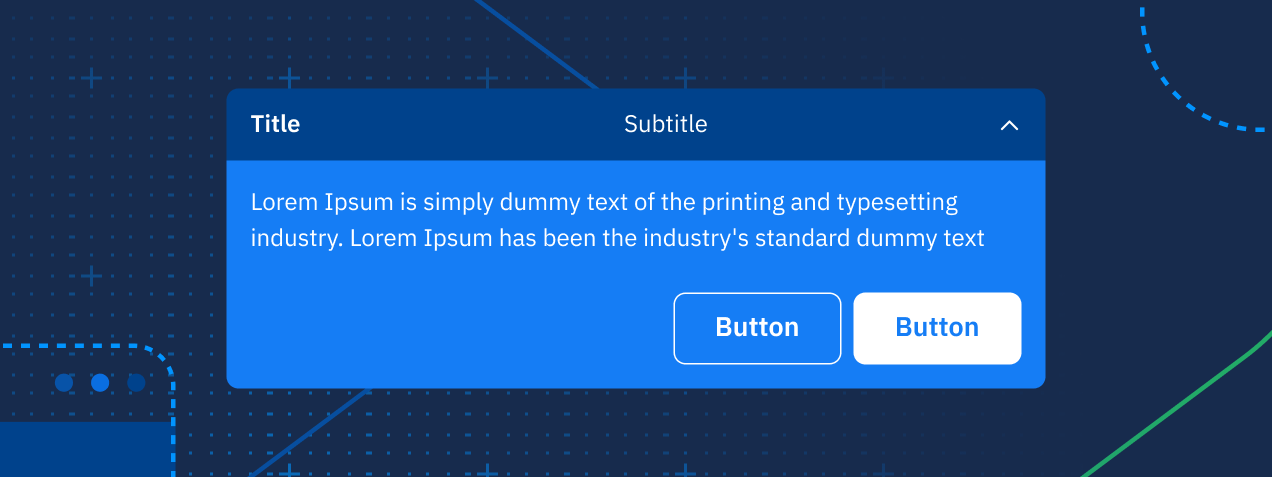
# Accordion



### **1. General Information**

| Component name: | **Accordion** |
| --- | --- |
| Category: | **Interactive Components** |
| Description: | The Accordion component is a versatile UI element used to expand and collapse sections of content. |
| Objective: | The objective of an Accordion is to organize large amounts of content into collapsible sections, allowing users to expand only what they need—reducing visual clutter and improving content accessibility. |

### 

| illustration: | |
| --- | --- |
|  |  |
| ✅Maintain standard sizes for different contexts to ensure uniformity across the user interface. | ❌Avoid inconsistent sizing that can lead to a disjointed UI. |

### 

### **2. Usage Guidelines**

| ✅It should be used when: | Use Accordion when displaying categorized or grouped content—like FAQs, product/service details, or multi-section forms—where expanding one section at a time improves usability. |
| --- | --- |
| ❌Should not be used when: | Avoid using Accordion when all content is equally important and should be shown simultaneously, or when the content is too short to justify hiding. |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | A well-designed Accordion should have clear section titles, visual indicators (like arrows) for open/close states, keyboard accessibility, and smooth content transitions. |
| ❌Guidelines to avoid: | Avoid hiding critical content inside an Accordion without emphasis, opening too many panels at once if it causes confusion, or adding overly complex interactions inside the sections. |

### 

| Example of Best Practices: |
| --- |
| On a customer support page, Accordions categorizes FAQs by topic. Users can click on a question to expand the section and read the answer, keeping the page neat and enabling users to find the information they need quickly. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **Yes, it supports the responsive design but only the width.** |
| illustration: | |
|  | |
| **Headers** | The text is bold, and the subtitle is here. There are clickable titles with icons indicating the expansion state. |
| **Content Area** | Contains detailed information, accessible upon interaction. |
| **Icons** | Collapse icon for expanding sections for open sections, guiding user interaction. |
| **Design** | Features light borders or separation to distinguish between sections, maintaining a clean and organized layout. |

### **4. Accessibility**

| **Screen Reader Support** | Accordion headers and content are properly labeled with ARIA attributes, ensuring screen readers can accurately convey each section's state and purpose to users. |
| --- | --- |
| **Keyboard Navigation** | * Tab moves focus between accordion headers. * Enter or Spacebar toggles the expansion of content. * Arrow Keys (optional) can navigate between headers, with Home and End jumping to the first and last header. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Accordion often pairs with components like Cards, Icons, Lists, or Form elements to create structured, scannable content blocks. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Tabs, Expandable Cards, or simply showing all content in a flat layout. The best choice depends on the content and user goals. |
| illustration: | |
| Supports linking or buttons within accordion content for dynamic user engagement. | |

### 

### **7. Specification**

| Spacing - Collapse |
| --- |
|  |
|  |
|  |
|  |

#### 

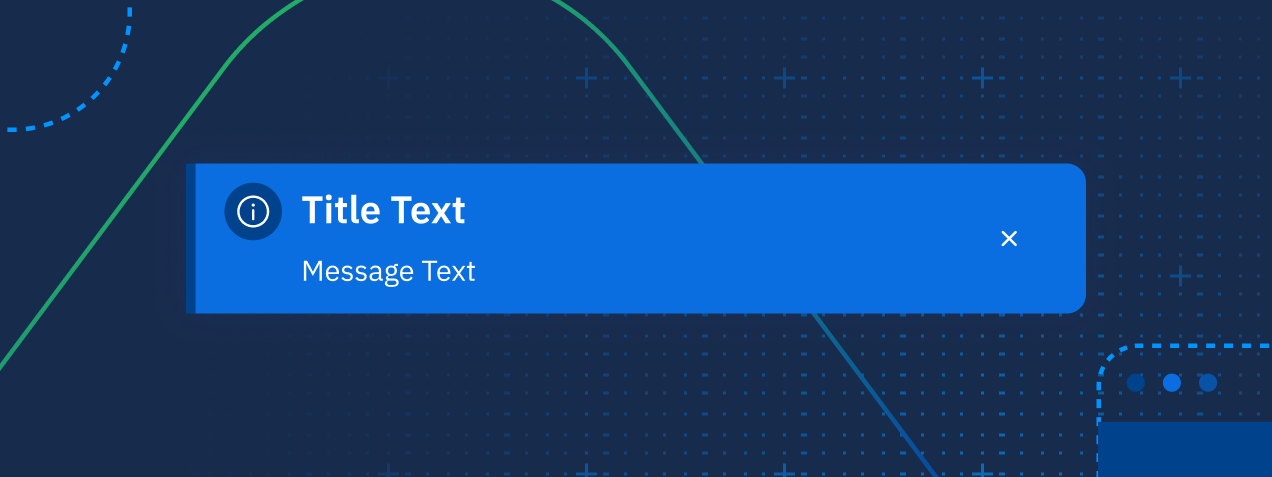
| Properties - Collapse |
| --- |
|  |
|  |
|  |
|  |

#### 

| Spacing - Expanded |
| --- |
|  |
|  |
|  |
|  |
|  |

| Properties - Expanded |
| --- |
|  |
|  |
|  |
|  |

# Alert



### **1. General Information**

| Component name: | **Alert** |
| --- | --- |
| Category: | **Display & Feedback Components** |
| Description: | **The Alert component serves as an interactive feedback mechanism. It is triggered in response to specific user actions or system events.** |
| Objective: | It is divided into four types: success text, information text, warning text, and error text. Toasts are used to tell users what they are facing or what state is happening in their user journeys.  When writers design notification messages, they should consider why users have to notice what happened so they realize what to do in the next step. Each type of toast is different based on its purpose. |
| Type of Alert: | |
| Success | Text is represented in green toast; it indicates that procedures, transactions, or final steps proceed ultimately. |
| Information | Text is always shown in blue toast; it informs users about details that might be useful for taking another proper action in their user journey. |
| Warning | Text is generally displayed on yellow toast. This will inform users about crucial issues affecting the journey. A message also generates beneficial solutions or suggestions, guiding the user to act properly. |
| Error | Text is represented in red toast; it informs users about failures, problems, or something wrong with the processes. |

### 

| illustration: | |
| --- | --- |
| ✅Use concise, clear language that directly relates to the action taken.  Ensure that the alert appears in a consistent location across the application. | ❌Do not use alerts for non-essential information that could be conveyed less intrusively.  Avoid stacking too many alerts, which can overwhelm the user. |

### 

### **2. Usage Guidelines**

| It should be used when: | Use an Alert to notify users about important information like errors, warnings, success messages, or general updates that require immediate user awareness. it can be categorized into the following main types:  **1. Success**  A **Success Alert** indicates that a process or action has been successfully completed. It is commonly used for actions such as saving data, submitting forms, or confirming a completed operation. This type of alert typically uses a green color and a checkmark icon to signify success and completion.  **2. Warning**  A **Warning Alert** is used to caution users about potential risks or issues, such as actions that may have unintended consequences, security concerns, or session expiration. Yellow is typically used to indicate caution, prompting users to review details before proceeding.  **3. Error**  An **Error Alert** notifies users of critical issues or failures, such as incorrect input, failed network connections, or actions that cannot be completed. It usually appears in red with an exclamation mark icon to grab attention and encourage users to take corrective action.  **4. Informational**  An **Informational Alert** provides general updates or important information that users should be aware of but do not need to act on immediately. Examples include notifications about new features, system updates, or policy changes. Blue is commonly used to represent neutrality and trustworthiness. |
| --- | --- |
| Should not be used when: | Avoid using Alerts for trivial information or too frequently, as users may become desensitized and start ignoring them.. |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | Use icons, colors, and text that match the alert type (e.g., green for success, red for error). Display the alert in a noticeable location, and keep the message short, clear, and easy to understand. |
| ❌Guidelines to avoid: | Avoid displaying too many alerts at once. Don’t use overly long messages or complex wording. And never use the wrong color or icon, as it can confuse users or cause misinterpretation. |

| illustration: |
| --- |
| Example of Best Practices: |
|  |
|  |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **No, you can only choose these 4 colors:** Green (Success), Yellow (Warning), Red (Error) and Blue (Informational) |
| Does it support Responsive Design? | **Yes** |
| illustration: | |
| Anatomy of Alert | |
| Apperarnce | |
| Colors | Based on the alert type, the left border and icon colors vary   * Green (success) * Yellow (warning) * Red (error) * Blue (informational) |
| Iconography | Each alert type has a corresponding icon (checkmark for success, exclamation mark for warning, etc.) for quick recognition. |
| Typography | It utilizes AXONS's font, ensuring readability across various screen sizes. The text size and weight are balanced to stand out without being overwhelming. |
| Action Button | A subtle close button (×) on the center-right maintains the component's global design. |

### **4. Accessibility**

| Screen Reader Support | Each alert is assigned an appropriate ARIA role and labeled to ensure it is announced correctly by screen readers. The aria-live attribute is assertive for critical alerts and polite for less urgent ones. |
| --- | --- |
| Keyboard Navigation | * Focus Management: The component does not steal focus upon appearance but can be dismissed with the Esc key. (Optional) * Visibility Time Extension: Hovering over or focusing on the alert extends its visibility duration for keyboard users. (Optional) |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

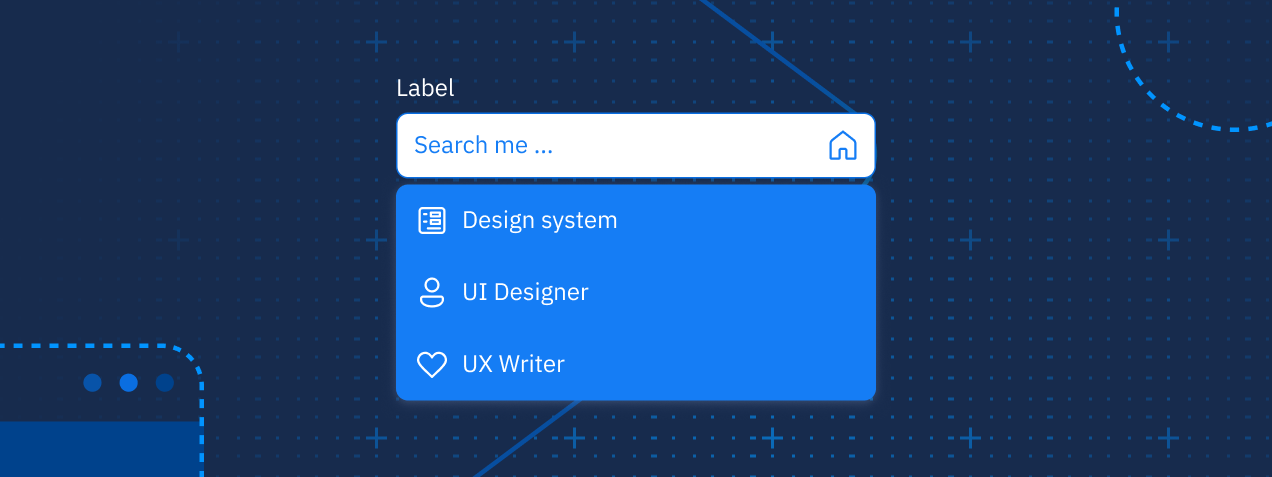
| Components that are often used together: | Alert components often work with Icons, Buttons, Toasts, Modals, or Status Indicators, and can appear inside Forms or Cards for contextual feedback. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Toasts for temporary notifications, Modals for critical user decisions, or inline text for passive feedback. |

### **7. Specification**

| Spacing |
| --- |
|  |
|  |
|  |
|  |
|  |

| Properties |
| --- |
|  |
|  |
|  |
|  |

## Autocomplete



### **1. General Information**

| Component name: | **Autocomplete** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | **The Autocomplete component enhances text input fields by providing real-time suggested options based** |
| Objective: | For writing autocomplete, writers only focus on writing messages in the input field. Writers should select short, clear, and valuable texts to guide users in what they will type and get accurate keywords that they are looking for. Text in the input field might be said to inform and guide users to act appropriately when interacting in a specific state.  Also, the text input field is divided into four types of writing based on tone, voice, and writing methods that writers consider and select the proper one from: |
| Style of Autocomplete | |
| Interrogative style | It is a sentence that asks a question, urges usersit'sthink, and suits platforms that set an informal and casual tone and voice and want to initiate more conversations with users. |
| Categorized style | It is the main keyword of a technical or specific wording group used in a specific context. This wording suits personas with specialized skills and requires jargon or particular wording in communication. Writers can apply this type of text to scope keywords used in search boxes or guide them to specify wording more accurately. |
| Exemplifying style | It uses sample words or phrases directly indicating specific texts to clarify what users should type in the input fields. |
| Suggesting style | It is a sentence started with sentences "Enter," "Please [do some "hing]," , "r other verb actions b "in context. It's used to suggest how to enter text in the input fields correctly. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅ Provide clear visual feedback for matching text in suggestions. | ❌Don't overwhelm the user with too many suggestions; limit the number displayed. |
|  | ❌ |
|  | ❌Don't use visual or icon elements too much in the suggestion |

### 

### **2. Usage Guidelines**

| It should be used when: | Use Autocomplete when users need to input something from a known list—like searching for products, selecting users, or choosing a country. It helps speed up input and reduces typing errors. |
| --- | --- |
| Should not be used when: | Avoid using Autocomplete when the list is short and easily shown—like menus with fewer than 5 options. Also, don’t use it when you don’t have reliable data to support meaningful suggestions. |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | Suggestions should appear as soon as the user starts typing, with matching terms highlighted. It should support keyboard navigation and allow both selection from the list or manual input when appropriate. |
| ❌Guidelines to avoid: | Avoid slow responses or lag when typing quickly. Don’t overload with too many suggestions. Also, don’t force users to pick only from the list unless it’s truly required. |

| illustration: |
| --- |
| Example of Best Practices: |
|  |
| In a platform, the Autocomplete component is used in the search bar. As users begin to type the name of a person, group, or hashtag, suggestions appear, allowing them to quickly find and navigate to the desired profile or page. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Medium** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **Yes** |
| illustration: | |
| **Appearance** | |
|  | |
| **Input Field** | Styled consistently with the application's design system, users type their search or selection criteria in the input field. |
| **Suggestions List** | A dropdown list appears as the user types, displaying options matching the input. It can include text, icons, or images as suggestions. |
| **Behavior** | |
|  | |
| **Suggestions** | The list of suggestions updates in real-time as the user types, based on the current input value. |
| **Selection** | Users can select an option from the suggestions list by clicking on it, navigating with keyboard arrows, or pressing Enter. |
| **Limiting Suggestions** | To keep the list manageable, limit the number of displayed suggestions. |
| **Debouncing** | Implements a slight delay in fetching and displaying suggestions after the last keystroke to reduce the load on the server and improve performance. |
| **User Interaction** | |
|  | |
| **Typing Input** | As the user types, suggestions appear based on the entered text. |
| **Keyboard Navigation** | Users can use the arrow keys to navigate up and down the suggestions list and Enter to select an option. |
| **Mouse Interaction** | Clicking on a suggestion selects it, filling the input field with the chosen option. |

### **4. Accessibility**

| Screen Reader Support | Utilize ARIA roles and properties, such as role="content" and role=" option," to enhance accessibility. Ensure suggestions are readable by screen readers, providing context about the number of suggestions available and how to navigate them. |
| --- | --- |
| Keyboard Navigation | Ensure all interactions, including navigating and selecting suggestions, can be performed with the keyboard. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Autocomplete typically works with Text Inputs, Dropdowns, Chips (for multi-select), Icons, and Loaders to indicate data fetching. |
| --- | --- |
| Components that can be used interchangeably: | If Autocomplete is not used, alternatives include:   * Dropdown (Select Menu): Best for a limited number of options that users can scroll through easily. * Radio Buttons: Ideal for a small set of choices where users need to see all options clearly. * Checkboxes: Suitable for selecting multiple options at once. * Manual Input (Text Field): Works well when the input is highly specific or unpredictable. * Predictive Search with Filters: Allows users to search and filter results manually instead of seeing automatic suggestions.   The best choice depends on the type of data, the number of options, and the desired user experience for optimal usability. |
| illustration: | |
|  | |
| **Single** | Autocomplete can be designed for single selection (Choosing one option) |
|  | |
| **Multi-Select** | Multi-select (Tagging multiple options). |

### **7. Specification**

| Spacing |
| --- |
|  |
|  |
|  |
|  |

| Properties |
| --- |
| State |
|  |
|  |
|  |
|  |
|  |
| Error State |
|  |
|  |
| Disabled State |
|  |
|  |

| Show List |
| --- |
|  |
|  |

## Breadcrumb



### **1. General Information**

| Component name: | **Breadcrumb** |
| --- | --- |
| Category: | **Navigation Components** |
| Description: | **Breadcrumbs are a navigational aid in user interfaces.** |
| Objective: | As a component of navigation controls, breadcrumbs maintain the primary function of guiding users to the survey platform structure. It usually appears on the first page of an application or website to display the main structure; breadcrumbs navigate the journey path in orderly sequences from main category to subcategory, such as **Home > Men > Clothing > Trousers.** |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
|  |  |
| Keep breadcrumbs concise. Use short names for each level. | Don't use breadcrumbs as the primary navigation element. |
|  |  |
| Place the breadcrumb at the top of the page, below the navigation bar but above the main content. | Avoid using breadcrumbs if the site structure is too shallow to benefit from them. |
|  |  |
| Ensure the current page is the last item in the breadcrumb trail and is not clickable. | Don't create dead ends. Users should be able to navigate to any page in the breadcrumb trail. |

### 

### **2. Usage Guidelines**

| It should be used when: | Breadcrumbs should be used when users need to navigate systems with deep or complex hierarchies—such as e-commerce websites, document management systems, or content-heavy platforms. They help users understand their current location within the system and provide an easy way to backtrack to previous levels, enhancing navigation efficiency and reducing confusion. Breadcrumbs are especially useful when users can enter the site from various entry points (e.g., via search) and need orientation within the structure. |
| --- | --- |
| Should not be used when: | Breadcrumbs should be avoided in simple or flat structures, such as apps with only a few main pages or without clear content hierarchies. In these cases, breadcrumbs may unnecessarily clutter the interface without adding real value. They are also not suitable for linear experiences, like onboarding flows or step-by-step forms, where navigation is meant to follow a specific sequence. In such scenarios, breadcrumbs could confuse users rather than help them. |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | Breadcrumbs should be used in systems with deep structures of three or more levels, such as Home > Category > Subcategory > Product, to help users easily understand their current location. They should be placed at the top of the content area, above the main page title, and show the full path from the homepage to the current page. All links, except the last one, should be clickable to allow users to navigate back. |
| ❌Guidelines to avoid: | * Don’t use breadcrumbs for flat or simple structures. * Avoid breadcrumbs in linear flows like onboarding or step-by-step forms. * Don’t use breadcrumbs as a replacement for primary navigation. * Avoid vague labels like “Page 1” or “Level 2” — use clear, descriptive names. * Don’t hide breadcrumbs inside dropdowns or menus. * Avoid showing breadcrumbs if they don’t match the actual nDon’t include breadcrumbs on homepages or single-level pages. |

| illustration: |
| --- |
| Example of Best Practices: |
|  |
| In an e-commerce application, a user navigates to a product detail page through the following path:  Home > Electronics > Laptops > Gaming Laptops > Product Name. The breadcrumb at the top of the product detail page reflects this path, with each section except "Product Name" being a clickable link. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **It is typically displayed as a horizontal list of links separated by a delimiter (e.g., a right angle >).** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **Use subtle colors that contrast with the background but don't dominate the page. The active or current location is usually displayed in a bolder color or font weight without a hyperlink.** |
| Does it support Responsive Design? | **Font size should be smaller than the main content but large enough to be legible. Use the application's default font family for consistency.** |
| illustration: | |
| Appearace Color | |
| Appearace Sytle | |
| User Interaction | |
| Clickable Links | **Except for the last item, which represents the current page, each element in the breadcrumb trail is a clickable link. Users can interact with these links to quickly navigate to previous sections or the home page.** |
| Navigation Feedback | **When a user hovers over a breadcrumb link, visual feedback (such as a change in color or an underline) indicates that the text is clickable. This feedback helps to distinguish navigation elements from static text.** |
| Truncation Interaction | **On smaller screens or for lengthy breadcrumb paths, breadcrumbs might be truncated to conserve space. Interaction with a truncated part (often indicated by an ellipsis ...) might expand the entire breadcrumb trail or offer a dropdown menu to access hidden sections.** |

### **4. Accessibility**

| Screen Reader Support | **Use aria-label="breadcrumb" on the navigation element to identify it as a breadcrumb navigation for screen readers.** |
| --- | --- |
| Keyboard Navigation | **Users can navigate through breadcrumb links using the keyboard. This includes tabbing through links and selecting them with the Enter key, ensuring the component is accessible to users relying on keyboard navigation.** |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

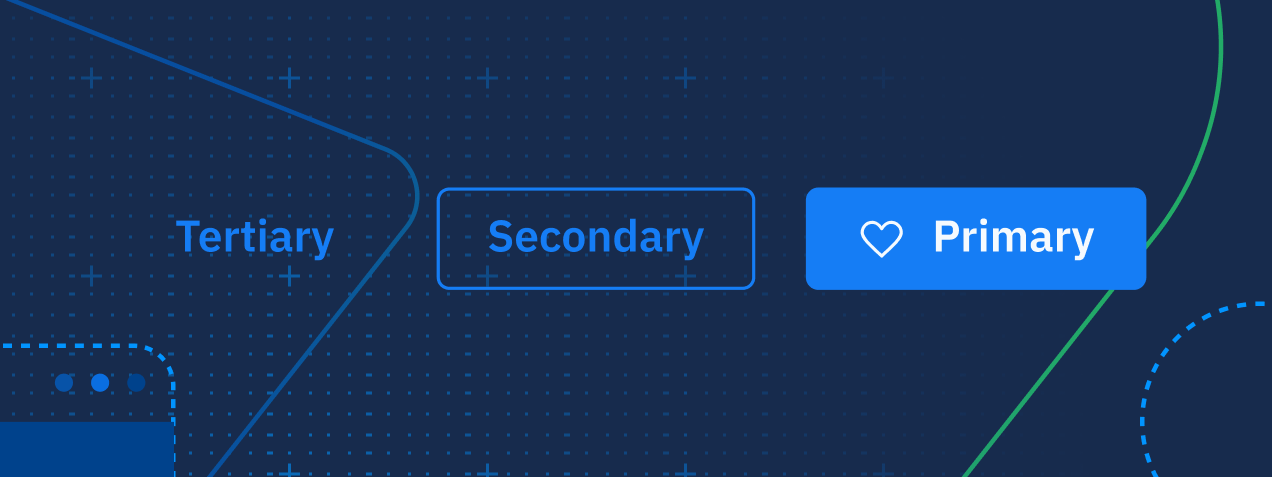
### **6. Related Components**

| Components that are often used together: | Tooltip |
| --- | --- |
| Components that can be used interchangeably: | **Sticky Header or Navigation Bar** - Keep the main menu always visible so users can navigate easily.  **Tab Navigation** - Use tabs for switching between related sections at the same level.  **Search Functionality** - Let users find content directly without browsing through levels.  **Progress Indicator** - Show steps clearly in flows like sign-up or checkout.  **Contextual Back Button** - Use back buttons with clear labels, like “Back to Category,” to guide users.  **Card Layout with Clear Labels** - For simple systems, well-labeled cards or content boxes help users know where they are. |
| illustration: | |
|  | |

### **7. Specification**

| Spacing |
| --- |
|  |
|  |
| Properties |
| State |
|  |
|  |
|  |
|  |
|  |
|  |

## Button



### **1. General Information**

| Component name: | **Button** |
| --- | --- |
| Category: | **Interactive Components** |
| Description: | **This component is crucial for creating a seamless and intuitive user experience, enabling users to interact with the application effectively.** |
| Objective: | **The button is a crucial UI component for user experience design; users use it to determine what to do next.**  **Also, buttons mainly tell users what to do in the final steps, like CTA (Call-To-Action).** |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| Use clear and concise labels. | Overuse primary buttons to avoid diluting user focus. |
|  |  |
| Employ primary buttons for the most common action on a page. | Place buttons in locations where they might be missed or confused with other elements. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Button when you want the user to perform an action—like submitting a form, confirming a purchase, or opening a modal. Buttons are essential for prompting clear, intentional actions.** |
| --- | --- |
| Should not be used when: | **Avoid using Buttons when there’s no real action—like just displaying information or simple navigation (use a Link instead). Also, don’t overload the UI with too many buttons, as it can confuse users.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Buttons should have clear, concise labels like “Submit” or “Save.” Use consistent color schemes to indicate types (e.g., Primary, Secondary, Danger). Ensure there’s enough spacing so buttons aren’t placed too close together.** |
| ❌Guidelines to avoid: | Avoid vague button labels like “OK” without context. Don’t use colors that blend into the background, making the button hard to see. Also, don’t forget to show button states like loading, hover, or disabled for proper feedback. |

| illustration: |
| --- |
| Example of Best Practices: |
|  |
|  |
|  |
| Consider a form submission scenario. The user fills out the form and clicks the primary 'Submit' button. The button changes to a loading state, providing immediate feedback on the action, and then returns to its default state upon successful submission, possibly accompanied by a success message. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Small (Size S) / Medium (Size M) / Large (Size L)** |
| --- | --- |
| Icon supported or not: | **Yes. We can include relevant icons to enhance user understanding (e.g., a plus icon for an 'Add' button).** |
| Can the color be adjusted? | * **Brand CI colors -** Brand CI colors are primary colors derived from the company's branding palette. These colors are a visual representation of the brand identity.   + Appearance: Vibrant and distinctive, aligning with AXONS brand palette. * **Functional colors** - Functional colors indicate the nature or status of an action or element, such as success, warning, or error.   + Usages: Applied to highlight specific actions like 'Submit' (success/green), 'Warning' (yellow), or 'Error' (red).   + Appearance: The color corresponds to the function: green for success, yellow for warning, red for error, blue for information, etc. * **Neutral Color** - Neutral colors are unsaturated colors like grey and off-white. They are versatile and less visually demanding.   + Usages: They are typically used for secondary or tertiary buttons where less emphasis is needed.   + Appearance: Subtle and understated, it is often used for button backgrounds or borders in secondary and tertiary buttons. For readability, the text is usually a darker shade. |
| Does it support Responsive Design? | **Yes** |
| illustration: | |
| **Appearance**  Adaptable to various themes while maintaining sufficient contrast for accessibility. | |
| **Colors** | |
|
| **Iconography** We can include relevant icons to enhance user understanding | |
| **Typography**  It uses the AXONS default font, with size and weight that ensure legibility and clarity. | |
| **Shape** Rounded corners that align with AXONS design language, promoting a friendly and approachable interface. | |
| **Anatomy** | |

### 

| **Behavior** - Visual and tactile feedback on interaction, such as color change or ripple effect, to confirm user action. | | | |
| --- | --- | --- | --- |
| **Action State** | **Default** | Description: | The default state is how the button appears under normal, un-interacted conditions. |
| Appearance: | The text will be in solid, outline, or ghost style depending on the variant. It will be clear, legible, and have an optional icon. |
| User Interaction: | The button is clickable and fully functional. |
| Best Practices: | Maintain consistent styling across similar buttons for uniformity. |
| **Hover** | Description: | The hover state is activated when the user's cursor is over the button. It provides visual feedback that the button is interactive. |
| Appearance: | Slight visual change, such as a color shift, brightness change, or border thickening. For Ghost Buttons, an underline or color change can be effective. |
| User Interaction: | Indicates to users that the button is clickable. |
| Best Practices: | When signaling interactivity, ensure the hover state is noticeably different from the default state. Avoid drastic changes that might confuse the user. |
| Pressed | Description: | The active state signifies the user has clicked or is pressing the button. |
|  | Appearance: | It often includes a more pronounced color change, giving the impression of being pressed down. It may also feature a shadow effect or an animation like a ripple. |
|  | User Interaction: | This state is usually brief, occurring during the click before an action is initiated. |
|  | Best Practices: | The active state should provide immediate, tangible feedback to reinforce the user's action. The visual effect should be subtle yet satisfying, contributing to a responsive user experience. |
| **Disabled** | Description: | The disabled state shows that the inactive button cannot be interacted with. |
|  | Appearance: | Typically, the button appears faded or has a lower contrast. Text and icons may also appear muted, and the cursor may change to a 'not-allowed' symbol on hover. |
|  | User Interaction: | The button does not respond to clicks or any form of interaction. |
|  | Best Practices: | Use this state to indicate that certain conditions must be met before the button can be activated. Ensure it's visibly distinct from the active states to prevent user confusion. |
| Loading State |  | Description: | An optional loading state with a spinner for actions that require processing time. |

### 

| **Variants** | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | | |
|  | | | | | | | | | | | |
| **Solid Button**  (Primary Button) | | | | Description: | | | | The Solid Button is the most prominent variant, designed to stand out and used for primary actions. It's characterized by a filled background, making it highly visible. | | | |
| Usage: | | | | Ideal for critical actions such as 'Submit,' 'Save,' or 'Next.' It draws attention to the primary action on a page or form.- | | | |
| Appearance: | | | | - Bold, solid color background (usually the primary brand color).  - Contrasting text color for high legibility. | | | |
| Best Practices: | | | | - Limit to one per interaction area to maintain focus on the primary action.  - Ensure the color contrasts nicely with the background for accessibility. | | | |
| **Outline Button** (Secondary Button) | | | | Description: | | | | The Outline Button is less pronounced than the Solid Button and is used for secondary actions. It features a bordered style without a filled background. | | | |
| Usage: | | | | It suits secondary actions like 'Cancel,' 'More Info,' or 'View Settings.' It complements the Solid Button by providing an alternative without competing for attention. | | | |
| Appearance: | | | | - A clear, visible border with a transparent background.  - The text color typically matches the border color.  - It may highlight with a subtle color fill or darken the border when hovering. | | | |
| Best Practices: | | | | - Use alongside Solid Buttons to provide a secondary option.  - Ensure sufficient border contrast against the background. | | | |
| **Ghost Button**  (Tertiary Button) | | | | Description: | | | | The Ghost Button is the most subtle variant, used for the least prioritized actions. It typically appears as plain text without a solid background or border. | | | |
| Usage: | | | | Ideal for tertiary actions such as 'Learn More', 'Help', or additional options. It's used where a button is necessary but should not draw much attention. | | | |
| Appearance: | | | | - No background or border; appears as plain, clickable text.  - An icon next to the text may be included for added clarity.  - Subtle color changes or underlining on hover to indicate interactivity. | | | |
| Best Practices: | | | | - Employ where actions are secondary or tertiary in importance.  - Ensure text color maintains readability and indicates interactivity. | | | |

### **4. Accessibility**

| Keyboard Navigation: | **Fully accessible via keyboard, with focus indicators.** |
| --- | --- |
| Screen Reader Support: | **Properly labeled with ARIA attributes for assistive technologies.** |
| Click Action: | **Triggers a predefined action or event in the application.** |
| Disabled State: | **Indicates when a button is inactive or unavailable for interaction.** |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Dropdown |
| --- | --- |
| Components that can be used interchangeably: | * **Text Links:** Ideal for secondary actions like “Learn more” or “View details.” * **Icon Buttons**: Useful in limited space, e.g., for delete, edit, or share actions. * **Gestures (on mobile):** Such as swiping left/right to navigate or reveal options. * **Clickable Cards or List Items:** When entire blocks are tappable, like chat items or product menus. * **Switches, Toggles, Checkboxes, Radios:** For selections or settings without immediate actions. |

### **7. Specification**

| Spacing |
| --- |
| Size S |
|  |
|  |
|  |
| Size M |
|  |
|  |
|  |
| Size L |
|  |
|  |
|  |

| Properties |
| --- |
| State |
|  |
|  |
|  |
|  |

| Variant |
| --- |
|  |
|  |
|  |
|  |

## Carousel

### **1. General Information**

| Component name: | **Carousel** |
| --- | --- |
| Category: | **Interactive Components** |
| Description: | **The Carousel component displays a series of content items, such as images or text, in a sequential, rotating banner.** |
| Objective: | **It's a dynamic way to showcase featured content, products, or galleries in a confined space, allowing users to browse through multiple items without navigating away from the page.** |

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| Provide precise navigation controls. | Don't overload the carousel with too much content; keep it concise and engaging. |
|  |  |
| Use high-quality, optimized images to reduce load times. | Don't hide important information in carousel slides only; ensure key messages are accessible elsewhere. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Carousel when you want to display multiple pieces of content within a limited space—like product images or featured articles—allowing users to scroll through them without navigating away from the page.** |
| --- | --- |
| Should not be used when: | Avoid using a Carousel for critical or primary content like important alerts or decision-driving calls to action. Users often overlook additional slides or don’t realize that the content is scrollable. |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Keep the number of slides limited (ideally under 5), provide clear controls like arrows or dots, allow both manual and automatic sliding with user control to pause/play, and include text descriptions for better content clarity.** |
| ❌Guidelines to avoid: | Avoid placing important info beyond the first slide, as many users don’t scroll. Don’t make it autoplay too quickly or without controls. And never use a Carousel just for decoration—it should serve a clear purpose. |

| illustration: |
| --- |
| Example of Best Practices: |
|  |
| The website's homepage features a Carousel highlighting the latest collections. Users can browse through the featured collections manually using navigation arrows or pause the carousel to focus on a particular item of interest. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **No** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **Yes** |
| illustration: | |
|  | |
| It's a dynamic way to showcase featured content, products, or galleries in a confined space, allowing users to browse through multiple items without navigating away from the page. | |
| **Appearance** | |
| Slides | Individual items are displayed within the carousel. It can contain images, text, or other multimedia content. |
| Navigation Controls | Typically includes arrows on either side for moving between slides and a series of dots or thumbnails representing each slide for direct navigation. |
| Animation | Smooth transition effects (e.g., slide, fade) between slides to enhance user experience. |
| Layout | Responsive design ensures the carousel looks and functions well on all devices, adjusting size and interaction modes accordingly. |
| **Behavior** | |
| Auto-Rotation | Slides can automatically transition at a set interval, allowing users to pause and resume. |
| Manual Navigation | Users can manually advance to the next or previous slide using navigation controls. |
| Looping | After the last slide, the carousel can loop back to the first slide, creating a continuous cycle. |
| Accessibility | Provide options for users to control the carousel's autoplay feature and ensure keyboard navigability. |

### **4. Accessibility**

| Screen Reader Support | **Use ARIA roles (e.g., role="region", aria-live, aria-label) to describe the carousel and its components for screen readers.** |
| --- | --- |
| Pause/Play Controls | **Controls should be included to stop auto-rotation, which is crucial for users who need more time to consume the content or who use screen readers.** |
| Visible Controls | **Ensure that navigation controls are visible and accessible, with high contrast against the slide backgrounds.** |
| **User Interaction** | |
| Navigation Controls | Clicking arrows or indicator dots allows users to move through the carousel content. |
| Swipe Gestures | On touch devices, users can swipe left or right to change slides. |
| Keyboard Navigation | It supports left and right arrow keys to navigate between slides. |
| Focus Management | Ensures that focusing on a carousel item doesn't automatically change slides, providing control to users who rely on keyboard navigation. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

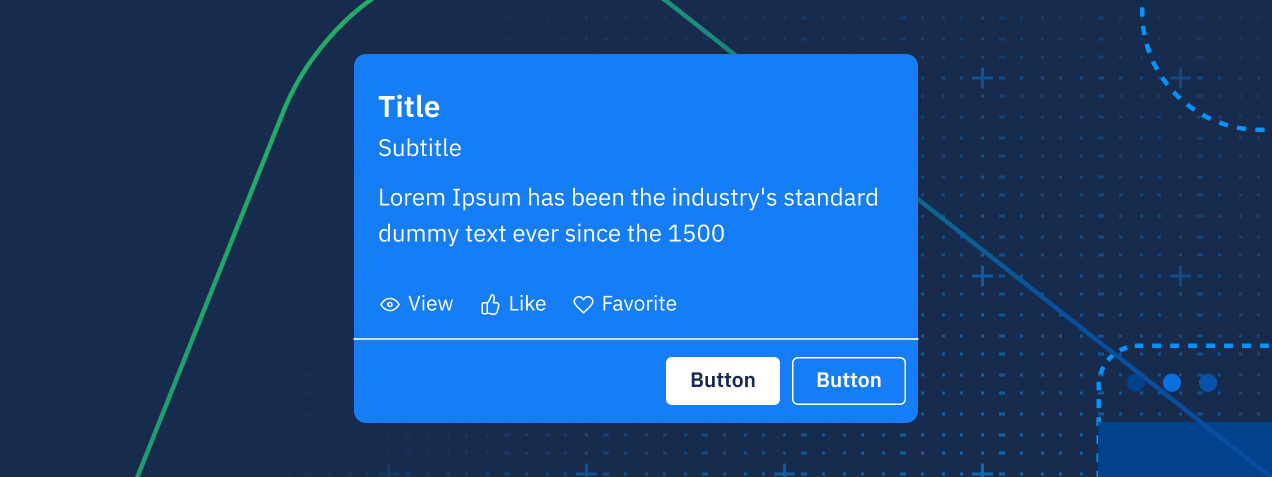
### **6. Related Components**

| Components that are often used together: | Card, Image / Media, Button / Icon Button, Dot Indicator / Pagination, Typography / Text, Dialog / Modal, Responsive Layout / Grid |
| --- | --- |
| Components that can be used interchangeably: | If you don’t use a Carousel, you can show all content at once using a Grid or List. Use Tabs to organize categories, or a simple Horizontal Scroll for natural side-scrolling. For text-heavy sections, use an Accordion. If showing one item at a time is important, a Modal with Stepper is a good alternative. |
| illustration: | |
|  | |

### **7. Specification**

|  |
| --- |
|  |
|  |
|  |

## Card



### **1. General Information**

| Component name: | **Card** |
| --- | --- |
| Category: | **Display & Feedback Components** |
| Description: | The Card component is a versatile UI element designed to present information in a compact and visually appealing format. |
| Objective: | Typically used to display a combination of images, text, and actions about a single subject, cards are ideal for showcasing products, articles, profiles, and more. They facilitate a clean, organized content presentation. |

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| Use cards to group related information into digestible units. | Don't overload cards with too much information or too many actions. |
|  |  |
| Provide clear visual and text cues for interactive elements. | Avoid making non-interactive elements look clickable, which can confuse users. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Card when you want to group related content visually, such as product items, articles, user profiles, or data blocks in a dashboard. Each Card should represent a single unit of related information.** |
| --- | --- |
| Should not be used when: | **Avoid using Cards when the content is highly sequential, like form steps or long articles, or when users need to scan lots of information quickly, like in data tables.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **A well-designed Card should be self-contained with clear visuals, title, description, and actions. Make good use of space and maintain consistent spacing for readability.** |
| ❌Guidelines to avoid: | **Avoid overloading Cards with too much content and don’t mix inconsistent card sizes in the same layout, as it creates visual clutter and disrupts the design flow.** |

| illustration: |
| --- |
| Example of Best Practices: |
|  |
| The template displays details using cards. Each card includes an image, name, detail, and button. Users can click on a card to view the following detailed page. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Mutisize** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Variants | |
| Image Cards | **Primarily feature an image with minimal text, suitable for galleries or portfolios.** |
| Content Cards | **Focus on text content, with optional images for articles, blog posts, or detailed information.** |
| Action Cards | **Include prominent action buttons or links, ideal for product listings or profiles.** |
| illustration: | |
| **Structure** - A card is usually a rectangular box containing image(s), text, and action buttons. It may include a header, a leading content area, and a footer. | |
| **Color** - Utilizes background colors that contrast nicely with the text, often using neutral shades or muted tones from the defined color palette to ensure readability. | |
| **Typography** - Employs straightforward, legible typography consistent with the brand's design system, distinguishing between titles, details, and action buttons. | |
| **Image** - Often feature an image at the top or integrated within the layout to visually represent the card's content. | |

### 

### **4. Accessibility**

| Screen Reader Support | **Use ARIA labels for interactive elements to clearly describe their actions, especially for icons or buttons without text. Provide alternative text for images within the card, describing the image's content or function.** |
| --- | --- |
| Semantic HTML | **Use appropriate HTML elements (<article>, <section>, <div>) with proper roles and attributes to ensure that the card content is accessible to screen readers.**  **Use ARIA roles (e.g., role="region", aria-live, aria-label) to describe the carousel and its components for screen readers.** |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

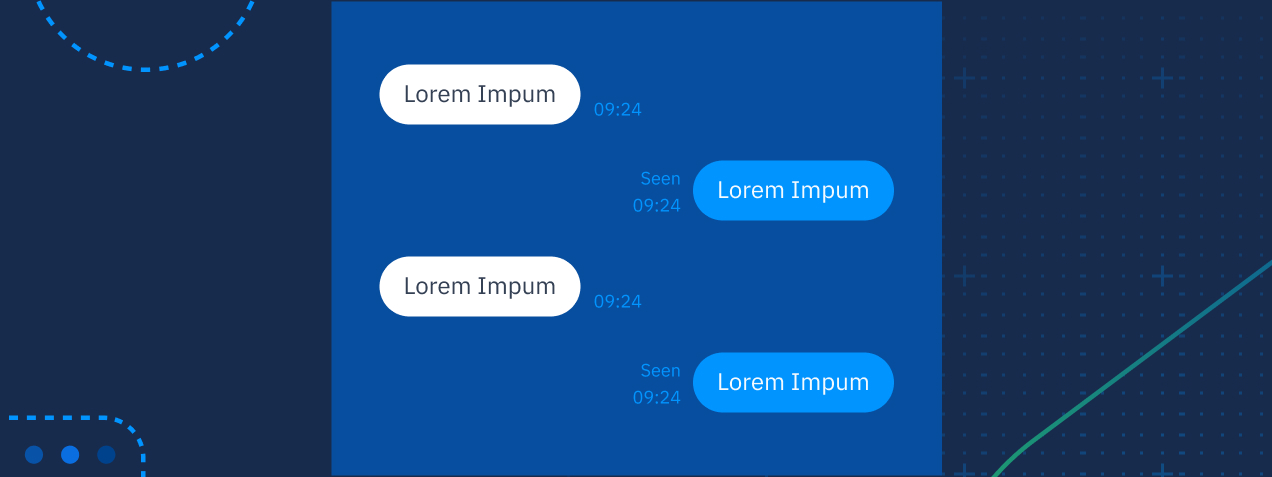
### **6. Related Components**

| Components that are often used together: | Cards often work with components like Images, Buttons, Icons, Badges, Checkboxes, or Menus to support actions like adding to cart, reading more, or editing items. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Lists, Tables, or basic layout Sections, depending on whether you want a continuous flow of content or need to support side-by-side comparisons. |
| illustration: | |
| (Attached pictures of use with other components) | |

### **7. Specification**

|  |
| --- |

## Chat Element



### **1. General Information**

| Component name: | **Chat Element** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Chat Element component is a user interface element designed to facilitate real-time messaging between users. It is commonly used in social media, customer support systems, collaborative workspaces, and any application requiring instant communication. |
| Objective: | When talking about chat element writing, the writer first focuses on the purpose of the writing because it’s the direction of communication, including style, voice, wording, and language usage.  Additionally, the purposes of chat element writing, also known as chatbot messages in user-experience writing, are different based on contexts, such as to inform, ask frequently asked questions, or report results or statistics. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
|  |  |
| Clearly distinguish between messages from different users using visual cues such as color and alignment. | Don’t overload the chat interface with too many features that could distract from the primary messaging function. |
|  |  |
| Ensure the input field is always visible and accessible. | Avoid small or hard-to-click send buttons and attachment icons, especially on mobile devices. |
|  |  |
| Provide clear indicators for message status (sent, delivered, read) to inform users about the delivery state of their messages. | Don’t neglect error handling; Inform users if a message fails to send and provide retry options. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use the Chat component when the system needs real-time, interactive communication between users, support agents, or automated bots to provide seamless and human-like experiences.** |
| --- | --- |
| Should not be used when: | **Avoid using Chat when the interaction is one-way or doesn’t require real-time communication—such as system alerts, passive notifications, or content that’s better structured in lists or cards.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **A good Chat design should clearly distinguish between message senders, support attachments like images or files, and show relevant statuses like “Seen,” “Online,” or “Typing…”** |
| ❌Guidelines to avoid: | **Avoid overly long messages in a single bubble, lack of spacing between messages, and unclear status indicators that might confuse users or harm usability.** |

| illustration: |
| --- |
| Example of Best Practices: |
| **Message Display Area** - A scrollable area where chat messages are displayed. Messages from the user and other participants are styled differently for clarity. |
| **Input Field** - A text input field at the bottom of the chat window where users can type their messages. |
| **Send Button** - A button adjacent to the input field to send messages. Often represented with an icon like a paper plane. |
| **Attachments and Emojis** - Optional buttons for attaching files and inserting emojis enhance the messaging experience. |
| **Typing Indicators** - Visual cues that show when a participant is typing are usually displayed at the bottom of the chat area. |

### **3. Properties & Customization**

| Real-time Updates | **Messages appear instantly in the chat display area as they are sent and received, providing a seamless communication experience.** |
| --- | --- |
| Auto-Scroll | **The chat display area automatically scrolls to show the most recent messages, keeping the conversation in view.** |
| Message Formatting | **Supports basic text formatting, links, emojis, and possibly media previews within messages.** |
| Message Status | **Indicates the status of messages (sent, delivered, read) with appropriate icons or text.** |
| illustration: | |
| **Basic Chat** - Simple text-based chat with essential features like typing, sending, and receiving messages. | |
| **Multimedia Chat** - It supports sending images, videos, files, and text messages. | |
| User Interaction | |
| Typing and Sending | Users type their messages in the input field, click the send button, or press Enter to send. |
| Attachments | Users can click an attachment icon to upload and send files or images. |
| Emojis | Users can insert emojis into their messages using an emoji picker. |
| Message Interaction | Users can interact with messages depending on the application's features (e.g., replying and reacting). |

### **4. Accessibility**

| Keyboard Navigation | **Fully navigable via keyboard, allowing users to type messages, send them, and navigate through the chat history using standard keyboard shortcuts.** |
| --- | --- |
| Screen Reader Support | **They are correctly labeled with ARIA roles and attributes to ensure compatibility with screen readers. Each message should be read clearly, along with any relevant meta information (e.g., sender, timestamp).** |
| Focus Management | **t manages focus correctly, ensuring the input field and send button are easily accessible and have clear indicators for the current focus.** |

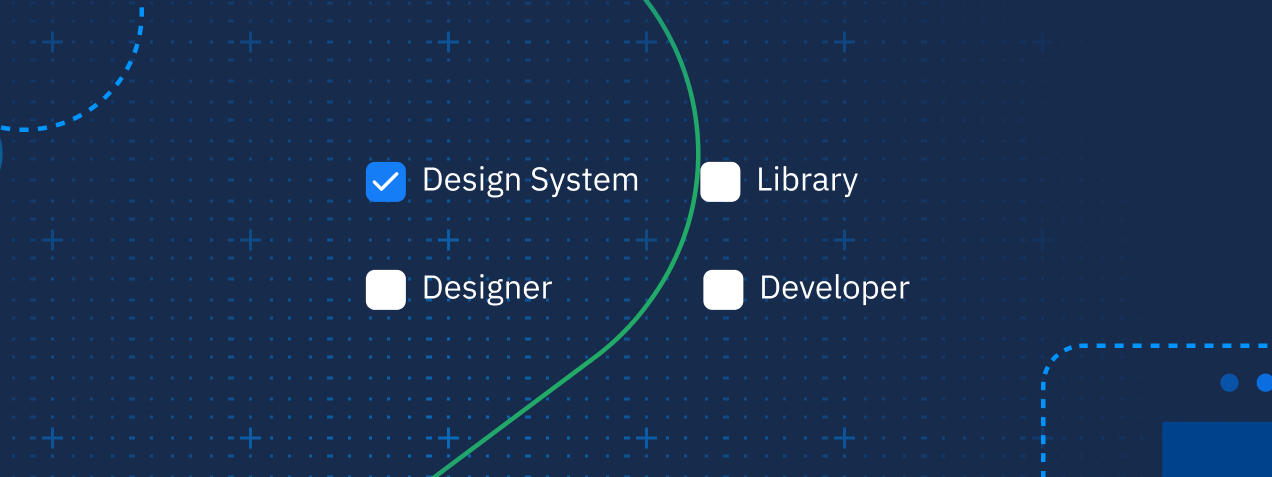
### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Chat works closely with components like Text Input, Buttons, File Uploads, Avatars, Timestamps, Seen Indicators, and Badges for unread message alerts. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Forms for contact, Notifications for one-way messages, or Comment Threads for asynchronous discussions. |
| illustration: | |
|  | |
| The chat element in a customer support application connects users with support agents. Users can type their queries, send them instantly, and receive real-time responses. The chat supports attachments for screenshots and documents; typing indicators let users know when an agent is responding. | |

## Checkbox



### **1. General Information**

| Component name: | **Checkbox** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Checkbox component is a user interface element that allows users to make one or more selections from a set of options. |
| Objective: | When talking about checkbox in user experience design, it’s known as one of the input controls that guides users to take action in the next step. Checkbox functions generate multiple user selection choices. So, the wording in each checkbox position, like topics and choices, might be short keywords.  For checkbox writing, writers should consider the thumbs-up rules of user experience writing: clear and concise. Users can grasp the main idea when reading the context and know what to select correctly. Also, writers consider selecting similar types of wording, such as a noun, phrase, or sentence. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
|  |  |
| Associate each checkbox with its label for easy identification. | Use checkboxes for independent options; consider radio buttons for exclusive choices. |
|  |  |
| Don't use checkboxes for actions that have immediate effect; buttons are more appropriate. | Avoid cluttering the UI with too many checkboxes; consider alternative controls if the list is lengthy. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Checkbox when users can select multiple options from a group, such as preferences, settings, or items in a list where multiple selections are allowed.** |
| --- | --- |
| Should not be used when: | **Avoid using a Checkbox when only one option is allowed, such as gender or satisfaction level. In such cases, use a Radio Button instead.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Use clear labels to describe what each Checkbox does. It should be easy to toggle and support checked, unchecked, and indeterminate states when needed.** |
| ❌Guidelines to avoid: | **Avoid using Checkboxes without clear labels or using them for single-choice scenarios, as it can confuse users about what is expected.** |

| illustration: |
| --- |
| Example of Best Practices: |
| Color - Includes states for default, hover, pressed, disabled, and selected. |
|  |
| Design - They are typically represented by a square box that is either empty or checked. A check mark appears in the box when the option is selected. |
| It is commonly used in forms, filters, and anywhere a binary choice is presented, such as agreeing to terms of service or selecting multiple items from a list. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **No** |
| Can the color be adjusted? | **Primary Color or Success Color** |
| Does it support Responsive Design? | **Yes** |
| User Interaction | |
| Click to toggle | **Users can click a checkbox to select or deselect the option.** |
| Keyboard Navigation | **When focused, checkboxes can be toggled with the spacebar, allowing for keyboard-only navigation.** |
| Label Clicking | **Clicking on the associated label also toggles the checkbox, enhancing usability.** |
| illustration: | |
|  | |
| Standard Checkbox | **The basic form is used for a single option that can be turned on or off.** |
| Indeterminate State | **They represent mixed selection within a group, where some but not all sub-options are selected.** |

### **4. Accessibility**

| Screen Reader Support | Use the <label> element or aria-label attribute to provide accessible labels for screen reader users. Use appropriate ARIA roles (e.g., role="checkbox") and properties (aria-checked) to convey state and functionality. |
| --- | --- |
| Focus Indicators | Ensure checkboxes have clear focus indicators to aid keyboard navigation. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

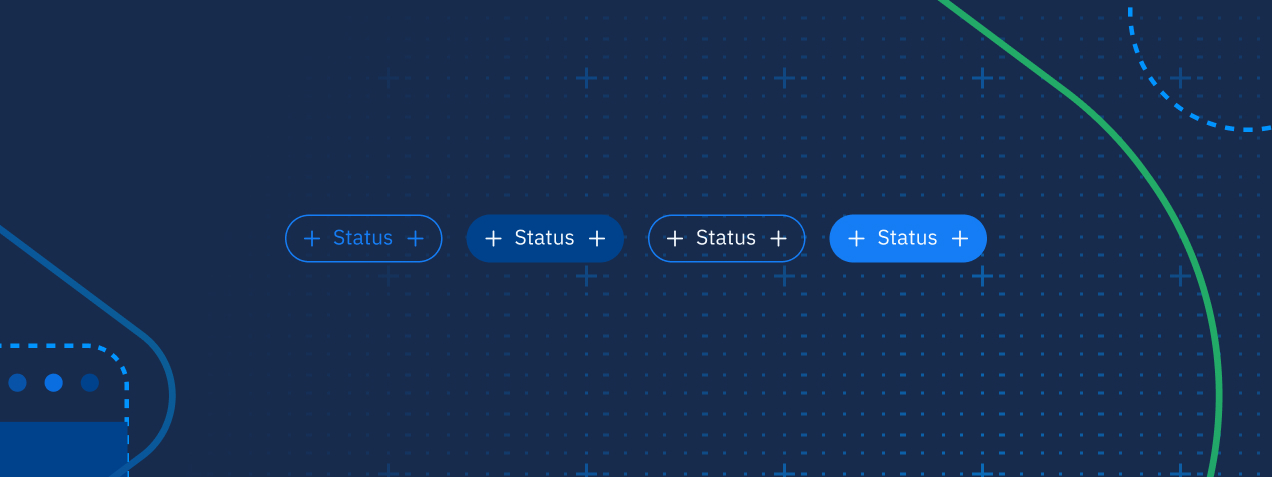
### **6. Related Components**

| Components that are often used together: | Checkboxes are often used with Labels, Fieldsets, Tooltips, Form Validation components, and sometimes with Buttons or Cards for interactive selections. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Toggles for enabling/disabling features, or Multi-Select Dropdowns when space-saving and multiple choices are needed. |
| illustration: | |
|  | |
| In a sign-up form, checkboxes allow users to opt-in to email newsletters and agree to the terms of service. Each checkbox is accompanied by a label that users can click for easy selection. | |

### **7. Specification**

| Spacing |
| --- |
|  |
|  |
|  |
| Properties |
|  |
|  |
|  |
|  |
|  |

## Chips



### **1. General Information**

| Component name: | **Chips** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Chip component is a compact, interactive element used to display information, such as tags, contacts, or actions, in a visually appealing and easily digestible format. |
| Objective: | Chips can be used in various UI contexts to represent small blocks of information or interactive elements that can be selected, removed, or navigated through. |
|  | |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
|  |  |
| ✅Use chips to categorize content, filter options, or enter tags to streamline user interactions. | ❌Don't overcrowd the UI with too many chips; consider using overflow strategies for managing large sets. |
|  |  |
| ✅Keep the chip text concise for clarity and readability. | ❌Avoid using chips for primary actions or navigation where buttons or links would be more appropriate and accessible. |
|  |  |
| ✅Provide a clear indication of the chip's interactivity, such as cursor change on hover. |  |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use the Chip component when you want to display short, easily distinguishable pieces of information such as tags, categories, statuses, or selected items in a form. Chips can be interactive, allowing users to remove or select items.** |
| --- | --- |
| Should not be used when: | **Avoid using Chips when the content is too long, difficult to scan, or cannot be summarized briefly. Also, Chips should not replace primary buttons or components that require complex interaction, like dropdowns or menus.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Chips should use concise, easy-to-understand labels and be sized appropriately for a clean UI. Place them where they can be easily scanned by users. If interactive, such as removable or selectable Chips, ensure clear icons and feedback are provided.** |
| ❌Guidelines to avoid: | **Avoid overloading Chips with too much text or placing too many in the same area, as it creates visual clutter. Don’t use non-descriptive icons, and never use Chips as primary navigation components.** |

| **Variants** | | | |
| --- | --- | --- | --- |
|  | | | |
| **Standard Chips** | Display text and icons for categorizing or tagging. | | |
| **Input Chips** | Allow users to enter and edit information directly within a chip. | | |
| **Choice Chips** | Enable a selection among options, acting similarly to radio buttons. | | |
| **Filter Chips** | Used to filter content; can be toggled on or off to apply or remove filters. | | |
| **Style** | | | |
|  |  |  |  |
| Solid Chip | Light Chip | Outline Chip | Filter Chips |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Small and Medium** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **CI Color and System Color** |
| illustration: | |
| **Shape** - They are presented as a rounded rectangle or pill shape to stand out from other text or button elements. | |
| **Size** - Compact enough to fit multiple chips in a row or stack, yet large enough to be easily interactable. | |
| **Colors** - Utilizes the AXONS color scheme, with distinguishable colors for different states (default, selected, disabled) or types (information, success, warning, error). | |
| **Typography** - It features concise text, often accompanied by an icon or avatar on one side and a delete/remove icon on the other if the chip is removable. | |
| **Icon** - Icons or avatars can visually represent the chip's purpose or category, enhancing user recognition. | |
| **Behavior** | |
| **Interactivity** | Chips can be designed to be clickable, selectable, or removable, with feedback indicated through visual cues (e.g., color change, elevation). |
| **Removal** | If a chip is removable, clicking the delete icon removes the chip from the interface. |
| **Selection** | Selectable chips toggle between selected and deselected states upon user interaction. |
| **Overflow** | In limited space, chips can be contained within a scrollable container or summarized into an expandable chip that displays the number of hidden items. |
| **User Interaction** | |
| **Click Action** | Activates the chip's primary action, such as opening a detail view, toggling its selected state, or any context-specific action. |
| **Removal** | Clicking on a removal icon (typically an "x" or a trash can icon) deletes the chip, potentially triggering a confirmation prompt in sensitive contexts. |
| **Keyboard Navigation** | Supports navigation through arrow keys when part of a collection, with enter and space keys to select or activate the focused chip. |

### **4. Accessibility**

| ARIA Roles and Attributes | Apply appropriate ARIA roles (e.g., role="option" for selectable chips within a set) and states (e.g., aria-selected="true" for selected chips) to convey the component's state and functionality to assistive technologies. |
| --- | --- |
| Keyboard Accessible | Ensure chips are navigable and operable with keyboard controls, including selection and removal actions. |
| Focus Indicators | Provide clear focus styles to indicate when a chip is focused during keyboard navigation. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Chips often work with other components like Text Fields (to display selected values), Filter Panels, Cards, or multi-select Forms. They help clearly display user selections in context. |
| --- | --- |
| Components that can be used interchangeably: | If not using Chips, you can use Badges for displaying statuses or use Dropdowns and Checkboxes when collecting multi-value inputs that don’t need to be shown as separate UI elements. |
| illustration: | |
|  | |
| Chips are used to tag issues with statuses (e.g., "In Progress", "Closed") in a table. Users can click on a chip to filter the issues list based on the selected category. | |

### **7. Specification**

| Size S |
| --- |
|  |
|  |
|  |
|  |
|  |
| Size M |
|  |
|  |
|  |
|  |
|  |
| Variant |
|  |
|  |
|  |
|  |
| State |
|  |
|  |
|  |
|  |
|  |

## Date Input



### **1. General Information**

| Component name: | **Date Input** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Date Input component allows users to input dates directly into a field, facilitating precise and efficient data entry. |
| Objective: | The primary purpose of date input is to guide and facilitate users' selection of dates in specific contexts. Therefore, the pattern of date input contains various types, including long, medium, and short dates. |
|  | |
| It is commonly used in forms for reservations and registrations, and a specific date must be captured anywhere. This component can be enhanced with a date picker for ease of use. | |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
|  |  |
| ✅Provide clear formatting guidelines and placeholder text to aid manual entry. | ❌Avoid using non-standard date formats that can confuse users. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Date Input when users need to select or enter a specific date, such as birthdays, appointments, or date ranges. It ensures accurate and standardized input.** |
| --- | --- |
| Should not be used when: | **Avoid using Date Input if the date is approximate or not critical, such as “sometime next month,” or when the exact date isn’t needed for the task.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **A good Date Input should include a calendar picker, use localized date formats like DD/MM/YYYY, and validate input to catch formatting errors.** |
| ❌Guidelines to avoid: | **Avoid forcing users to type dates without assistance and don’t use unclear or non-localized date formats that might cause confusion.** |

| illustration: |
| --- |
| Example of Best Practices: |
| **Validation State** - It provides visual feedback (such as a border color change) to indicate whether the entered date is valid or invalid. |
| **Input Field** - It is styled to match the application's design system, featuring straightforward, legible typography. The field typically shows a placeholder or format example (e.g., MM/DD/YYYY) to guide user input.  **Icons** - It often includes a calendar icon within or adjacent to the input field to indicate that the field is specifically for data entry. This icon may also be a button to trigger an associated date picker. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **No** |
| Behavior | |
| **Manual Entry** | Users can type a date directly into the input field, adhering to a specified format. |
| **Date Picker Integration** | Optionally integrates with a date picker component, allowing users to select a date from a visual calendar instead of typing. |
| **Validation** | Automatically checks the entered date for validity against a predefined format and logical constraints (e.g., no future dates for birthdate fields). |
| **Formatting** | On focus loss, the component can auto-format the entered date into a consistent presentation style. |
| User Interaction | |
| **Typing** | Users enter dates manually, with the component guiding the format. |
| **Picking** | Clicking on the calendar icon opens the date picker for intuitive date selection. |
| **Error Correction** | If the input doesn't match the expected date format or falls outside acceptable ranges, the component provides immediate feedback, prompting the user to make corrections. |

### **4. Accessibility**

| Keyboard Navigation | Ensures that users can navigate to and interact with the data input using only a keyboard. |
| --- | --- |
| Screen Reader Support | Correctly labels the input field and provides instructions for the expected format, ensuring screen readers can correctly convey how to use the input. |
| Focus Management | Handles focus appropriately, especially when transitioning between typing in the input field and selecting dates from an associated selector. |
| illustration: | |
| **With/Without a Date Picker** - Some implementations may solely rely on manual entry, while others include a date picker for convenience. | |
| **Date Range** - An extension of the primary date input to allow for the selection of a start and end date, often used for booking periods. | |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

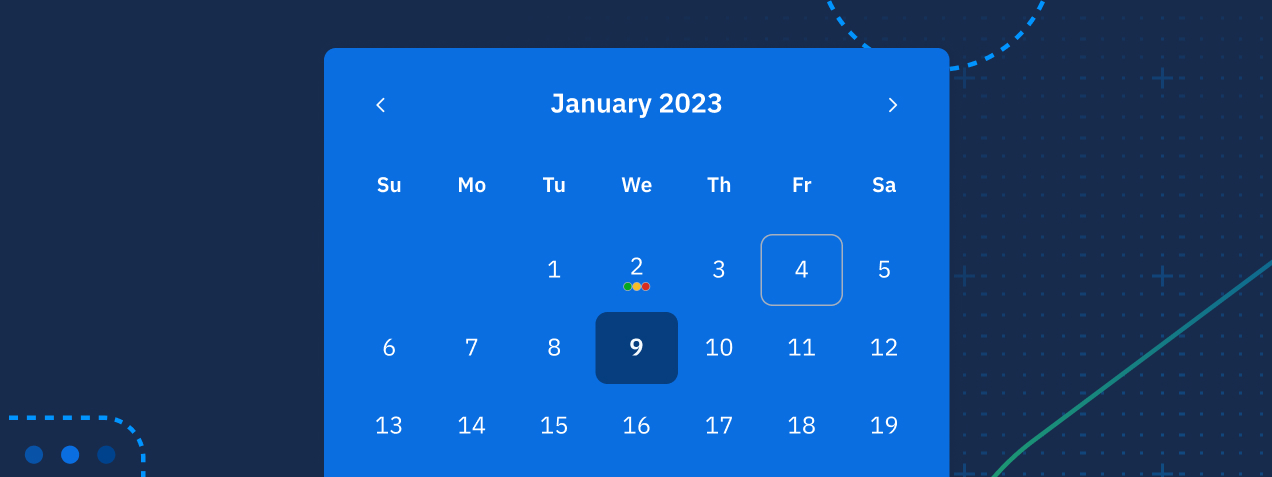
### **6. Related Components**

| Components that are often used together: | Date Input commonly works with Calendar Pickers, Labels, Tooltips, Form Validations, and Dropdowns (e.g., for selecting year or month) to enhance the UX. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include a regular Text Input with date validation, or a Date Range Picker when selecting a span of multiple dates. |

### **7. Specification**

| Spacing |
| --- |
|  |
|  |
|  |
| Properties |
| State |
|  |
|  |
|  |
|  |
| Error |
|  |
|  |
| Disabled |
|  |
|  |

## Date Picker



### **1. General Information**

| Component name: | **Date Picker** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Date Picker component allows users to select a date or a range of dates from a pop-up calendar interface. |
| Objective: | It enhances user experience by providing an intuitive and efficient way to input dates without manual typing. This component is widely used in forms for booking systems, event creation pages, and anywhere date selection is required. |
|  | |
| **Calendar Pop-Up** - A graphical calendar interface appears when the input field is focused or clicked. It includes navigation controls to change months and years. | |
|  | |
| **Visual Cues** - Uses visual highlights to indicate the current date, selected date(s), and any disabled dates (e.g., past dates or dates outside a specified range). | |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
|  |  |
| ✅Indicate the current and selected dates within the calendar. | ❌Don't rely solely on color to convey information; use icons or text to indicate disabled or selected dates. |
|  |  |
| ✅Ensure the calendar pop-up is positioned close to the input field to maintain a clear visual connection. | ❌Avoid complex or non-standard navigation controls that can confuse users. |
|  |  |
| ✅Provide visual feedback and clear error messages if an invalid date is entered. | ❌Don’t assume all users will interact with the date picker similarly; provide alternative text entry if necessary. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Date Picker when users need to select a date from a calendar, such as booking dates, appointments, or expiration dates, to improve accuracy and reduce input errors.** |
| --- | --- |
| Should not be used when: | **Avoid using a Date Picker when the date is vague or not exact, such as “soon,” or when keyboard input is preferred—like in terminal-based or highly technical environments.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **A good Date Picker should display an intuitive calendar, allow easy month/year navigation, support single or range selection, and localize the date format properly.** |
| ❌Guidelines to avoid: | **Avoid overly complex Date Pickers that require many clicks to select a date, and don’t rely on tiny calendar icons without clear labels or instructions.** |

| illustration: |
| --- |
| Example of Best Practices: |
| **Single Date Picker** - Allows users to select a single date. |
| **Range Date Picker** - It enables a start and end date selection and is helpful in booking periods or date ranges. |
| **Time Picker Integration** - Combines date and time selection in a single interface. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **No** |
| Can the color be adjusted? | **No** |
| Behavior | |
| **Date Selection** | Users can select a date by clicking on a day in the calendar pop-up. To select a range, they can click on a start date and an end date. |
| **Navigation** | Includes buttons or controls to switch between months and years. Some implementations may allow quick navigation by clicking on a month or year to select it from a dropdown menu. |
| **Validation** | Can restrict selectable dates based on predefined rules, such as disabling past dates or limiting the selection to weekdays only. |
| User Interaction | |
| **Opening the Calendar** | Triggered by clicking on the input field or an associated calendar icon. |
| **Selecting a Date** | Clicking on a specific day in the calendar sets the date in the input field. |
| **Closing the Calendar** | The calendar closes automatically after a date is selected, or it can be closed manually by clicking outside the calendar area or pressing the Esc key. |
| **Keyboard Accessibility** | Users can navigate the calendar using arrow keys, select a date with Enter, and close the calendar with Esc. |

### **4. Accessibility**

| Keyboard Navigation | It is fully operable via keyboard, with clear focus indicators and support for navigation and selection using keyboard controls. |
| --- | --- |
| Screen Reader Support | Ensures all elements are appropriately labeled and described, providing an accessible experience for screen reader users. |
| ARIA Roles and Properties | This uses appropriate ARIA roles such as role="dialog" for the calendar pop-up and aria-haspopup="true" for the input field. The selected date and current focus are communicated with aria-selected and aria-activedescendant. |
| illustration: | |
| **With/Without a Date Picker** - Some implementations may solely rely on manual entry, while others include a date picker for convenience. | |
| **Date Range** - An extension of the primary date input to allow for the selection of a start and end date, often used for booking periods. | |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Date Pickers typically integrate with Text Inputs, Labels, Calendar UIs, Dropdowns (for month/year), Form Validations, and sometimes Time Pickers. |
| --- | --- |
| Components that can be used interchangeably: | If not using a Date Picker, alternatives include a Text Input with date validation or separate Dropdowns for day, month, and year in simpler use cases. |

## Divider

## 

### **1. General Information**

| Component name: | **Divider** |
| --- | --- |
| Category: | **Display & Feedback Components** |
| Description: | The Divider component serves as a visual tool to separate content or sections within a page. |
| Objective: | It enhances the organization and readability of the interface by clearly delineating different areas, making it easier for users to navigate and understand the layout. Dividers can be used in menus, lists, between paragraphs, or in layouts to create a clear visual distinction. |
| Appearance: | |
| **Orientations** - Dividers can be oriented horizontally, commonly used in lists and between content sections, or vertically to separate sidebars from main content areas. | |
| **Style Variants** - It is available in various styles, including solid lines, dashed lines, or lines with a decorative element (such as text) centered along the line. | |
| **Color** - It is typically rendered in colors subtly contrasting with the background to ensure visibility without dominating the design. We have specific AXONS colors for the outline. | |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
|  |  |
| ✅Use dividers sparingly to avoid cluttering the interface and diluting their effectiveness in separating content. | ❌Don't use dividers to substitute for proper layout or grouping. Use padding and margins for spacing and containers or sections for grouping content. |
|  |  |
| ✅Ensure dividers align with the interface's overall design language and color scheme. | ❌Avoid using overly decorative or eye-catching dividers that might distract from the content. |
|  |  |
| ✅Consider dividers' visual weight in the page layout context to maintain balance and harmony. |  |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Divider when you need to visually separate sections of content, such as between form areas, menu items, or content blocks within the same view.** |
| --- | --- |
| Should not be used when: | **Avoid using Dividers when the content is already clearly separated with spacing, or when overused to the point of visual clutter.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **A good Divider should be thin and subtle, aligning with the layout direction (horizontal or vertical), and spaced appropriately from surrounding content to maintain visual balance.** |
| ❌Guidelines to avoid: | **Avoid thick or brightly colored Dividers that overpower the content, and don’t stack multiple Dividers close together unnecessarily.** |

| Behavior | |
| --- | --- |
| Static Component | Dividers are static components that do not involve user interaction. Their primary role is to enhance the visual layout and organization of content. |
| Responsive | Adapts to different screen sizes and orientations, ensuring that dividers maintain their purpose across devices by adjusting their length or orientation as necessary. |

| User Interaction | |
| --- | --- |
| N/A | The Divider component does not involve direct user interaction but plays a crucial role in structuring content in a user-friendly manner. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **0.5px, 1px, 1.5px, and 2px** |
| --- | --- |
| Icon supported or not: | **Text support only** |
| Can the color be adjusted? | **Outline color only** |
| Does it support Responsive Design? | **horizontal and vertical** |
| illustration: | |
|  | |

### **4. Accessibility**

| Semantic HTML | When applicable, use the <hr> element for horizontal dividers, which semantically represent a thematic break between paragraph-level elements. Appropriate ARIA roles or attributes should be used for decorative or non-semantic dividers to ensure they are not mistakenly read as meaningful by screen readers. |
| --- | --- |
| Visual Contrast | Ensure that the divider has sufficient contrast with the background to be visible to all users, including those with visual impairments, without being overly distracting. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

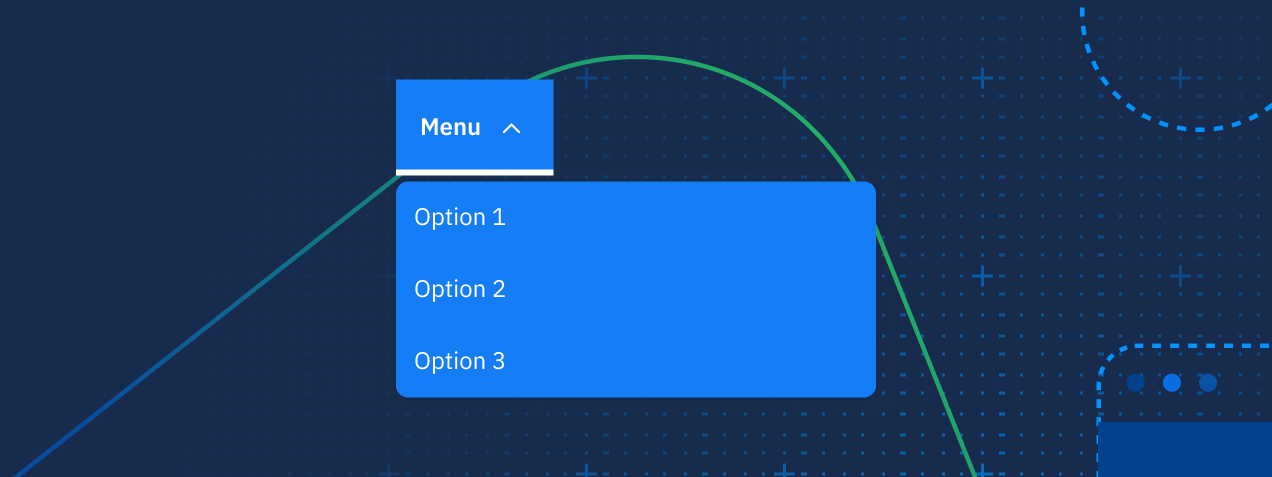
### **6. Related Components**

| Components that are often used together: | Dividers often work with Cards, Lists, Menus, Form Sections, and Typography to improve content structure and scannability. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include using spacing (like margin or padding), background color variations, or bordered Boxes/Sections to create visual separation. |
| illustration: | |
|  | |
| In an information page, a horizontal Divider separates the user's bio section from their posted content, creating a clear distinction that helps users navigate the data intuitively. | |

### **7. Specification**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

## Dropdown



### **1. General Information**

| Component name: | **Dropdown** |
| --- | --- |
| Category: | **Navigation Components** |
| Description: | The Dropdown component is a graphical control element that allows users to choose one value from a list. |
| Objective: | Allow users to select one or more options from a hidden list, saving space and keeping the UI clean and minimal. |
| Appearance: | |
|  | |
| When inactive, it displays a single value. Upon interaction, it expands to show a list of options. This component is widely used in forms and UIs to save space and simplify the selection process for items like settings, filters, or forms.  **Visual Design** - Typically, it features an area or object to indicate its expandability. The design is consistent with the application's style guide, utilizing brand colors, fonts, and spacing. | |
|  | |
| **List Items** - Options within the dropdown are listed vertically. Selected items are highlighted or marked with a check. | |
|  | |
| **States** - Visual cues for different states (default, hover, active, disabled) enhance usability and guide user interaction. | |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
|  |  |
| Limit the number of options to avoid overwhelming the user. Consider sub-categories or a search function for long lists. | Don't use dropdowns for actions or navigation. Use buttons or links instead. |
|  |  |
| Ensure the dropdown and all its options are accessible on all devices, especially for touch interactions on mobile. | Avoid deeply nested dropdowns, which can confuse users and create navigation issues. |
|  |  |
|  | Don't clutter the UI with too many dropdowns close together, which can overwhelm users. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Dropdown when users need to choose from a medium to large list of options—such as selecting a country, language, category, or filter—especially when screen space is limited.** |
| --- | --- |
| Should not be used when: | **Avoid using a Dropdown when there are very few options (e.g., 1–3) or when all choices should be visible at once, such as gender or satisfaction level.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Use clear labels, descriptive placeholders, and sort the list in a logical way. Ensure the Dropdown supports keyboard navigation for better accessibility.** |
| ❌Guidelines to avoid: | **Avoid unordered or excessively long lists without filtering. Don’t use Dropdowns for real-time dynamic data without preloading or loading indicators.** |

| illustration: |
| --- |
| Example of Best Practices: |
|  |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal in websites but in mobile apps use the native os.** |
| --- | --- |
| Icon supported or not: | **Yes, front text and back text.** |
| Can the color be adjusted? | **No** |
| Behavior | |
| Expand/Collapse | Clicking on the dropdown header toggles the visibility of the list. Some implementations close the dropdown when an item is selected or if the user clicks outside the component. |
| Selection | Users can select an option by clicking on it, which updates the header to reflect the chosen value. |
| Keyboard Navigation | Supports navigation through the list using keyboard inputs (arrow keys, Enter to select, Esc to close). |
| Scrollable | Long lists within the dropdown can be scrollable, allowing users to navigate many options without expanding the dropdown beyond the viewport. |
| User Interaction | |
| Opening | It is activated by clicking or tabbing into and pressing Enter/Space. The list of options appears. |
| Choosing an Option | Selections are made by clicking with a mouse, navigating with keyboard keys, and pressing Enter. |
| Closing | The dropdown collapses when an option is selected, or the focus is moved away (clicking outside the dropdown or pressing Esc). |

### **4. Accessibility**

| Keyboard Accessibility | Ensure the dropdown can be entirely operated with the keyboard alone. |
| --- | --- |
| Screen Reader Support | Use ARIA roles and attributes to ensure the dropdown's purpose and state are communicated to users of assistive technologies. For example, role="listbox", role="option", aria-expanded, and aria-selected. |
| Focus Management | Properly manage focus when navigating the options to ensure a seamless experience for keyboard users. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

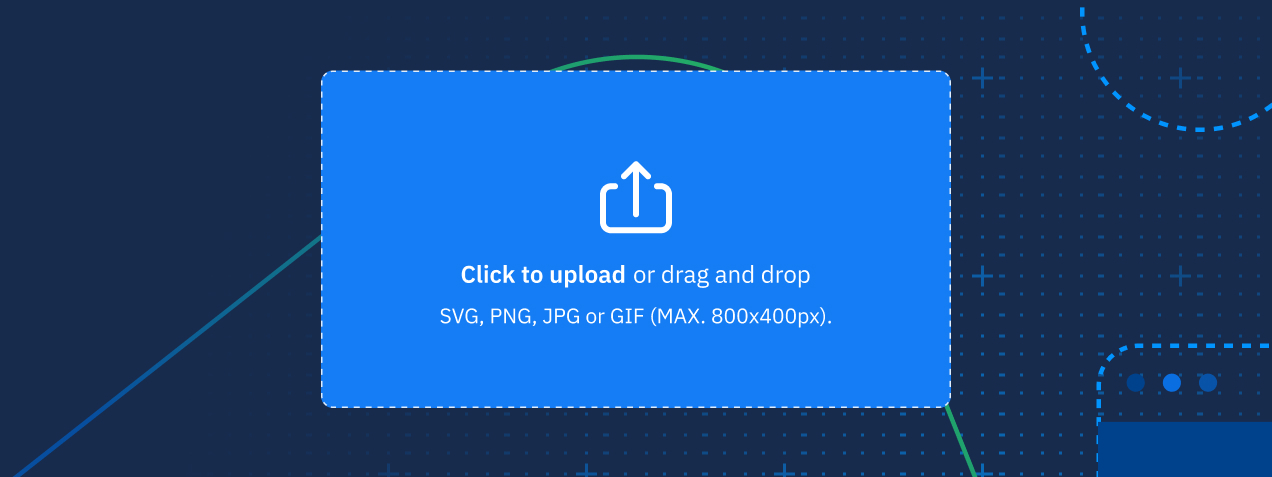
### **6. Related Components**

| Components that are often used together: | [Dropdown, Autocomplete, Icon Button, etc.] |
| --- | --- |
| Components that can be used interchangeably: | [Other options that may be more appropriate in some cases] |
| illustration: | |
|  | |
| Single select allows the user to pick a single option from the dropdown list. | |
|  | |
| illustration: | |
| A dropdown in a user profile form selects the user's country of residence. Upon clicking the dropdown, a list of countries appears. The user selects their country and then updates the dropdown's header to show the country chosen, streamlining the form-filling process. | |

### **7. Specification**

|  |
| --- |
|  |

## File Upload



### **1. General Information**

| Component name: | **File Upload** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The File Upload component allows users to select and upload files from their device to a web application or server. |
| Objective: | The objective of File Upload is to allow users to select and transfer files from their devices into the system, such as documents, images, videos, or data files, for further processing. |
| Appearance: | |
|  | |
| It is commonly used in forms for document submission, profile picture updates, and file attachments in email services. This component simplifies transferring files and ensures compatibility and security in file handling. | |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
|  | (Attached image of poorly designed UI Component) |
| ✅Indicate allowed file types and size limits to users before they select files. |  |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use File Upload when users need to submit files as part of a process, such as attaching documents in an application form, uploading profile pictures, or submitting contracts.** |
| --- | --- |
| Should not be used when: | **Avoid using File Upload if file submission isn’t necessary, such as text-only forms or when structured input is better suited than unstructured file uploads.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Clearly communicate file requirements, like supported file types, size limits, and quantity restrictions. Provide visual feedback, such as upload progress bars or filenames after selection.** |
| ❌Guidelines to avoid: | **Avoid letting users upload files without clear instructions or failing to show validation errors, such as exceeding size limits or uploading unsupported file types.** |

| Variants | |
| --- | --- |
| Basic File Upload | A simple click-to-select file uploader. |
| Advanced Upload | Drag-and-drop file uploader. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Muti-size** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **Yes** |
| Does it support Responsive Design? | **Yes** |
| illustration: | |
|  | |
| **Input Field** | Typically, it includes a button or a drag-and-drop area that users interact with to select files from their device. It might display the selected file's name or a list of selected files. |
| **Icons and Labels** | Utilizes clear, descriptive icons and text labels (e.g., "Choose File," "Upload") to guide users through the file selection and upload process. |
| Behavior | |
| **File Selection** | Users can select files by clicking on the upload area to open the file browser or dragging and dropping files into a designated area. |
| **Multiple File Uploads** | The component can be configured to accept multiple files simultaneously if the use case requires it. |
| **Validation** | Performs client-side validation on selected files based on criteria like file size, type, and number of files. |
| **Asynchronous Upload** | Files are typically uploaded in the background, allowing users to continue interacting with the application during the upload process. |
| **User Interaction** | |
| **Activating the File Browser** | Clicking on the file upload area opens the device's file browser, where users can navigate and select files. |
| **Drag-and-Drop** | Users can drag files from their device and drop them into the upload area, triggering the upload process. |
| **Feedback on Completion** | Once the upload is complete, the user receives visual feedback, such as a success message or the display of the uploaded file. |

### **4. Accessibility**

| Keyboard Accessibility | Ensure the file upload component is operable with keyboard shortcuts, including opening the file dialog and triggering the upload. |
| --- | --- |
| Screen Reader Support | Properly label the file upload components with ARIA attributes so screen readers can accurately describe the action and status. |
| Focus Management | Manage focus throughout the interaction, especially for error handling and indicating upload progress. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | File Upload typically works with Buttons (for file selection), Drag-and-Drop Areas, Progress Bars, Toast Notifications (for success or failure), and Validation Messages. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include allowing users to submit links (URLs), uploading files directly from cloud storage (Google Drive, Dropbox), or using connected online forms that auto-link files. |
| illustration: | |
|  | |
| The File Upload component is crucial for enabling efficient, user-friendly file selection and upload processes in web applications, ensuring users can easily and securely transfer files as needed. | |

## From Control



### **1. General Information**

| Component name: | **From Control** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Form Control component encompasses various elements used within a form to capture user input, including text Inputs, text areas, checkboxes, dropdowns, and more. |
| Objective: | The objective of Form Control is to structure and manage form inputs like text fields, checkboxes, and dropdowns consistently, supporting labels, helper texts, and error messages to improve usability and clarity. |
| Appearance: | |
| Label | |
| Hint | |
| Uniformity | Designed with a consistent look and feel that aligns with the application's design system, ensuring that all form controls are visually cohesive. |
| Labels | Each form control includes a clear, descriptive label near the input field to inform users about the expected input. |
| Validation Feedback | It incorporates visual cues (e.g., color changes, icons, and helper text) to indicate field validation status (error, success, warning). |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Design with accessibility in mind, ensuring all users can interact with each form control. | ❌Don't clutter forms with unnecessary fields; keep them concise to improve user engagement. |
|  |  |
| ✅Provide clear, actionable error messages to help users correct mistakes. | ❌Avoid using placeholder text as the only labeling method; always include visible labels. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use Form Control when building forms with multiple inputs, such as registration forms, product booking forms, or surveys, to manage the structure and presentation of each field consistently.** |
| --- | --- |
| Should not be used when: | **Avoid using Form Control if the form is extremely simple, with only one or two basic inputs, or when there’s no need for separate labels, errors, or helper texts.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Each Form Control should have a clear label, optional helper text for special instructions, and error messages that appear immediately upon validation errors to guide users effectively.** |
| ❌Guidelines to avoid: | **Avoid omitting labels or error messages, and don’t scatter inputs without structure, as it leads to a confusing and hard-to-use form.** |

| Variants | |
| --- | --- |
| **Text Input** - For single-line text input. | |
| **Text Areas** - For multi-line text input. | |
| **Select** - For selections and choices. | |
| **Date Input** - For selections and choices. | |
| **Autocomplete** - For searching and selecting from a list of options. | |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **No, Just have the information icon and remark (\*)** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **No** |
| User Interaction | |
| **Entering Data** | Users can enter or select data within each form control according to its type, with the interface responding appropriately to the input method (keyboard, mouse, touch). |
| **Navigating Through Fields** | Supports efficient navigation through the form, allowing users to move between fields using tab keys or touch gestures. |
| **Correcting Errors** | When errors are detected, users are provided clear instructions on correcting them, usually through error messages displayed near the offending input. |
| Behavior | |
| **Input Types** | It supports various input types, each optimized for the data collection type (e.g., text for names, select for selections). |
| **Validation** | It includes real-time validation, providing users with immediate input feedback, and guiding them in correcting errors before submission. |
| **Focus Management** | Prioritizes logical tab order and focus management to ensure easy navigation through the form, especially for keyboard and screen reader users. |

### **4. Accessibility**

| Semantic HTML | Utilizes semantic HTML elements for all form controls to ensure proper accessibility, including <input>, <select>, <textarea>, and <button>. |
| --- | --- |
| ARIA Roles and Properties | Applies appropriate ARIA roles and properties to enhance the accessibility of custom controls, ensuring they are accessible to users with disabilities. Ensures every form control is properly labeled with a <label> element or ARIA labels, clearly describing each input's purpose. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Form Controls typically integrate with Text Inputs, Dropdowns, Radio Buttons, Checkboxes, Labels, Helper Texts, and Validation Messages to build complete, user-friendly forms. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include using general layout components like Grids or Stacks, but you’ll need to manually manage labels, errors, and helper texts, making the form more complex to maintain.may be more appropriate in some cases] |
| illustration: | |
|  | |
| In a registration form, various form control components are used to capture the user's name (text field), gender (radio buttons), interests (checkboxes), country of residence (dropdown), and bio (text area). Each field includes real-time validation to ensure the data is entered correctly, improving the overall quality of the submitted information. | |

### **7. Specification**

| Default |
| --- |
| Required |
| Optional |
| info icon |

## Icon Button



### **1. General Information**

| Component name: | **Icon Button** | |
| --- | --- | --- |
| Category: | **Interactive Components** | |
| Description: | The Icon Button component is a user interface element that combines the compact visual appeal of an icon with the functionality of a button. | |
| Objective: | The objective of an Icon Button is to allow users to trigger an action using only an icon instead of text—such as delete, share, or settings—offering a compact and visual alternative to labeled buttons. | |

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Use easily recognizable and standard icons across UIs to ensure users quickly understand the button's function. | ❌Don't overcrowd your UI with too many icon buttons to avoid overwhelming users. |
|  |  |
| ✅Include text alternatives or tooltips that appear on hover or focus to explain the button's purpose for accessibility and clarity. | ❌Avoid using icon buttons for critical actions without accompanying text labels unless the icon's function is universally understood. |
| ✅Ensure the clickable area is large enough for easy interaction but not so large that it interferes with surrounding UI elements. | ❌Don't rely on color alone to indicate button states, considering users who may have difficulty distinguishing colors. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use Icon Buttons when users need to take quick actions and the icon is universally recognizable—like a trash can (delete), pencil (edit), or gear (settings)—especially in toolbars or compact mobile layouts.** |
| --- | --- |
| Should not be used when: | **Avoid using Icon Buttons when the icon’s meaning is unclear or if the action is critical and not easily understood without a text label—like using only a “lightning” icon without context.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Use standard, easily understood icons with proper aria-labels or tooltips for accessibility, especially when no visible label is present. Place them in intuitive, contextually appropriate positions.** |
| ❌Guidelines to avoid: | **Avoid ambiguous icons or placing many Icon Buttons too close together without spacing or context, as this can confuse users and increase the chance of misclicks.** |

| Behavior | | | |
| --- | --- | --- | --- |
| Visual and tactile feedback on interaction, such as color change or ripple effect, to confirm user action. | | | |
|  | | | |
| **Action State** | Default | Description: | The default state is how the button appears under normal, un-interacted conditions. |
| Appearance: | Solid, Outline, or Ghost style, depending on the variant. |
| User Interaction: | The button is clickable and fully functional. |
| Best Practices: | Maintain consistent styling across similar buttons for uniformity. |
| Hover | Description: | The hover state is activated when the user's cursor is over the button. It provides visual feedback that the button is interactive. |
| Appearance: | Slight visual change, such as a color shift, brightness change, or border thickening. For Ghost Buttons, an underline or color change can be effective. |
| User Interaction: | Indicates to users that the button is clickable. |
| Best Practices: | When signaling interactivity, ensure the hover state is noticeably different from the default state. Avoid drastic changes that might confuse the user. |
| Pressed | Description: | The active state signifies the user has clicked or is pressing the button. |
| Appearance: | It often includes a more pronounced color change, giving the impression of being pressed down. It may also feature a shadow effect or an animation like a ripple. |
| User Interaction: | This state is usually brief, occurring during the click before an action is initiated. |
| Best Practices: | The active state should provide immediate, tangible feedback to reinforce the user's action.  The visual effect should be subtle yet satisfying, contributing to a responsive user experience. |
| Disabled | Description: | The disabled state shows that the inactive button cannot be interacted with. |
| Appearance: | Typically, the button appears faded or has a lower contrast. Icons may also appear muted, and the cursor may change to a 'not-allowed' symbol on hover. |
| User Interaction: | The button does not respond to clicks or any form of interaction. |
| Best Practices: | Use this state to indicate that certain conditions must be met before the button can be activated.  Ensure it's visibly distinct from the active states to prevent user confusion. |
| **Loading State** |  | Description: | An optional loading state with a spinner for actions that require processing time. |

| Anatomy | |
| --- | --- |
|  | |
|  | |
|  | |
| Variants | |
| Solid Button | Features a solid background color that fills the button shape, increasing visibility. |
| Outline Button | Includes a border around the icon, offering a more defined button look. |
| Ghost Button | The Ghost Button is no background until hovered, focusing attention on the icon itself. |
| User Interaction | |
| Click Action | Triggers a predefined action or event in the application. |
| Disabled State | Indicates when a button is inactive or unavailable for interaction. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Small / Medium / Large** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **Brand CI Color, Functions Color and Neutral Color** |
| Does it support Responsive Design? | **Yes** |

### 

| Appearance: | | |
| --- | --- | --- |
| Adaptable to various themes while maintaining sufficient contrast for accessibility. | | |
|  | | |
|  | | |
| Colors | | |
| **Brand CI colors** | Description: | Brand CI colors are primary colors derived from the company's branding palette. These colors are a visual representation of the brand identity. |
| Appearance: | Vibrant and distinctive, aligning with the AXONS brand palette. |
| **Functional colors** | Description: | Functional colors indicate the nature or status of an action or element, such as success, warning, or error. |
| Usages: | Applied to highlight specific actions like 'Submit' (success/green), 'Warning' (yellow), or 'Error' (red). |
| Appearance: | The color corresponds to the function: green for success, yellow for warning, red for error, blue for information, etc. |
| **Neutral Color** | Description: | Neutral colors are unsaturated colors like grey and off-white. They are versatile and less visually demanding. |
| Usages: | They are typically used for secondary or tertiary buttons where less emphasis is needed. |
| Appearance: | Subtle and understated, it is often used for button backgrounds or borders in secondary and tertiary buttons. |
| Iconography | | |
|  | | |
| We can include relevant icons to enhance user understanding (e.g., a plus icon for an 'Add' button). | | |
| Shape | | |
|  | | |
| Rounded corners that align with AXONS design language, promoting a friendly and approachable interface. | | |

### 

### **4. Accessibility**

| Keyboard Navigation | Fully accessible via keyboard, with focus indicators. |
| --- | --- |
| Screen Reader Support | Properly labeled with ARIA attributes for assistive technologies. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Icon Buttons often work with Tooltips, Badges (notifications), Dialogs/Modals (triggered on click), Menus, and sometimes with Text Buttons inside Toolbars or Card Actions. |
| --- | --- |
| Components that can be used interchangeably: | If not using an Icon Button, alternatives include Text Buttons or Buttons with both icons and labels, which are better when clarity is needed—especially for new or unfamiliar users. |
| Example Usage | |
|  | |
| In a photo editing application, an Icon Button with a trash can icon lets users delete selected items quickly. Hovering over the button shows a tooltip saying "Delete," and clicking it removes the item from the workspace. | |

### **7. Specification**

| Spacing | |
| --- | --- |
| Size S | |
|  | |
|  | |
| Size M | |
|  | |
|  | |
| Size L | |
|  | |
|  | |
| Properties | |
| Variant | |
|  | |
|  | |
|  | |
| State | |
|  | |
|  | |
|  | |
|  | |
|  | |

## Loading Sprinner



### **1. General Information**

| Component name: | **Loading Sprinner** |
| --- | --- |
| Category: | **Display & Feedback Components** |
| Description: | The Loading Spinner component is a visual indicator used to communicate to users that a process is ongoing and that they should wait for it to complete. |
| Objective: | To indicate that the system is processing or loading data and not yet ready, helping users understand that they need to wait and preventing repeated actions. |
| Appearance: | |
| Colors | |
| Size | |
| **Visual Design** | Typically circular, featuring a rotating animation that gives the illusion of motion. The spinner can be a simple ring, segmented circle, or any creative variation that fits the design language of the application. |
| **Colors** | Often uses the primary or accent color of the application's color palette to maintain consistency and visibility. |
| **Size** | Available in various sizes to accommodate different spaces and visibility needs, from small spinners in buttons to large ones for page loads. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Use loading spinners for processes that take more than a few seconds to provide feedback to users. | ❌Don't overuse spinners for instantaneous or quick actions, as they may cause unnecessary flicker or visual disturbance. |
|  |  |
| ✅Ensure the spinner is visually distinct and visible against the background. | ❌Avoid blocking user actions unnecessarily; allow users to interact with other application parts if the loaded operation does not affect them. |
|  |  |
| ✅Keep the design simple and avoid overly complex animations that might distract or mislead users. | ❌Don't forget to remove the spinner immediately after the loading process finishes to avoid confusion. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Loading Spinner when fetching real-time data, loading new views, or processing tasks that take a few seconds (typically under 5–10 seconds).** |
| --- | --- |
| Should not be used when: | **Avoid using a Spinner for long or uncertain wait times without additional feedback. For lengthy operations like large file uploads, use progress bars or detailed indicators instead.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Position the Spinner near the content being loaded—like centered in a section or on a button—and optionally include loading text or skeleton UI to provide context.** |
| ❌Guidelines to avoid: | **Avoid showing a Spinner with no time limit or without status updates, as it may lead users to think the system is broken or frozen.** |

| illustration: |
| --- |
|  |
| A loading spinner on a web application dashboard appears when a user requests to view a large dataset. The spinner activates while the data is fetched and processed, disappearing once the content is ready to be displayed. This keeps users informed about the status of their requests, enhancing their overall experience. |

| Behavior | |
| --- | --- |
| Continuous Rotation | The spinner rotates continuously without interruption to signify ongoing activity. |
| Responsive | Adjusts its size and performance based on the device and screen size, ensuring it looks good and functions nicely across all platforms. |
| Visibility | It appears during the loading process and disappears once the process is complete. It should not block user interaction with other elements unless necessary. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Dynamic Size** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **Yes** |
| illustration: | |
| **Fullscreen Spinner** - It covers the entire screen or a large section for significant loading processes requiring user attention. | |
| **Inline Spinner** - It is smaller and positioned next to an element or within a button, indicating the loading of specific components or data. | |
| User Interaction | |
| **Non-interactive** | Spinners are purely informational and do not involve direct user interaction. |
| **Contextual Use** | **They are used during tasks where user patience is required, effectively reducing frustration associated with waiting times.** |

### **4. Accessibility**

| Visually Hidden Text | Include visually hidden text within the spinner element that explains the ongoing action, like "Loading..." or "Please wait," which can be read by screen readers. |
| --- | --- |
| Prevent Focus | Ensure that the spinner is not focusable, as it does not require interaction. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Spinners commonly work with Buttons (inline loading), Modals/Dialogs, Cards, Skeleton Loaders, Notifications, or Overlays to indicate loading states at different UI levels. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Progress Bars (for predictable durations), Skeleton Screens (to represent loading content), or loading text such as “Fetching data…” for better clarity. |
| illustration: | |
|  | |
|  | |

## Modal

## 

### **1. General Information**

| Component name: | **Modal** |
| --- | --- |
| Category: | **Display & Feedback Components** |
| Description: | The component captures user attention by presenting critical information or interactive content in a focused manner. It temporarily pauses the main workflow, requiring users to interact with the modal content before returning to the original page. |
| Objective: | To present important information or actions that require immediate user attention, shown in a layer over the main content to capture user focus. |

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| Use modal for critical interactions. | Overuse modal for non-essential information. |
|  |  |
| Make sure the modal is easily dismissible. |  |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Modal when you need users to complete an important action before continuing, such as confirming deletions, filling out short forms, or acknowledging critical messages.** |
| --- | --- |
| Should not be used when: | **Avoid using Modals for non-urgent content, complex multi-step interactions, or stacking multiple modals, as it can overwhelm or confuse users.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **A good Modal should include a clear title, concise content, and distinguishable actions like “Confirm” and “Cancel.” It should also support dismissal via ESC key or clicking outside the Modal.** |
| ❌Guidelines to avoid: | **Avoid long scrollable content or too many confusing actions within a Modal. Don’t use a Modal just to notify—use a Toast or Banner instead for passive messages.** |

| illustration: |
| --- |
|  |
| The Dialog Modal typically features a prominent header, a central content area, and a footer with action buttons. It should overlay the main content and include a semi-transparent backdrop to maintain focus on the dialog. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Small / Medium / Large** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **Color adjustment conditions only the Function Color and Brand CI Color.** |
| Does it support Responsive Design? | **Yes** |
| illustration: | |
|  | |
| **The component should be responsive, adapting to various screen sizes and orientations. It should not scroll with the page but remain centered.** | |

### **4. Accessibility**

| Accessibility | Ensure the dialog is correctly labeled with ARIA roles such as role="dialog" and described by aria-describedby. Focus management should transfer to the modal when opened. (Optional) |
| --- | --- |
| Keyboard Navigation | The modal should be navigable via the keyboard, with the focus trapped within the modal content until dismissed. Standard keys include Tab, Shift + Tab, and Escape. (Optional) |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

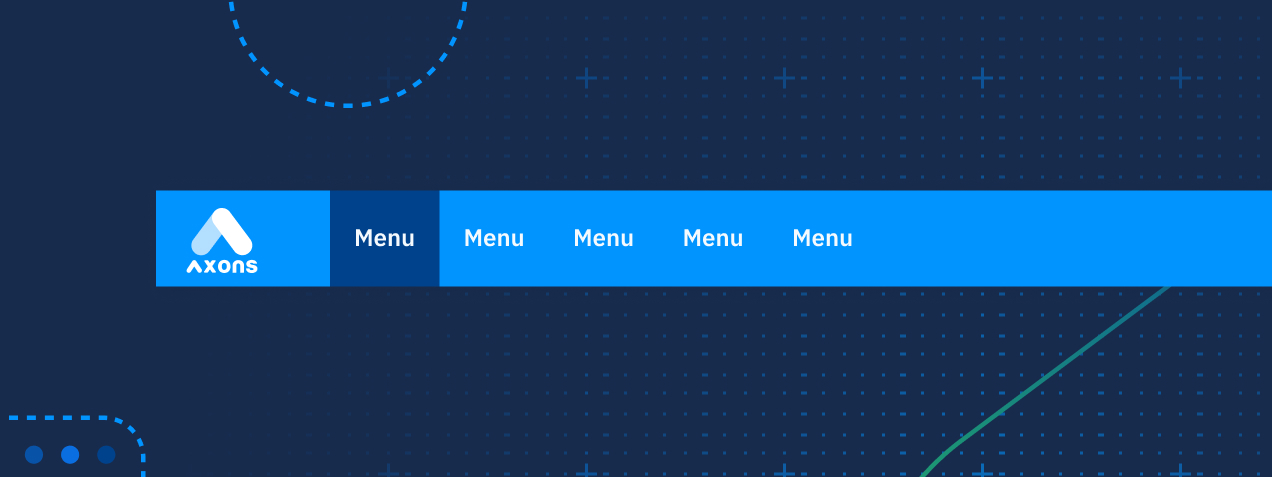
### **6. Related Components**

| Components that are often used together: | Modals typically work with Buttons (to trigger), Form Inputs, Icons, Validation Messages, and Overlays to manage user interaction inside the modal window. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Drawers (side panels), Inline Expansions, Toasts, or Popovers—depending on the urgency and nature of the content. |
| illustration: | |
|  | |
| A typical use case is a confirmation dialog, where users are prompted to accept or reject changes before proceeding. | |

### **7. Specification**

|  |
| --- |
|  |

## Navigation Bar



### **1. General Information**

| Component name: | **Navigation Bar** |
| --- | --- |
| Category: | **Navigation Components** |
| Description: | The Navbar (Navigation Bar) component is a crucial interface element typically located at the top of web applications and websites. |
| Objective: | The objective of a Navigation Bar is to help users quickly access key pages or sections of an application or website. It serves as the main navigational structure for user orientation. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| * Keep navigation simple and intuitive, with clearly labeled links and a logical structure. * Ensure the navbar is responsive and functioning effectively across all devices and screen sizes. * Provide visual cues for interactive elements and current page indicators to help users understand their location within the application. | * Don't overload the navbar with too many links or options, which can overwhelm users and make navigation difficult. * Avoid complex dropdown menus that are difficult to navigate on mobile devices. * Don't sacrifice accessibility for aesthetics; ensure all users can navigate the site effectively. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Navigation Bar when your product includes multiple pages or sections that users need to access frequently—such as web apps, dashboards, or content-rich websites.** |
| --- | --- |
| Should not be used when: | **Avoid using a Navigation Bar in single-page flows with no menu options, or when a minimal experience is desired—such as onboarding screens or short landing pages.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **A good NavBar clearly displays key navigation items, uses intuitive icons or labels, highlights the active section, supports responsive layouts, and allows keyboard accessibility.** |
| ❌Guidelines to avoid: | **Avoid overcrowding the NavBar with too many items, using unclear labels or ambiguous icons, and nesting important links too deeply without clear breadcrumbs or cues.** |
| Behavior: | |
| **Static or Fixed** | This can be designed to remain visible at the top of the screen as users scroll down (fixed) or scroll away with the page (static). |
| **Expandable Sections** | This may include dropdown menus or expandable sections for complex site navigation structures. |
| **Accessibility Features** | Implements ARIA roles and attributes to enhance accessibility, ensuring all navigation links are accessible via keyboard and readable by screen readers. |

| illustration: |
| --- |
|  |
| It provides users with easy access to the main areas of the site or application, often including navigation links, branding elements, and other functional items such as search bars or login controls. |

| Appearance |
| --- |
| **Layout** - It is usually displayed as a horizontal bar across the top of the screen. It can contain the brand's logo, primary navigation links, secondary information, and interactive elements like search fields or dropdown menus. |
| **Responsive** - Adapts to different screen sizes, often transforming into a hamburger menu or similar compact form on smaller devices to save space and maintain usability. |
| **Visual Consistency** - It is styled to match the application's or website's overall design aesthetic, using consistent colors, typography, and spacing. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Desktop, Tablet and Mobile Size** |
| --- | --- |
| Icon supported or not: | **Yes, but in still in AXONS Template** |
| Can the color be adjusted? | **No just 2 Color : Primary color template and White Style template** |
| Does it support Responsive Design? | **Yes** |
| illustration: | |
| **Primary Style and White Style** | |
| **Full Navbar** - They are often used in large websites with extensive content. | |
| **Mini Navbar** - | |
| User Interaction: | |
| **Navigation Links** | Users can click or tap on navigation items to move to different sections or pages. Hover effects often highlight interactive elements. |
| **Responsive Toggle** | On smaller screens, users can toggle the navbar visibility (e.g., expanding a hamburger menu) to access full navigation options. |
| **Search Functionality** | Users can interact with the search bar to input queries directly from the navbar if included. |

### **4. Accessibility**

| Keyboard Navigation | All navigation items can be accessed using the tab and arrow keys on the keyboard alone. |
| --- | --- |
| Screen Reader Support | Uses proper semantic HTML and ARIA roles to ensure elements are announced correctly by screen readers. For example, using role="navigation" for the main navbar area and aria-expanded for dropdowns. |
| Focus Management | Manages focus for interactive elements, providing visible focus states to aid users navigating via keyboard. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | NavBars typically integrate with components like Logos, Icon Buttons (e.g., hamburger menu), Dropdown Menus, Search Fields, Badges, Breadcrumbs, Sidebars, and Tabs. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Sidebars (for desktop), Bottom Navigation (for mobile), or Tab Navigation when dealing with limited content that needs quick switching. |
| illustration: | |
|  | |
| On a corporate website, the navbar includes the company logo on the left and links to critical sections like "About Us," "Services," "Blog," and "Contact" across the top. A search icon and a login link appear on the right. On mobile devices, these links collapse into a hamburger menu for space efficiency. | |

### **7. Specification**

| Web Responsive: | |
| --- | --- |
|  | |
|  | |
|  | |
|  | |
| Mobile Application: | |
|  | |
|  | |
|  | |
|  | |

## Number Input

### **1. General Information**

| Component name: | **Number Input** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Number Input component is a specialized type of input field designed for numeric entry. |
| Objective: | To allow users to enter numeric values—such as quantity, age, or price—ensuring precise and easily validated data input. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Provide clear instructions or labels indicating what the number input is for and any specific requirements (e.g., unit of measure, required precision). | ❌Don't use number inputs for data that isn't inherently numeric, like phone numbers or postal codes, since these might require specific formatting. |
|  |  |
| ✅Implement sensible defaults and range limits to prevent user error. | ❌Avoid cluttering the UI with too many number inputs close together, leading to user confusion and errors. |
|  |  |
| ✅Ensure the control size is adequate for interaction, especially on mobile devices. |  |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use Number Input when users need to input strictly numerical data—such as calculated values, amounts, weights, or percentages—especially when preventing non-numeric input is important.** |
| --- | --- |
| Should not be used when: | **Avoid using Number Input when the value might include letters (e.g., postal codes or reference numbers) or when users should choose from predefined numeric options—use a Dropdown instead.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **A good Number Input should define min, max, and step values to control the allowed range and display validation messages when users enter invalid values.** |
| ❌Guidelines to avoid: | **Avoid using Number Input without range limits or contextual hints, as users may enter unexpected or inappropriate values.** |

| Appearance: | |
| --- | --- |
|  | |
| Visual Design | Typically, it resembles a standard text input field but includes small arrow buttons (up and down) on one side to adjust the value incrementally. |
| Increment Buttons | These buttons, often styled as vertical arrows, allow users to increase or decrease the value without keyboard input. |
| Responsive | Adapts to various device sizes while maintaining usability and visibility. |
| State Indicators | Reflects active, hover, focus, and disabled states through changes in border color, background color, or opacity. |

| Behavior: | |
| --- | --- |
| **Value Adjustment** | Users can change the number by typing directly into the input field, using the up and down arrows or the keyboard's up and down arrow keys. |
| **Validation and Limits** | Supports minimum and maximum value constraints to prevent the entry of numbers outside a specific range. Additionally, it can validate against step values, ensuring numbers adhere to specified increments. |
| **Feedback** | Provides real-time validation feedback, such as highlighting or error messages, when entered values are invalid or out of range. |

### 

### **3. Properties & Customization**

| Supported sizes (Size Options): | **[Small / Medium / Large or other]** |
| --- | --- |
| Icon supported or not: | **[Yes / No]** |
| Can the color be adjusted? | **[Yes / No / Color adjustment conditions]** |
| Does it support Responsive Design? | **[Yes / No / What are the limitations?]** |
| illustration: | |
|  | |
| This control allows users to input, increment, or decrement numbers through direct input or using integrated controls. It's beneficial in forms requiring precise numerical values, such as quantities, measurements, or settings that adjust numerically. | |
| **Basic Number Input** - Just the field and increment controls, no additional formatting or units. | |
| **Extended Range Number Input** - They are designed for more extensive ranges, possibly with a slider for easier adjustments over broad values. | |
| User Interaction: | |
| **Direct Input** | Users can click into the number field and type a value directly. |
| **Incremental Adjustment** | Clicking the up or down arrows or pressing the up and down keys on the keyboard adjusts the number within the defined constraints. |
| **Mouse Wheel** | Optionally, users can adjust the value by scrolling the mouse wheel when the input is focused. |
| **Accessibility** | Each interactive element is fully accessible, with keyboard and screen reader support. |

### **4. Accessibility**

| Keyboard Accessible | The Tab key navigates to and from the component, and the arrow keys adjust the values. |
| --- | --- |
| Screen Reader Support | Proper labels for screen readers using aria-label or aria-labeled and communicates the role of the component with role= "spin button" along with properties like aria-value min, aria-value max, and aria-value now. |
| Focus Management | Indicates focus state to assist users navigating via keyboard. |
| illustration: | |
| **(Attached is an example image of supported usage. Accessibility)** | |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Number Input is commonly used with Labels, Unit Indicators (e.g., $, kg), Error Messages, Tooltips, and sometimes Steppers (+/− buttons) for easier adjustments. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Sliders (for continuous ranges), Dropdowns (for limited numeric choices), or Text Inputs with numeric validation when more flexible formatting is needed. |
| illustration: | |
|  | |
| In an e-commerce site, a number input might be used to select the quantity of a product before adding it to the cart. Users can quickly click the up arrow to increase the amount or type the number directly into the input field. | |

### **7. Specification**

| Spacing |
| --- |
| Button Inside |
|  |
|  |
|  |
| Button Outside + Input |
|  |
|  |
|  |
|  |
| Button Outside |
|  |
|  |
|  |
| Properties |
| Button Inside State |
|  |
|  |
|  |
|  |
| Button Outside + Input State |
|  |
|  |
|  |
|  |
| Button Outside State |
|  |
|  |
|  |
|  |

## Pagination



### **1. General Information**

| Component name: | **Pagination** |
| --- | --- |
| Category: | **Navigation Components** |
| Description: | The Pagination component is used to divide large sets of data into manageable pages, allowing users to navigate through them sequentially. |
| Objective: | To divide large sets of data into smaller, manageable pages, improving performance and allowing users to navigate content efficiently. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Provide ample space between pagination links to avoid mis-clicks, which is especially important on mobile devices. | ❌Don't use pagination as the only navigation option for accessing large data sets if a more dynamic method like "infinite scrolling" might be more appropriate. |
|  |  |
| ✅Limit the number of visible page links if the total number of pages is very high to prevent the pagination bar from becoming too large. | ❌Avoid confusing designs, such as placing non-standard controls or changing the order of pagination buttons. |
|  |  |
| ✅Use pagination with other navigation aids, such as filtering and sorting, to enhance usability. | ❌Don't overload the user with too many options; keep the interface simple and intuitive. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use Pagination when displaying large volumes of content, such as product listings, articles, or search results, where loading everything at once would hurt performance or overwhelm users.** |
| --- | --- |
| Should not be used when: | **Avoid using Pagination when the content is limited and fits comfortably on a single page, or when continuous scrolling (like social feeds) offers a better experience.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Show clearly labeled next/previous buttons and clickable page numbers. Highlight the current page, and use ellipsis or arrows to shorten long page ranges for cleaner UI.** |
| ❌Guidelines to avoid: | **Avoid forcing users to click “next” repeatedly without access to specific pages, and don’t make page numbers too small—especially on mobile devices.** |

| Appearance: | |
| --- | --- |
| **Design** - Typically, it consists of numbered buttons representing the pages and arrows or "Previous" and "Next" buttons for navigating between pages. Advanced designs may include "First" and "Last" buttons or ellipses (...) to indicate hidden pages. | |
| **Styling** - Styled to match the application's or website's design language, with attention to making active, hovered, and disabled states visually distinct. | |
| **Responsiveness** - Adapts to various screen sizes, ensuring that pagination controls are touch-friendly and spaced appropriately for smaller devices. | |
| This control is essential for improving user experience by reducing load times and making content more accessible. It is beneficial in contexts like search results, product listings, or long lists of articles. | |

| Behavior: | |
| --- | --- |
| **Navigation** | Users can move to a specific page by clicking its number or navigate sequentially using the "Previous" and "Next" buttons. Optionally, the "First" and "Last" buttons can jump to the beginning or end of the data set. |
| **Dynamic Update** | The page's content updates dynamically upon user interaction with the pagination controls, avoiding reloading the entire page. |
| **State Management** | Ensures the correct pagination button is highlighted as active based on the current page and appropriately turns off navigation buttons when applicable (e.g., disabling "Previous" on the first page). |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal but have 3 design for usage** |
| --- | --- |
| Icon supported or not: | **No** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **Yes** |
| Variants: | |
|  | |
| User Interaction: | |
| **Click Interaction** | Users interact with pagination controls primarily through mouse clicks or taps on mobile devices. |
| **Keyboard Accessibility** | Supports navigation through keyboard controls, such as tabbing to pagination links and using enter to select. |
| **Focus Indicators** | Provide clear focus indicators for keyboard navigation to improve accessibility. |

### **4. Accessibility**

| ARIA Labels | Includes appropriate ARIA labels and roles, such as aria-label="Pagination" for the navigation area and aria-current="page" for the active page link. |
| --- | --- |
| Screen Reader Support | Ensures that screen readers correctly announce all pagination controls, including the purpose of each button and the current page. |
| Non-Visual Navigation | Supports screen readers and other assistive technologies with proper semantic markup and aria attributes. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Pagination often works with components like Tables, Lists, Card Grids, Search Results, Dropdowns (for items per page), and Loaders (for async content loading). |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Infinite Scroll (for continuous feeds), “Load More” buttons (to fetch content in chunks), or Virtual Scrolling (for large datasets with performance focus). |
| illustration: | |
|  | |
| In an online store, a Pagination component at the bottom of the product listing page allows users to navigate through pages of products. When a user selects a page number or clicks "Next," the list updates to show the products from the chosen page without reloading the entire website. | |

### **7. Specification**

| Page type |
| --- |
|  |
|  |
|  |
|  |
| Page jump |
|  |
|  |
| Page show row |
|  |
|  |
|  |

## Pop Over

### **1. General Information**

| Component name: | **Pop Over** |
| --- | --- |
| Category: | **Display & Feedback Components** |
| Description: | The Popover component is a non-modal, floating panel that appears near a trigger element (such as a button or link) to provide additional content, context, or actions related to that element. |
| Objective: | The objective of a Popover is to present contextual information, additional actions, or details related to a trigger element without navigating away from the current view. |

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Use popovers to provide additional information or contextual tools without overcrowding the main content. | ❌Don't use popovers for critical information the user must see, as they might be missed or ignored. |
|  |  |
| ✅Keep the content within popovers concise and focused; avoid overloading with too much information or actions. | ❌Avoid using popovers for complex tasks; consider using a modal dialog if more space or user focus is needed. |
|  |  |
| ✅Ensure the popover does not obscure important information or disrupt the user's workflow. | ❌Don't make popovers too persistent; they should close automatically when users interact elsewhere. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use Popovers to display information or actions tied directly to a specific UI element—such as more options in a table row, item details, or small inline forms.** |
| --- | --- |
| Should not be used when: | **Avoid using Popovers for long or complex content, or stacking multiple Popovers. Don’t use them in place of Modals when the content is critical and requires full user attention.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **A Popover should appear near its trigger with a directional arrow, close automatically when clicking outside, support mouse and keyboard interactions, and remain readable with appropriate size.** |
| ❌Guidelines to avoid: | **Avoid placing Popovers too far from their trigger, using them for critical actions that shouldn’t be missed, or overloading them with too many interactive elements.** |

| Appearance: | |
| --- | --- |
|  | |
| Design | Typically, it is a small, floating box with an arrow pointing toward the element that triggered it. The design should be clean and distinct to stand out from the underlying content. |
| Colors and Styles | This should align with the application's design system, often featuring a background color that contrasts with the main UI to draw attention. |
| Shadow and Border | Usually, it includes a subtle shadow and border to help distinguish the popover from other page elements. |

| Behavior: | |
| --- | --- |
| Trigger | It is activated by a user action, such as clicking or hovering over a specific element. The particular trigger behavior should be consistent throughout the application (e.g., always on click to avoid accidental activation). |
| Positioning | It is dynamically positioned based on the trigger element's location and available space. Typical positions include above, below, to the left, or the right of the trigger. |
| Dismissal | This can be dismissed by clicking outside the popover, pressing the escape key, or performing another action within the popover, such as clicking a close button or completing a form. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **[Small / Medium / Large or other]** |
| --- | --- |
| Icon supported or not: | **[Yes / No]** |
| Can the color be adjusted? | **[Yes / No / Color adjustment conditions]** |
| Does it support Responsive Design? | **[Yes / No / What are the limitations?]** |
| Variants: | |
| **Informational Popovers** - Provide additional texts, links, or concise content related to the trigger element. | |
| **Form Popovers** - Include forms or input fields for quick interactions without leaving the page. | |
| **Interactive Popovers** - Contains elements like sliders, switches, or buttons for settings or adjustments. | |
| User Interaction: | |
| **Activation** | Users can open the popover by interacting with the trigger element, which should be indicated as interactive. |
| **Focus Management** | When opened, focus should be moved to the popover to facilitate accessibility and ease of use, particularly for keyboard and screen reader users. |
| **Interactive Elements** | These Can contain links, buttons, or other interactive elements that users can use. |

### **4. Accessibility**

| ARIA Roles and Attributes | Use appropriate ARIA roles such as role="dialog" or role="tooltip" depending on the content, and manage aria-has popup and aria-expanded attributes on the trigger element. |
| --- | --- |
| Keyboard Navigation | Ensure that all interactive elements within the popover can be navigated via the keyboard—support closing the popover using the ESC key. |
| Screen Reader Support | Ensure that the content within the popover is accessible to screen readers, with proper labeling and instructions if needed. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Popovers are commonly used with Buttons, Icon Buttons, Tooltips, Lists, Form Inputs, and Dropdowns—especially for inline or contextual menus. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Tooltips (for short descriptions), Modals (for important or larger content), or Dropdowns (when selecting from a list of options is the goal). |
| illustration: | |
|  | |
| In an e-commerce application, a popover might appear when a user hovers over a product image, showing a quick view of the product details and offering actions like "Add to Cart" or "Wishlist." | |

### **7. Specification**

|  |
| --- |
|  |
|  |
|  |
|  |

## Progress

### **1. General Information**

| Component name: | **Progress** |
| --- | --- |
| Category: | **Display & Feedback Components** |
| Description: | The Progress Component visually represents the completion status of a process or task, providing users with clear feedback on their progress towards a goal. |
| Objective: | The objective of a Progress component is to visually indicate the completion status of a time-consuming process—such as downloading, uploading, or multi-step actions—so users understand that progress is happening and how much is left. |

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Use determinate indicators for processes where progress can be calculated, giving users precise feedback. | ❌Avoid using progress indicators for actions that complete quickly, where an indicator may be more distracting than helpful. |
|  |  |
| ✅Pair progress components with descriptive text to communicate what is loading or the progress status. | ❌Do not confuse users; provide auxiliary information or estimates when possible, especially for indeterminate tasks. |

### **2. Usage Guidelines**

| It should be used when: | **Use Progress when a process takes time or involves multiple steps, such as file uploads, software installations, long forms, or order submissions.** |
| --- | --- |
| Should not be used when: | **Avoid using Progress when the task is instantaneous or when exact progress can’t be measured—use a Loading Spinner instead for indeterminate operations.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Use the correct type: determinate (with a known percentage) for predictable tasks and indeterminate (without a percentage) for unknown durations. Include clear labels like “Uploading…” when appropriate.** |
| ❌Guidelines to avoid: | **Avoid showing static or stuck progress bars without explanation, and don’t overwhelm users with multiple progress indicators on the same screen.** |

| Appearance: | |
| --- | --- |
| **Linear Progress Bar** - A horizontal bar that fills from left to right as progress is made. | |
| **Circular Progress Indicator** - A circular track that fills in a clockwise direction is often used for tasks with undetermined length. | |
|  | |
|  | |
| **Colors** | * Primary actions use the brand CI colors to maintain consistency and visibility. * Creation of color for background tracks, ensuring the progress indicator stands out. |
| **Size** | Configurable sizes and thicknesses to ensure visibility and fit within various layouts. |
| **Animation** | Smooth and continuous, reflecting real-time progress without causing distraction. |

| Behavior: | |
| --- | --- |
| Determinate | Shows exact progress made, ideal for tasks with a known duration or completion percentage. |
| Indeterminate | Displays a looping animation to indicate activity for tasks with unknown duration. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Small / Medium / Large** |
| --- | --- |
| Icon supported or not: | **No** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **Yes** |
| Variants: | |
|  | |
| **Progress bar** | Shows exact progress made, ideal for tasks with a known duration or completion percentage. |
| **Indeterminate** | Displays a looping animation to indicate activity for tasks with unknown duration. |

### **4. Accessibility**

| Screen Reader Support | Mark up with role="progress bar" and properties such as aria-value now, aria-value min, and aria-value max to convey progress information to assistive technologies. (Optional) |
| --- | --- |
| Keyboard Navigation | Provide descriptive labels using aria-label or aria-labelledby to communicate the purpose of the progress indicator. (Optional) |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Progress is often used with Buttons (e.g., “Submit”), Modals/Dialogs (to show status), Toasts/Notifications, Steppers, and Text labels (to display percentage or descriptions). |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include a Loading Spinner (for unknown durations), Skeleton Loaders (for content loading), or Steppers (for process-driven interactions with steps). |
| illustration: | |
|  | |
| In a file upload scenario, a user selects a large video file to upload. A Linear Progress Bar appears, initially in an indeterminate state, while the upload size is calculated. Once the calculation is complete, the progress bar switches to a determinate state, filling in proportion to the upload's completion. The text above the bar updates with the percentage of uploads completed, offering the user a clear indication of progress. | |

### **7. Specification**

|  |
| --- |
|  |
|  |
|  |

## Radio

## 

### **1. General Information**

| Component name: | **Radio** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Radio Button component is used to present a set of mutually exclusive options to the user. |
| Objective: | The objective of a Radio Button is to allow users to select only one option from a group, making it ideal for mutually exclusive choices. |
|  | The radio button mainly guides users to take action at each step in the overall journey. The radio button is used to form a form for choice selection.  On the other hand, it contains a specific pattern that generates total full options; users can see overall choices in one place and don’t have to scroll down, like in a Select menu. Also, it’s fixed for selecting only 1 option from the total choices; this differs from the checkbox that allows users to select more than 1 option. It can be said that a radio button might be used for mandatory selection cases.  For the writing part, there are similar rules applied to other input control components: clear, concise, and useful. In other words, writing a radio button should be short but fully comprehensive. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Group radio buttons logically represent all available options for a single choice. | ❌Don't use radio buttons if multiple selections are allowed; consider checkboxes instead. |
|  |  |
| ✅Place labels close to their radio buttons to facilitate easy selection and understanding. | ❌Avoid using radio buttons for complex choices requiring additional context or information to decide. |
|  |  |
| ✅Use radio buttons for clear, mutually exclusive choices where only one option is valid. | ❌Don't clutter the UI with too many radio buttons; consider other components like dropdowns for extensive lists of options. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use Radio Buttons when users must choose only one item from a list of clearly visible options, such as gender, payment method, or satisfaction level.** |
| --- | --- |
| Should not be used when: | **Avoid using Radio Buttons when multiple selections are allowed (use Checkboxes instead) or when there are too many options that clutter the interface.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Each Radio group should have a clear group label, with easy-to-understand labels for each option. Use vertical alignment for readability, and set a reasonable default if needed.** |
| ❌Guidelines to avoid: | **Avoid unlabeled Radios or combining them with hidden or hard-to-reach options. Also, don’t mix Radios with unrelated interactions that confuse the selection flow.** |

| Appearance: | |
| --- | --- |
|  | |
| Unlike checkboxes, where multiple selections are allowed, radio buttons require the user to select only one option from a group. This component is essential for forms and surveys where a single choice must be made clear and straightforward. | |
|  | |
| **Visual Style** | A small circle with an outer ring and a filled inner circle is typically represented when selected. Unselected radio buttons show only the outer ring. |
|  | |
| **Grouping** | Radio buttons are presented in groups to denote the set of options among which the user can make a single selection. |
|  | |
| **Labels** | Each radio button is accompanied by a label that clearly describes the option. To enhance usability, labels are clickable. |
|  | |
| **State Indication** | Visual feedback for different states (selected, unselected, hover, disabled) is provided through color changes, opacity adjustments, or other styling cues. |

| Behavior: | |
| --- | --- |
| **Mutual Exclusivity** | Selecting a radio button automatically deselects any other selected button in the same group, enforcing a single choice. |
| **Navigation** | Users can navigate through radio button options using keyboard controls, typically the arrow keys, to make a selection without relying on a mouse. |
| **Form Submission** | The selected option is submitted as part of a form, represented by the radio button's value attribute. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **No, just Primary CI Color and Success Color** |
| Does it support Responsive Design? | **No** |
| Variants: | |
| **Inline** - Radio buttons are displayed horizontally in a line, suitable for short lists or where vertical space is limited. | |
| **Stacked** - Radio buttons are stacked vertically, providing more apparent separation and readability for longer lists of options. | |
| User Interaction: | |
| **Click to Select** | Users can click a radio button or its associated label to select an option. Clicking on an already selected radio button does not change its state. |
| **Keyboard Accessibility** | Keyboard users can tab into the radio group and use arrow keys to select an option, with the spacebar or enter key to confirm the selection. |
| **Feedback on Selection** | Provides immediate visual feedback when an option is selected, typically by filling the inner circle of the radio button. |

### **4. Accessibility**

| Keyboard Navigation | Ensure the entire radio group is treated as a single tab stop with arrow essential navigation among the options. |
| --- | --- |
| Screen Reader Support | Use proper ARIA roles and attributes (role="radio group" for the container and role="radio" for each button) to ensure screen readers accurately convey the group's nature and the selection state. |
| Focus Indicators | Provide clear focus styles for keyboard navigation to indicate which radio button is currently focused. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Radios typically work with Labels, Fieldsets (to group options), Form Validation, Tooltips (for extra info), and Buttons (in sequential flows or forms). |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Dropdowns (for limited space), Segmented Controls (for quick taps), or single-select Lists—depending on layout and interaction needs. |
| illustration: | |
|  | |
| In a survey form asking about a favorite type of cuisine, radio buttons list options such as "Thai," "Italian," "Mexican," "Japanese," etc. Users can select only one cuisine as their favorite, and the selection is quickly submitted as part of the form. | |

### **7. Specification**

| Spacing |
| --- |
|  |
|  |
| Properties |
|  |
|  |
|  |
|  |
|  |

## Rating

## 

### **1. General Information**

| Component name: | **Rating** |
| --- | --- |
| Category: | **Display & Feedback Components** |
| Description: | The Rating component allows users to provide feedback in the form of a rating, typically using stars or similar icons. |
| Objective: | The objective of a Rating component is to let users express satisfaction or evaluate quality—such as for products, services, or content—through an intuitive visual scale like 1 to 5 stars. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Provide visual feedback as the user interacts with the rating component, such as changing the color of the stars on hover. | ❌Don't use the rating component for critical evaluations without additional qualitative feedback options, as numeric ratings alone can be ambiguous. |
|  |  |
| ✅Ensure the clickable area of each star is large enough to be easily selectable, accommodating both desktop and touch interactions. | ❌Avoid using non-standard symbols that might confuse users about the rating scale or expectations. |
|  |  |
| ✅If appropriate, allow users to clear or revise their rating after the initial selection. |  |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use Rating when you want users to quickly provide feedback without typing—such as after using a service, reviewing a product, or voting on content.** |
| --- | --- |
| Should not be used when: | **Avoid using Rating if detailed feedback is required or if the score has no real impact on decision-making or UX.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Use familiar icons like stars or hearts, with clear hover states and optional labels (e.g., “Excellent,” “Poor”) to help users understand their choice. Pre-select defaults if appropriate.** |
| ❌Guidelines to avoid: | **Avoid Ratings without context or labels, as users may not know what each level means. Don’t overcomplicate with too many levels (like 1–10), which can lead to indecision.** |

| Appearance: | |
| --- | --- |
|  | |
| **Visual Style** | Generally, it consists of a row of icons—commonly stars and hearts—that users can select to indicate a rating. Stars usually range from 1 to 5 or 1 to 10. |
|  | |
| **Icon** | The icons can be filled, outlined, or partially filled based on the user's rating. |
|  | |
| **Color** | Typically, it uses a standout color, such as gold or yellow for stars, but it can be customized to fit the application's or website's design language. |
|  | |
| **Size** | Adjustable size to ensure it is appropriately visible across devices and fits within different UI contexts. |

| Behavior: | |
| --- | --- |
| **Interactive Rating** | Users can set a rating by clicking or tapping a star. Hovering over the stars allows users to preview the rating before selection. |
| **Fractional Ratings** | Supports whole and fractional ratings (e.g., 3.5 stars), allowing for more precise user feedback. |
| **Accessibility Enhancements** | Optionally, keyboard navigability and screen reader support ensure all users can interact effectively with the rating component. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Dynamic Size** |
| --- | --- |
| Icon supported or not: | **-** |
| Can the color be adjusted? | **Yes** |
| Does it support Responsive Design? | **Yes** |
| Variants: | |
|  | |
| **Star Rating** | The most common variant uses stars as the rating symbol. |
| **Heart Rating** | It's like a star rating variant as the rating symbol, too. |
| User Interaction: | |
| **Setting a Rating** | Click or tap on a star to set the rating. The number of stars highlighted represents the rating given. |
| **Hover to Preview** | Hovering over the stars previews desktop device ratings. The preview updates as the mouse moves over different stars. |
| **Accessibility Features** | Users can navigate the rating component using keyboard arrows and select a rating with the Enter key. |

### **4. Accessibility**

| Keyboard Navigation | It includes full keyboard support for navigating and selecting ratings. |
| --- | --- |
| Screen Reader Support | Uses aria-label or similar ARIA attributes to communicate the purpose and value of the rating to screen reader users. |
| Focus Management | Indicates which star has focus when navigating via keyboard. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Rating commonly works with Review Text Areas (for optional comments), Tooltips (to explain levels), Submit Buttons, and Cards or Lists (to display overall scores). |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Dropdowns (for numeric selection), Likert Scales (in surveys), or simple Thumbs Up/Down for binary feedback. |
| illustration: | |
|  | |
| Users rate product details using the Star Rating component on an e-commerce platform. Each product detail's interface includes a 5-star rating system, where users can click to rate the product. The average rating based on all user feedback immediately reflects the product's rating. | |

### **7. Specification**

| Spacing |
| --- |
|  |
|  |
| Properties |
|  |
|  |
| State |
|  |
|  |
|  |

## Scroll Area

### **1. General Information**

| Component name: | **Scroll Area** |
| --- | --- |
| Category: | **Display & Feedback Components** |
| Description: | The Scroll Area component enhances the user interface by providing a defined space for scrollable content |
| Objective: | The objective of a Scroll Area is to allow users to view overflowing content within a limited space by scrolling, without needing to reload or navigate away from the current page. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Ensure that all critical content within the scroll area is accessible without relying solely on the scroll to view it, particularly for crucial actions or information. | ❌Don't hide essential UI elements, such as navigation controls or call-to-action buttons, within a scroll area where they might not be seen immediately. |
|  |  |
| ✅Design custom scroll bars to be wide enough to be easily clickable and maintain visibility without being obtrusive. | ❌Avoid overly sensitive scrolling that can cause content to move too quickly and be difficult to control, reducing usability. |
|  |  |
| ✅Provide alternatives or enhancements such as pagination or "load more" buttons for users who may find continuous scrolling difficult. | ❌Don't use scrolling areas for content that needs to be visible for comprehension or action. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use Scroll Area when displaying long or overflowing content such as lists, long texts, or multiple items—especially inside modals, sidebars, or compact UI sections.** |
| --- | --- |
| Should not be used when: | **Avoid using Scroll Areas when the content can fit without scrolling or when it would result in multiple nested scrolls, which can confuse users.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Ensure scrollbars are visible or appear on interaction. Use in constrained areas and support both vertical and horizontal scrolling when needed. Allow interaction via mouse, keyboard, and touch.** |
| ❌Guidelines to avoid: | **Avoid completely hidden scrollbars, placing scroll behavior on elements like title bars or footers, and unintentionally affecting primary content layout.** |

| Appearance: | |
| --- | --- |
| **Scroll Bars** - Custom-designed scroll bars that align with the application's overall style are often more subtle or stylized than default browser scroll bars. | |
| **Visual Design** - Typically, it appears as a container that encloses other elements, showing scroll bars when the content exceeds the container's dimensions. | |
| **Size and Padding** - It is appropriately sized to fit within its context in the layout, with adequate padding around the content to ensure it is visually appealing and does not clash with the scroll bars. | |

| Behavior: | |
| --- | --- |
| Overflow Control | Automatically displays horizontal or vertical scroll bars depending on the content's size relative to the container. |
| Smooth Scrolling | Optionally, it includes smooth scrolling mechanisms to enhance the user experience by making the movement within the scroll area more fluid. |
| Responsive | Adapts to varying screen sizes, ensuring functionality and appearance are maintained across devices. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Small / Medium / Large** |
| --- | --- |
| Icon supported or not: | **No** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **Yes** |
| Variants: | |
|  | |
| **Horizontal Scroll Area** | Used for horizontally-span content, such as image galleries or timelines. |
| **Vertical Scroll Area** | It is more common for text content, lists, and stacked elements. |
| User Interaction: | |
| **Scrolling Mechanisms** | Users can scroll using the mouse wheel, trackpad gestures, keyboard arrows, or by dragging the scroll bars. |
| **Touch Interaction** | It supports touch gestures on mobile devices, such as swiping to scroll through the content. |
| **Accessibility** | It provides keyboard navigability; all scrollable content can be accessed through assistive technologies. |

### **4. Accessibility**

| Keyboard Navigation | Ensures users can navigate through scrollable content using standard keyboard controls, such as arrow keys, page up, page down, home, and end. |
| --- | --- |
| Screen Reader Support | Uses appropriate ARIA roles and properties to indicate the presence and functionality of the scroll area, enhancing screen reader support. |
| Focus Management | Manages focus within the scroll area to prevent it from inadvertently trapping the user's focus. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Scroll Area often works with Lists, Tables, Card Grids, Modals, Sidebars, Dropdowns (with many options), and Virtual Scroll components for handling large data sets. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Pagination (to split content), Accordions (to expand/collapse sections), or “Load More” buttons to control content in smaller chunks. |
| illustration: | |
|  | |
| A vertical scroll area displays a list of tables in a data platform. The scroll area allows users to browse the list efficiently without navigating away from the main page, providing a seamless user experience. | |

## Segmented Control

### **1. General Information**

| Component name: | **Segmented Control** |
| --- | --- |
| Category: | **Navigation Components** |
| Description: | The Segmented Control component, often called a segment button or toggle button group, displays a small group of related options side-by-side. It allows users to quickly switch between multiple options where each choice represents a mutually exclusive selection. |
| Objective: | The objective of a Segmented Control is to allow users to quickly select one option from a small set, typically for switching views or modes with a clear, side-by-side layout. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Keep the number of segments reasonable to avoid overwhelming users and to maintain usability. | Don't use segmented controls for actions that require more complex inputs or multiple steps; instead, use more comprehensive form controls. |
|  |  |
| ✅Clearly label each segment to accurately describe its effect and ensure that the labels are concise. | ❌Avoid using segmented controls as the only method for critical actions, especially if users might need additional information to make a decision. |
|  |  |
| ✅Use segmented controls for precise, distinct choices that do not require additional user input to take effect. | ❌Please don't clutter the control with too many options, making it challenging to select quickly. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use Segmented Controls when users need to switch between views, modes, or filters—such as “Grid/List,” “Day/Week/Month,” or quick toggle options.** |
| --- | --- |
| Should not be used when: | **Avoid using Segmented Controls when there are too many options (typically more than 4–5), or when labels are long and make the layout cluttered or hard to read.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Each segment should have a short, clear label (or icon with tooltip), highlight the selected state clearly, and support both mouse and keyboard accessibility.** |
| ❌Guidelines to avoid: | **Avoid vague or similar-looking choices without context, like “Option A / Option B,” and don’t place segments too close together—especially on mobile—where misclicks can occur.** |

| Appearance: | |
| --- | --- |
| This component is commonly used for settings that affect the content or appearance of an application immediately upon selection. | |
|  | |
| Visual Design | It consists of two or more segments within a single control interface, each representing an option. Typically, each segment appears as a button, and the currently active segment is highlighted distinctly to indicate selection. |
| Consistency | It is styled consistently with the application language, using the same color scheme, typography, and borders as other form elements. |
| Compactness | They are designed to be compact, efficiently using space while maintaining tap-friendly sizes for touch interfaces. |

| Behavior: | |
| --- | --- |
| Mutual Exclusivity | Only one segment can be selected at a time, and selecting a new segment immediately deselects the others. |
| Immediate Action | When a segment is selected, changes take effect immediately, updating the associated content or configuration without the need to submit a form. |
| Accessibility | This feature provides accessible interaction patterns, ensuring that all users, including those using keyboards and screen readers, can use the control effectively. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **Yes** |
| Variants: | |
| **Text Segments** - Primary segments are labeled with text. Most common and universally applicable. | |
| **Icon Segments** - It uses icons instead of text to save space or to provide a more intuitive interface for global audiences. | |
| **Mixed Segments** - Combines text and icons to enhance clarity and user recognition. | |
| User Interaction: | |
| **Selection** | Users can select an option by clicking or tapping the segment. The control provides visual feedback to indicate the active state. |
| **Keyboard Navigation** | Users can navigate between segments using arrow keys and select options using the Enter or Space key. |
| **Feedback** | Visual changes and possibly auditory feedback occur upon selection to confirm the action. |

### **4. Accessibility**

| Keyboard Navigation | Fully navigable using the keyboard alone. |
| --- | --- |
| Screen Reader Support | They are correctly labeled with ARIA roles and states, such as role="group" for the entire control and role="button" for each segment, with aria-selected="true" for the active segment. |
| Focus Indicators | Displays clear focus indicators when segments receive keyboard focus to aid users who rely on visual cues. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Segmented Controls often work with Tabs, Filters, Toggle Groups, Card or Table Views, and Content Switchers to display view-specific content based on selection. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Dropdowns (for space-saving), Radio Buttons (in forms), or Tabs (for structured content switching). |
| illustration: | |
|  | |
| In a photo editing app, a segmented control might switch between "Edit," "Filters," and "Crop" modes. Users tap on each segment to switch views within the app, allowing for quick and easy access to different editing tools. | |

### **7. Specification**

| Spacing |
| --- |
|  |
|  |
| Properties |
|  |
|  |
| State |
|  |
|  |
|  |
|  |

## Select

## 

### **1. General Information**

| Component name: | **Select** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Select component, often referred to as a dropdown menu, allows users to choose one option from a collapsible list of options. |
| Objective: | The objective of a Select component is to let users choose one (or multiple) options from a predefined list that appears only on interaction—helping save space and keep the UI clean. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Use when space is limited or when presenting a long list of impractical options to display simultaneously. | ❌Don't overload with too many options. Consider other UI elements like searchable select boxes for very long lists. |
|  |  |
| ✅Provide a default selection or instructional placeholder to guide user interaction. |  |
|  |  |
| ✅Ensure the list is sorted in a logical order (alphabetically, numerically, or by popularity) to facilitate quick selection. |  |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use Select when there are more than 3 options and space-saving is important—like choosing a country, product category, or time range.** |
| --- | --- |
| Should not be used when: | **Avoid using Select when there are very few options (e.g. 1–3), or when all options should be visible at once—Radio Buttons may be more appropriate in those cases.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Include a clear label or placeholder, sort items logically, support scrolling or searching for long lists, and ensure accessibility via both keyboard and mouse.** |
| ❌Guidelines to avoid: | **Avoid overwhelming users with ungrouped or unfiltered items, omitting default states, and using Select when freeform input is more appropriate.** |

| Appearance: | |
| --- | --- |
| It's a fundamental UI element for forms and interfaces where space efficiency and user decision-making are prioritized. Select components simplify displaying lengthy options by collapsing them into a single, easily accessible list. | |
| **Visual Design** - Features a box with a downward-facing arrow to indicate expandability. When inactive, it displays the currently selected option or a placeholder. | |
| **Expanded List** - Upon interaction, it expands to show all available options, which can be scrolled through if the list exceeds the viewport. | |
| **Option Highlighting** - Highlights the hovered or focused option within the list for more precise navigation and selection. | |
| **States** - Adapts visual cues for different states, such as Default, Hover, Disabled, and Focused, enhancing usability and guiding user interaction. | |

| Behavior: | |
| --- | --- |
| Expand/Collapse | Clicking on the select box or focusing on it through keyboard navigation toggles the visibility of the options list. |
| Option Selection | Users can select an option by clicking on it with a mouse, navigating to it with keyboard controls, and pressing Enter. |
| Single Choice | Only one option can be selected at a time, updating the displayed value in the select box to reflect the user's choice. |
| Form Integration | The selected value gets submitted as part of a form, identified by the option's value attribute. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **Yes** |
| Variants: | |
| **(Attach sample images of UI customization such as size, color, icon)** | |
| User Interaction: | |
| **Opening the List** | It is triggered by clicking on the select box, tapping it, and pressing the down arrow key. |
| **Navigating Options** | They were performed using mouse clicks or keyboard arrows. Scrolling is enabled for lists extending beyond the initial view. |
| **Selecting an Option** | This is accomplished by clicking an option or highlighting it with the keyboard and pressing Enter. |
| **Closing the List** | It occurs automatically upon selection or when clicking outside the select box. It can also be closed with the Esc key without making a selection. |

### **4. Accessibility**

| Keyboard Accessibility | Ensures users can navigate and select options using only the keyboard. |
| --- | --- |
| Screen Reader Support | Properly label the select component and its options with descriptive text, leveraging ARIA attributes to enhance accessibility. |
| Focus Management | Provides clear focus indicators and manages focus to maintain a logical tab order within forms. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Select often works with Labels, Tooltips (for option explanations), Validation Messages, Dropdown Icons, and sometimes embedded Search Fields for long lists. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Radio Buttons (for fewer options), Segmented Controls (for quick view switching), or Autocomplete (when dealing with large, searchable datasets). |
| illustration: | |
|  | |
| On a travel booking site, a Select component is used to choose a departure airport from a list of options. When the user clicks on the select box, a dropdown menu appears with a list of airport options. The user selects their preferred airport and then updates the select box to display the chosen airport, streamlining the booking process. | |

### **7. Specification**

| Spacing |
| --- |
|  |
|  |
| State |
|  |
|  |
|  |
|  |
|  |
| Option List |
|  |
|  |

## Slider

## 

### **1. General Information**

| Component name: | **Slider** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Slider component is a user interface element that allows users to select a value or range by moving an indicator along a track. |
| Objective: | The objective of a Slider is to let users select a single value or a range within a defined limit quickly and intuitively, using a draggable control instead of typing. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Provide visual indicators for the selected value, mainly if the slider controls a non-visual setting like volume. | ❌Don't use sliders for binary choices; toggle switches are more appropriate. |
|  |  |
| ✅Ensure the slider's track has a sufficient clickable area for easy use. | ❌Avoid using sliders for precise data entry if the exact value is critical, as it can be challenging to select a specific value accurately. |
|  |  |
| ✅Use labels or ticks for significant values along the track to estimate the slider's position. | ❌Don't rely on color alone to convey information; ensure that text or icons are also used to enhance clarity. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Slider when users need to select a continuous value or range—such as volume level, price filters, or time range—especially when speed and rough accuracy are more important than precision.** |
| --- | --- |
| Should not be used when: | **Avoid using a Slider when precise input is required, like item quantities, or when choices are limited and discrete—Dropdowns or Radio Buttons are better in those cases.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Show the current value or tooltip while dragging, clearly define min, max, and step, and consider adding ticks or labels to help users estimate values more easily.** |
| ❌Guidelines to avoid: | **Avoid Sliders without contextual information, as users won’t know what they’re selecting. Also, don’t use Sliders on small screens without proper touch support or feedback.** |

| Appearance: | |
| --- | --- |
| It's typically used to adjust settings such as volume and brightness or apply filters like price ranges in search queries. Sliders provide a quick and interactive way for users to input values without needing precise numerical entry. | |
| **Visual Design** - It consists of a track (the line that the slider moves along) and a thumb (the control user's drag). Optionally, ticks or labels can indicate specific values along the track. | |
| **Colors and Styles** - They are typically designed with visually appealing, easy-to-distinguish colors that fit within the application's design language. The active part of the track might be highlighted. | |
| **State** | |
| **Range Label** | |
| **Size** | It is scaled appropriately for easy interaction on desktop and mobile devices, ensuring the thumb is large enough to be easily draggable. |

| Behavior: | |
| --- | --- |
| Adjustable Values | Users can adjust the value by moving the thumb along the track, dragging it, or clicking a point on the track. |
| Instant Feedback | Displays the value dynamically as the user moves the slider, either beside the thumb or in a tooltip that appears while adjusting. |
| Range Selection | Some sliders allow users to select a minimum and maximum value by providing two thumbs on the same track. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **No** |
| Can the color be adjusted? | **No just Primary Color and Success Color** |
| Does it support Responsive Design? | **-** |
| Variants: | |
|  | |
| **Single-thumb Slider** | A standard slider with one thumb for selecting a single value. |
| **Range Slider** | This slider features two thumbs to define a range between two values, which is helpful for settings like price filters. |
| User Interaction: | |
| **Dragging** | Users click and drag their thumb to the desired value in the primary interaction mode. |
| **Keyboard Navigation** | It supports using arrow keys for fine adjustments after the slider is focused, with more significant jumps via page up and down keys. |
| **Clicking** | Users can click anywhere on the track to move their thumb directly to that location. |

### **4. Accessibility**

| Keyboard Accessibility | Allows control through keyboard inputs, ensuring users can use the component without a mouse. |
| --- | --- |
| Screen Reader Support | This includes ARIA attributes such as role= "slider," aria-value min, aria-value max, and aria-value now to communicate the slider's function and current state to assistive technologies. |
| Focus Management | This indicator indicates focus on the slider when it is active, using visual cues to aid users who rely on keyboard navigation. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Sliders commonly work with Labels, Tooltips, Text Fields (for editable values), Steppers, Validation Messages, and Progress Bars (to visualize value progression). |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Number Inputs (for precision), Dropdowns (for limited predefined values), or Stepper Buttons (for controlled incremental changes). |
| illustration: | |
|  | |
| In a setting application, a slider might be used to adjust the unit of brightness. Users can drag the slider to change the opacity level, seeing the real-time effect on their video preview. | |

## Stepper

## 

### **1. General Information**

| Component name: | **Stepper** |
| --- | --- |
| Category: | **Interactive Components** |
| Description: | The Stepper component is a navigational element used to guide users through a series of steps in a linear or non-linear process. |
| Objective: | The objective of a Stepper is to show users the sequence of steps in a process—like registration, form filling, or checkout—so they understand where they are and how much is left. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Clearly indicate the number of steps to set proper expectations at the beginning and throughout the process. | ❌Don't allow navigation to a step where mandatory information from a previous step is missing or incorrect. |
|  |  |
| ✅Use concise, clear labels for each step to accurately convey what is required at each stage. | ❌Avoid overwhelming the user with too many steps; consider breaking up complex processes into smaller, manageable parts. |
|  |  |
| ✅Provide users with feedback on their progress and any validation errors that occur. | ❌Don't rely solely on color to indicate status changes; use icons, labels, or patterns to enhance clarity. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Stepper when a task involves multiple clearly defined steps that need to be completed in order—such as multi-page forms or checkout flows.** |
| --- | --- |
| Should not be used when: | **Avoid using a Stepper when the task has only one step or when steps are simple enough that splitting them offers no usability benefit.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **A good Stepper should display clear step numbers or labels, indicate step statuses (completed, current, upcoming), and allow smooth navigation between steps with “Back” and “Next” actions.** |
| ❌Guidelines to avoid: | **Avoid overly complex Steppers with too many minor steps that overwhelm users. Don’t use a Stepper if the steps aren’t clearly distinct or don’t involve different inputs.** |

| Appearance: | |
| --- | --- |
| It's commonly used in multi-step forms, wizards, or configuration setups to indicate progress and visually guide users from start to completion. | |
| **Visual Design** - It is typically displayed as a series of connected dots, numbers, or icons representing each step. Each element can be styled to indicate the current, completed, and upcoming steps. | |
| **States** - Includes states such as active, inactive, complete, and error, each represented by different colors, icons, or fonts. | |
| **Orientation** - Depending on the layout and design requirements, it can be oriented horizontally or vertically. | |

| Behavior: | |
| --- | --- |
| Linear Navigation | Users must complete steps in a specific sequence to return to previous steps if needed. |
| Non-linear Navigation | Users can navigate to any step, which is not strictly sequential and is often used in settings where steps do not depend on the completion of others. |
| Interaction | Clicking or tapping a step can navigate the user to that part of the process, with appropriate validation at each stage. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **Yes** |
| Does it support Responsive Design? | **Yes** |
| Variants: | |
| **Basic Stepper** - Displays steps in a simple line with numbers or icons. | |
| **Wizard Stepper** - Includes more detailed interactions, possibly with transitions and animations between steps. | |
| **Editable Stepper** - Allows users to return to completed steps and modify information before final submission. | |
| User Interaction: | |
| **Step Selection** | Users can select a step either by clicking on it directly (if allowed in non-linear steppers) or by navigating through the process using the "Next" and "Back" buttons. |
| **Progress Feedback** | The stepper provides visual feedback on progress, highlighting which steps have been completed, which are active, and which need to be started. |
| **Error Handling** | Indicates errors in a specific step by altering the visual state of the step (e.g., color change, error icon) and possibly preventing progress until the error is resolved. |

### **4. Accessibility**

| Keyboard Navigation | Supports full keyboard interactions, including navigating between steps and initiating actions associated with each step. |
| --- | --- |
| Screen Reader Support | Properly labels each step with ARIA roles and states, such as aria-current="step" for the active step and aria-completed="true" for finished steps. |
| Focus Management | Manages to ensure that the currently active step and its content are clearly highlighted and accessible via keyboard. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Steppers commonly work with Forms, Buttons (Back/Next/Submit), Progress Bars, Validation Messages, and Modals/Dialogs for overlay-based flows. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Wizards (for more guided experiences), Accordions (to expand/collapse sections), or simple multi-page layouts with breadcrumb navigation. |
| illustration: | |
|  | |
| In a software register form, a stepper displays steps like "Personal Info," "Business Info," "Other Info," and "Completion." The user progresses through each step, with the stepper visually indicating which steps have been completed and which are currently active. | |

### **7. Specification**

Display only the primary color.

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
| State |
|  |
|  |
|  |
| Orientation |
|  |
|  |

## Switch

## 

### **1. General Information**

| Component name: | **Switch** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Switch component, often referred to as a toggle switch, is a user interface control that allows users to change a setting between two states, such as on/off or enabled/disabled. |
| Objective: | The objective of a Switch is to allow users to instantly toggle a setting or feature on or off through a simple and intuitive binary interaction. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Use switches for actions that have an immediate effect without further confirmation. | ❌Don't use switches when the action requires more user input or confirmation (use checkboxes instead). |
|  |  |
| ✅Ensure the visual design is large enough to be easily interacted with, especially on touch devices. | ❌Avoid using switches for actions that do not have a clear on/off or active/inactive state. |
|  |  |
| ✅Clearly label the switch with its function so users understand what setting they are changing. | ❌Don't place switches in locations where they can be easily toggled by accident. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Switch when users need to enable or disable a feature—like notifications, visibility, or system settings—without requiring extra confirmation steps.** |
| --- | --- |
| Should not be used when: | **Avoid using a Switch for critical actions with serious consequences—like deleting data or confirming payments. Use Buttons with confirmation modals or Checkboxes instead.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **A good Switch should have a clear label indicating what it controls. It should visibly show the state (on/off) through color or position, and support accessibility (keyboard, screen reader).** |
| ❌Guidelines to avoid: | **Avoid using Switches without labels or when toggling triggers complex outcomes that require further explanation, as it can confuse users.** |

| Appearance: | |
| --- | --- |
| It is commonly used for binary settings, quick configuration options, and feature toggles, providing an intuitive way for users to control app features. | |
| **Visual Design** - They are typically represented as a horizontal track containing a movable knob or slider that users can drag or click to toggle between states. The design is compact and straightforward, with apparent visual differences between the on and off states. | |
| **Colors and Styles** - This method indicates the status or brand's primary color. | |
| **Feedback** | Provides immediate visual feedback when toggled—the position of the knob and the background color of the track change to reflect the current state. |

| Behavior: | |
| --- | --- |
| Toggle Action | Clicking or dragging the switch changes the state from on to off or vice versa. This action triggers an immediate change in the system or application setting. |
| Instantaneous Update | The state change effected by the switch is immediate, without a submit button. |
| Accessibility | Implements an easily clickable and draggable knob, even on smaller screens, ensuring that the component is touch-friendly. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Custom Design or Native Design** |
| --- | --- |
| Icon supported or not: | **No** |
| Can the color be adjusted? | **No (Primary CI Color and Success Color)** |
| Does it support Responsive Design? | **No** |
| Variants: | |
|  | |
| **Primary Switch** | The simplest form, suitable for most on/off settings. And use Primary color. |
| **Green Switch** | The simplest form, suitable for most on/off settings. And use green color. |
| User Interaction: | |
| **Click to toggle** | Users can click anywhere on the switch to toggle the state. The switch provides clear auditory or tactile feedback if possible. |
| **Keyboard Interactions** | When the switch has focus, these can be toggled using keyboard shortcuts, typically the space bar. |
| **Drag to toggle** | On devices with touch interfaces, users can drag the knob across the track to change the setting. |

### **4. Accessibility**

| Keyboard Navigation | The switch is fully accessible via the keyboard. Users can focus on it using tab navigation and toggle it using space or enter keys. |
| --- | --- |
| Screen Reader Support | Properly labeled with ARIA roles and properties, such as role="switch" and aria-checked="true" or false" depending on the state, to ensure it is announced correctly by screen readers. |
| Focus Indicators | It provides a clear focus state when the switch is navigated via the keyboard, aiding users who rely on visual cues for navigation. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

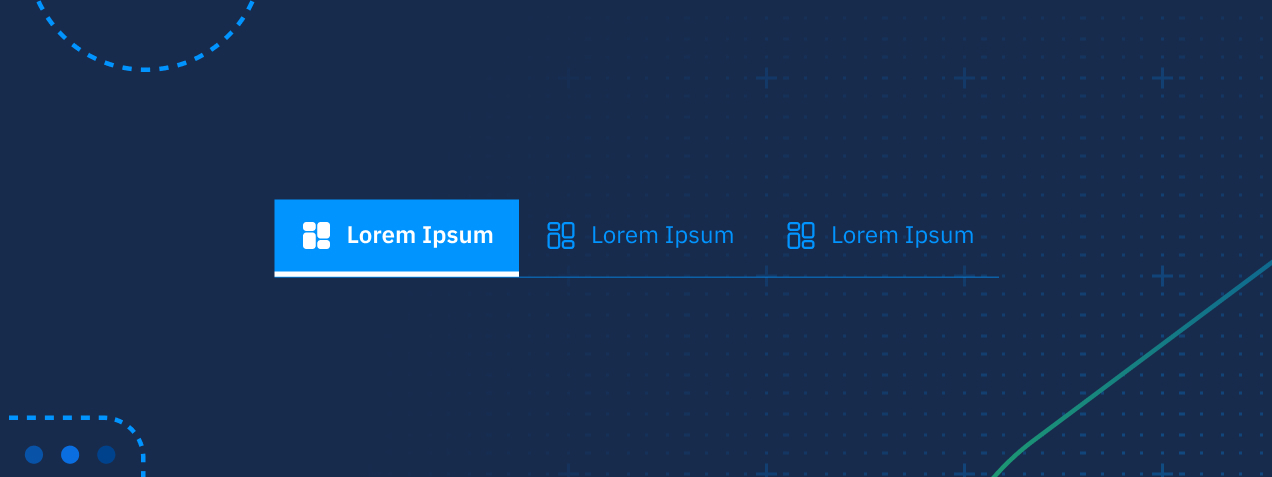
### **6. Related Components**

| Components that are often used together: | Switches commonly work with Labels, Tooltips (for extra info), Form Controls, List Items, or Cards in settings UIs, and may include Icons for added clarity. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Checkboxes (in forms), Radio Buttons (for choosing one of many), or Buttons (when confirmation is needed before toggling a setting). |
| illustration: | |
|  | |
| A mobile app's settings menu switch turns notifications on or off. Users can quickly toggle the switch to change their notification preferences, seeing the immediate effect of their changes without additional steps. | |

### **7. Specification**

| Spacing |
| --- |
|  |
|  |
| State |
|  |
|  |
|  |
|  |
|  |

## Tabs



### **1. General Information**

| Component name: | **Tabs** |
| --- | --- |
| Category: | **Navigation Components** |
| Description: | The Tabs component is a fundamental UI element used for navigating between different views or sections within an application. |
| Objective: | The objective of Tabs is to let users switch between multiple related content sections within the same view, without navigating to a new page. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Ensure all tabs are easily accessible for touch, mouse, and keyboard users. | ❌(Attached image of poorly designed UI Component) |
|  |  |
| ✅Use clear and concise labels for each tab to communicate the content they represent effectively. | ❌Avoid using tabs for unrelated content groups; instead, tabs should organize content belonging to the same hierarchical level or category. |
|  |  |
| ✅Provide a visual indicator for the active tab, such as a highlight or an underline. | ❌Don't change the order of tabs dynamically, as this can disorient users who rely on consistent navigation paths. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use Tabs when displaying multiple related sections of content—like user profiles, product details, or grouped settings—where users benefit from quick switching without leaving the page.** |
| --- | --- |
| Should not be used when: | **Avoid using Tabs when each section has very different content or when the flow is sequential, like in a checkout process—use a Stepper or Wizard instead.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Tabs should have clear, concise labels; clearly indicate the selected tab; support keyboard navigation; and load content responsively when switching tabs.** |
| ❌Guidelines to avoid: | **Avoid long or vague tab labels, nested tabs that add complexity, and using Tabs when the content is incomplete or loads too slowly after clicking.** |

| Appearance: | |
| --- | --- |
| It is typically positioned at the top or bottom of a page and contains multiple tabs that can be activated to display associated content without reloading the page. | |
| **Visual Design** - It consists of a series of tabs, each representing different content sections. The active tab is usually highlighted or underlined to distinguish it from inactive tabs. | |
| **Responsive Design** - Adapts to different screen sizes, ensuring tabs are accessible and visually consistent across devices. | |
| **Customization** - Supports various styles, including icons and text labels, to clearly communicate each tab's purpose. | |

| Behavior: | |
| --- | --- |
| Switching Tabs | Clicking on a tab immediately brings its associated content into view, usually without a page refresh, making the interaction quick and seamless. |
| State Persistence | Switching back and forth maintains the state of previously interacted tabs unless specifically designed to refresh. |
| Keyboard Navigation | Allows users to navigate between tabs using arrow keys when the tabs has focus. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **[Small / Medium / Large or other]** |
| --- | --- |
| Icon supported or not: | **[Yes / No]** |
| Can the color be adjusted? | **[Yes / No / Color adjustment conditions]** |
| Does it support Responsive Design? | **[Yes / No / What are the limitations?]** |
| Variants: | |
| **Icon Tabs** - Use icons instead of text to save space or enhance the visual appeal. | |
| **Scrollable Tabs** - Allow horizontal scrolling to accommodate many tabs on smaller screens. | |
| User Interaction: | |
| **Tab Selection** | Users can select a tab by clicking or tapping on it. The content associated with the tab is displayed instantly. |
| **Accessibility** | Provides full keyboard and screen reader support, including appropriate ARIA roles and properties to enhance user accessibility. |
| **Feedback** | Offers visual and possibly auditory feedback when changing tabs to reinforce the interaction. |

### **4. Accessibility**

| Focus Management: | Ensures that navigating through tabs is straightforward and logical, with clear focus indicators. |
| --- | --- |
| Screen Reader Support | Uses roles such as role="tablist", role="tab", and role="tabpanel" to define the structure. Attributes like aria-selected and aria-controls link tabs with their corresponding content. |
| Readable Labels | Each tab is clearly labeled, and icons are accompanied by text or tooltips to ensure understanding, regardless of the user's ability to interpret visual cues. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Tabs often work with Cards, Content Panels, Icons, Badges (to show counts), Tooltips, and Scroll Areas (when tabs overflow horizontally). |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Accordions (for expanding/collapsing content), Segmented Controls (for short content toggles), or separate pages when each section is complex. |
| illustration: | |
|  | |
| In a productivity application, the tabs might include tabs labeled "Email," "Calendar," "Contacts," and "Tasks." Selecting each tab displays the relevant application section, allowing users to switch between communication and organization forms quickly. | |

### **7. Specification**

| Tabs (Horizontal) |
| --- |
|  |
|  |
|  |
|  |
| Tabs (Vertical) |
|  |
|  |
|  |
| Properties |
| State |
|  |
|  |
|  |
|  |
| Direction |
|  |
|  |

## Table

## 

### **1. General Information**

| Component name: | **Table** |
| --- | --- |
| Category: | **Data Display Components** |
| Description: | The Table component is used to display data in a structured format of rows and columns. |
| Objective: | The objective of a Table is to present large amounts of structured data in rows and columns, allowing users to compare values across different entries efficiently. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Use clear, concise headers to describe the content of each column. | ❌Don’t overload the table with too many columns or rows, making it difficult to read and navigate. |
|  |  |
| ✅Ensure the table is responsive, maintaining usability on all devices. | ❌Avoid using tables for layout purposes; they should be used strictly for displaying data. |
|  |  |
| ✅Implement sorting, filtering, and pagination to handle large datasets efficiently. | ❌Don’t rely solely on color to convey information; use text, icons, or patterns to enhance accessibility. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Table when you need to display multiple items with multiple attributes—such as user lists, order records, or financial data—where row-and-column comparison is important.** |
| --- | --- |
| Should not be used when: | **Avoid using Tables when the dataset is small, content is visual-heavy, or comparison isn’t necessary—use cards or simple lists instead.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **A good Table should have clear headers, support sorting and filtering, allow search within rows, and render responsively for both desktop and mobile (e.g., with scroll or stacking).** |
| ❌Guidelines to avoid: | **Avoid overly dense tables without spacing, logical grouping, or helpful features like fixed headers, row highlighting, or column alignment for easier reading.** |

| Appearance: | |
| --- | --- |
|  | |
| **Headers** | Typically includes a header row at the top with labels for each column, providing context for the data. |
| **Rows and Columns** | It consists of multiple rows and columns, each representing a data record and each representing a data field. |
| **Borders and Gridlines** | Borders and gridlines separate rows and columns, enhancing readability. |

| Behavior: | |
| --- | --- |
| Sorting | Columns can often be sortable, allowing users to reorder data by clicking on the column headers. |
| Pagination | For large datasets, tables can be paginated, displaying a subset of rows per page with navigation controls to move between pages. |
| Filtering | Provides filtering options to allow users to narrow down the data displayed based on specific criteria. |
| Responsive Design | Adapts to different screen sizes, possibly converting to a stacked layout on smaller screens to maintain usability. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **-** |
| Does it support Responsive Design? | **-** |
| Variants: | |
|  | |
| Header | The top section of the table displays column titles for easy data identification. |
|  | |
| Body | The main section of the table contains rows of data entries. |
|  | |
| Input | A table cell designed for user data entry, often featuring text fields or other input controls. |
|  | |
| Selection Control | Table elements like checkboxes or radio buttons allow users to select one or more rows. |
|  | |
| Avatar | A table cell includes a small image or icon representing a user or item. |
|  | |
| Button | A table element featuring interactive buttons for actions like editing or deleting rows. |
|  | |
| Chip | A compact element within a table cell that displays information such as tags or status indicators. |
|  | |
| Index | A numbered column that provides a sequential reference for each row in the table. |

### **4. Accessibility**

| Keyboard Navigation | Fully navigable using the keyboard, allowing users to move between cells, rows, and columns using arrow keys. |
| --- | --- |
| Responsive Design | Ensures the table is accessible on all devices, with features like adjustable column widths and stacking on smaller screens. |
| ARIA Roles and Properties | Use appropriate ARIA roles such as role="table," role="row," and role="cell" to enhance screen reader support. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Tables often work with Pagination, Search Fields, Filters, Checkboxes (for row selection), Action Buttons, Tooltips, Tags/Badges (for statuses), and Loading Indicators. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Card Grids (for visual data), List Views (for vertical reading), or Accordions (to expand/collapse row-level details). |
| illustration: | |
|  | |
| A Table component displays a list of tasks in a project management application. | |

## Text Input

### **1. General Information**

| Component name: | **Text Input** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Text Input component is a fundamental UI element used for capturing user input in the form of text. |
| Objective: | The objective of a Text Input is to allow users to enter freeform text—such as names, addresses, search terms, or open-ended responses. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Use labels consistently and position them near the input fields to improve clarity and accessibility. | ❌Don't rely on placeholder text as the only method of providing instructions or labels. |
|  |  |
| ✅Provide clear and helpful placeholder text that disappears when the user begins to type. | ❌Avoid using small input fields that make typing difficult, especially on mobile devices. |
|  |  |
| ✅Implement responsive design to ensure text inputs are usable across all devices and screen sizes. | ❌Don't overcrowd forms with too many text inputs; aim for a balance between comprehensive data collection and user fatigue. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Text Input when users need to type personalized or unique information—such as in forms, search fields, or custom settings.** |
| --- | --- |
| Should not be used when: | **Avoid using Text Inputs when the input must come from a limited set of values—like categories, countries, or statuses. Use Dropdowns or Selects instead.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Provide a clear label, a helpful placeholder, ensure smooth keyboard input, and display error messages for validation failures.** |
| ❌Guidelines to avoid: | **Avoid using unlabeled inputs, unclear or missing validation, overly strict character limits without notice, or enforcing formats without examples.** |

| Appearance: | |
| --- | --- |
| It is versatile and essential for forms, search bars, and any situation requiring free-form data entry. Text inputs allow users to enter and edit text, numbers, or special characters based on the application's specific needs. | |
| **Visual Design** - Typically, it appears as a rectangular box with a border that may change color to indicate focus or error states. The design is simple yet functional, ensuring it integrates seamlessly with various UI styles. | |
| **Placeholder Text** - It often includes placeholder text within the input field to provide hints or examples of the required input, which disappears once the user begins typing. | |
| **Size and Padding** - Adequately sized for easy interaction, with sufficient padding to ensure that text remains legible and accessible. | |

| Behavior: | |
| --- | --- |
| Input Types | It supports various input types, including 'text,' 'password,' 'email,' etc., to cater to data entry needs and enhance form validation. |
| Validation | Can validate input on the client side, providing immediate feedback to users about errors or requirements (like format and length). |
| Enhancements | These may include features like clear buttons, visibility toggles for passwords, or autocomplete functionality to enhance usability. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **Yes** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **No** |
| Variants: | |
| **Single-Line Text Input** - Standard for most data entry tasks. | |
| User Interaction: | |
| **Entering Data** | Users can type directly into the input field, with the text being displayed or obscured depending on the type (e.g., passwords are masked). |
| **Accessibility** | Enhancements such as labels, aria-labels, and roles ensure accessibility for all users, including those using screen readers. |
| **Focus and Blur** | Gains focus when clicked or tabbed into and loses focus when the user clicks away or tabs out, potentially triggering validation. |

### **4. Accessibility**

| Keyboard Navigation | It is fully accessible via the keyboard, allowing users to tab into and out of the text input. |
| --- | --- |
| Screen Reader Support | Proper labeling is crucial so screen readers can accurately describe the function and status of the input field. |
| Error Handling | Accessibility features include providing error messages that are not solely conveyed by color changes and ensuring they are readable by screen readers. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Text Inputs commonly work with Labels, Icons (e.g., search or clear), Helper Texts, Error Messages, Tooltips, Form Validation, and Autocomplete. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Select (for predefined options), Radio Buttons (for limited choices), or Sliders/Steppers (for numeric values with control). |
| illustration: | |
|  | |
| In a customer feedback form, text inputs enter a name, email, and a short message. Each field includes a label above the input and placeholder text within, guiding users on what information to enter. | |

### **7. Specification**

| Spacing |
| --- |
|  |
|  |
|  |
|  |
| Properties |
| State |
|  |
|  |
|  |
|  |
| Error |
|  |
|  |
| Disabled |
|  |
|  |

## Text Area

### **1. General Information**

| Component name: | **Text Area** |
| --- | --- |
| Category: | **Form & Input Components** |
| Description: | The Text Area component is a versatile input field designed for capturing multi-line text input from users. |
| Objective: | The objective of a Text Area is to allow users to input multi-line text—such as comments, descriptions, or open-ended responses. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Clearly label the text area, positioning the label close to the input field to improve accessibility and usability. | ❌Don’t rely solely on placeholder text for instructions; always include a visible label. |
|  |  |
| ✅Use placeholder text to provide examples or hints about the expected input, enhancing user guidance. | ❌Avoid making the text area too small, making it difficult for users to enter and review their text. |
|  |  |
| ✅Ensure the text area is sufficiently large to accommodate the expected amount of text without excessive scrolling. | ❌Please don’t turn off the resize functionality unless necessary, as users may need to adjust the viewable area to suit their preferences. |

### 

### **2. Usage Guidelines**

| It should be used when: | **Use a Text Area when users are expected to write long or detailed input—such as feedback forms, product reviews, additional notes, or multi-line content.** |
| --- | --- |
| Should not be used when: | **Avoid using a Text Area for very short input—such as names or passwords—where a standard Text Input would be more appropriate.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Provide a clear label, a helpful placeholder or helper text explaining what to write, and ensure the area can scroll or resize appropriately for longer content.** |
| ❌Guidelines to avoid: | **Avoid overly small Text Areas that hinder reading/writing, omitting character limits when needed, or failing to show validation errors.** |

| Appearance: | |
| --- | --- |
| It is ideal for collecting comments, messages, descriptions, and other lengthy text content. Unlike single-line text inputs, text areas provide ample space for users to enter and view more significant amounts of text. | |
| **Visual Design** - Typically, a rectangular box larger than standard text input fields is designed to expand vertically as more text is entered. It includes a border and may display scroll bars when the text exceeds the visible area. | |
| **Placeholder Text** - It can include placeholder text to guide users on what to enter, which disappears when they start typing. | |
| **Resizable** - It often includes a handle at the bottom-right corner, allowing users to resize the text area manually. | |

| Behavior: | |
| --- | --- |
| Multi-line Input | This input method supports the entry of multiple lines of text, automatically wrapping the text to the following line as the user types. |
| Auto-Expanding | Some text areas can dynamically adjust their height based on the content, removing the need for scroll bars. |
| Scroll Bars | If the content exceeds the visible area, vertical and horizontal scroll bars are displayed, ensuring all text can be accessed. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **No** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **Yes** |
| Variants: | |
| **Standard Text Area** - An essential multi-line text input field with a scroll bar for overflow content is needed. | |
| **Auto-Expanding Text Area** - Dynamically adjusts height based on content, providing a more streamlined user experience. | |
| **Rich Text Area** - Includes formatting tools for bold, italics, lists, etc., allowing users to style their text. | |
| User Interaction: | |
| **Typing** | Users can enter and edit text directly within the text area, and they can create new lines using the Enter key. |
| **Resizing** | If enabled, users can click and drag the resize handle to adjust the size of the text area for better visibility of the content. |
| **Copy-Paste** | This feature supports standard clipboard operations, allowing users to copy and paste text into and out of the text area. |

### **4. Accessibility**

| Keyboard Navigation | The text is fully navigable via the keyboard, allowing users to move through it using arrow keys and select text with shift and arrow keys. |
| --- | --- |
| Screen Reader Support | Properly labeled with aria-labels or aria-labelledby to ensure the purpose and content of the text area are communicated to users of assistive technologies. |
| Focus Indicators | Provides clear visual indicators when the text area is focused, enhancing accessibility for users relying on keyboard navigation. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Text Areas typically work with Labels, Helper Texts, Error Messages, Character Counters, Tooltips, and sometimes Submit or Preview Buttons. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Text Inputs (for short text), Dropdowns with freeform entry (for structured + custom values), or Rich Text Editors (for formatted content). |
| illustration: | |
|  | |
| A text area is used for the post's main content in a blog post submission form. The text area is large and auto-expanding as the user types, allowing them to input and review their content without excessive scrolling. | |

### **7. Specification**

|  |
| --- |
|  |
| State |
|  |
|  |
|  |
|  |
|  |
|  |

## Tooltip

## 

### **1. General Information**

| Component name: | **Tooltip** |
| --- | --- |
| Category: | **Display & Feedback Components** |
| Description: | Tooltips are small, informative text boxes that appear when a user hovers over or focuses on an element. |
| Objective: | The objective of a Tooltip is to provide brief additional information or descriptions about a UI element without permanently occupying space on the screen. |

### 

| illustration: | |
| --- | --- |
| ✅ | ❌ |
| ✅Use for brief, supplementary information that enhances understanding. | ❌Avoid using tooltips for information crucial to completing a task; not all users will see or be able to access tooltips. |
|  |  |
| ✅Ensure the text is concise and directly related to the element it describes. | ❌Do not overcrowd the interface with tooltips; use them sparingly and intentionally. |

### **2. Usage Guidelines**

| It should be used when: | **Use a Tooltip when explaining icons, buttons, or technical terms that users might not understand at first glance, especially without disrupting the main layout—e.g., for settings buttons or abbreviations.** |
| --- | --- |
| Should not be used when: | **Avoid using Tooltips for essential content or on mobile-only interfaces (which lack hover), and never use them as a replacement for primary labels in forms.** |
| Good guidelines for design (Best Practices): | |
| ✅Correct approach: | **Tooltips should appear on hover or focus, dismiss automatically when the pointer moves away, contain brief and clear text, and not block key elements.** |
| ❌Guidelines to avoid: | **Avoid placing lengthy content in Tooltips, stacking multiple Tooltips, or designing UIs that rely on Tooltips for core usability.** |

| Appearance: | |
| --- | --- |
| Tooltips are small, informative text boxes that appear when a user hovers over or focuses on an element. They provide additional context, help, or guidance about the component or its function, enhancing user understanding without cluttering the interface. | |
| **Colors** | **Background:** Typically, it is a neutral color to contrast with the text, such as black or deep gray. **Text:** Light colors like white or light gray for readability against the dark background. |
| **Typography** | Small, legible font size and style that matches AXIO design system for consistency |
| **Shape and Layout** | Rounded corners with a short, descriptive message. Optionally includes an arrow pointing to the target element. |

| Behavior: | |
| --- | --- |
| Activation | - It appears on hover or focuses on the target element for desktop users.  - For touch devices, it may appear on tap if necessary. |
| Position | Based on available screen space, it is dynamically positioned relative to the target element (above, below, to the side, etc.). |
| Delay | A brief delay before showing and hiding prevents accidental activation and allows for deliberate user action. |
| Dismissal | Automatically disappears when the cursor or focus moves away from the element. |

### **3. Properties & Customization**

| Supported sizes (Size Options): | **Normal** |
| --- | --- |
| Icon supported or not: | **No** |
| Can the color be adjusted? | **No** |
| Does it support Responsive Design? | **No** |

### **4. Accessibility**

| Keyboard Navigation | Accessible via keyboard navigation, ensuring that all users can trigger tooltips. |
| --- | --- |
| Screen Reader Support | Properly use ARIA attributes (such as aria-describedby) to make the tooltip content accessible to screen readers. |
| Contrast Ratio | High contrast between the text and background colors, meeting WCAG guidelines for text visibility. |

### **5. Common Issues & Troubleshooting**

| **problem** | **How to fix** | **illustration** |
| --- | --- | --- |
| [Explain the problems encountered] | [Solutions] | (Attached sample picture) |

### **6. Related Components**

| Components that are often used together: | Tooltips commonly pair with Icon Buttons, Text Links, Form Labels, Badges, Chart Data Points, and Switches to provide contextual help or descriptions. |
| --- | --- |
| Components that can be used interchangeably: | Alternatives include Helper Text (always visible), Modals/Dialogs (for extended help), or Inline Text permanently placed within the layout. |
| illustration: | |
|  | |
| Imagine a user navigating information with various icons. When the user hovers over an information icon next to data, a tooltip appears, providing a brief explanation of how the metric is calculated. This enhances understanding without the need to navigate away or search for documentation. | |

### **7. Specification**

|  |
| --- |
|  |
|  |
|  |