Buttons (H1)

**Intro text:**

Button elements draw attention to events to let users know what will happen next and give users a choice of options. Button labels clearly tell a user what action will be triggered when the button is used.

**Default button (H2)**

Default buttons help users execute primary actions on a page.

**(HTML/CSS – PRIMENG)**

**(Code snippets)**

**When to use** **(H2)**

* Use the button component to help users carry out an action like starting an application, saving their information, or expand and collapse accordions.
* Use this button for:
  + Submit
  + Save and continue
  + Next page

For color requirements: (H3)  
Go to: Branding link

**When to not use** **(H2)**

* Do not use buttons in place of links. Use links for link and style to look like a button in CSS.

**Guidelines (H2)**

Drawer – Accessibility for buttons

**Accessibility (H3)**

Make button labels concise and clear enough to indicate the action the user will take. Use strong action verbs or phrases that clearly indicate action (for example: "submit," "register," "sign up," "donate," etc.)

When using icons as buttons, use [aria-label] or [aria-labelledby] to give the button an accessible name (for example: an arrow icon for a next page button).

When using an icon in addition to text within a button, make the icon hidden to screen readers using [aria-hidden=”true”]. The additional icon image is redundant if a button is already labelled.

For buttons that expand or collapse content, use [aria-expanded] attribute on the button to let screen reader users know if an accordion’s state is expanded (“true”) or collapsed (“false”).

Avoid using disabled buttons, especially in forms. Disabled buttons provide no feedback, making it hard to know why the button isn't usable. Instead, use validation and errors to show what needs to be done. If you need to have a disabled button, use the aria-disabled attribute to notify screen reader users of the button’s existence. Additional styling and event listeners will need to be created in conjunction to ensure the button is disabled.

Do not rely only on the use of color to convey the seriousness of the action. Not all users can distinguish color or know what it signifies (warning buttons are often red). Instead, rely on a combination of color, context, and button text to make clear what will happen if the user clicks the button.

Drawer – Best practices for buttons

* Overview (H3):
* Use one button style for the initial action and the warning button for the final confirmation.
* Prioritize the most important actions. Too much action on a page can confuse users.
* Use consistent locations for button positioning on the interface.
* Buttons should be no smaller than 44px by 44px. This is a comfortable touch point size for both desktop and mobile devices
* Button label (H3)
* Avoid long, redundant button labels. Drop unnecessary articles, such as *a* or *the*, for a more concise label. Use a verb and a noun (for example: *start survey*).
* Write buttons in sentence case. Capitalize only the first word unless using a proper noun.
* Patterns that use this component: (H3)
* Forms
* Modal dialog
* Drop-down menu

**Secondary button (H2)**

Secondary buttons help users execute supporting actions related to a primary button. Examples of secondary buttons include *Back* and *Cancel*. Secondary buttons use more subtle colors because they should not be as noticeable as the primary button.

**(HTML/CSS – PRIMENG)**

**(Code snippets)**

**Warning or danger button (H2)**  
Warning buttons are designed to make users think carefully before taking a destructive action (for example: permanently deleting an account). They only work if used very sparingly. Most services should not need one. When needed, include an additional step asking users for final confirmation.

**(HTML/CSS – PRIMENG)**

**(Code snippets)**

**Disabled button (H2)**Disabled buttons are greyed out and unavailable. Users cannot interact with them and should understand they cannot do the action associated with them. Use disabled buttons very sparingly because of their inherent issues with accessibility. It is better to keep a button enabled and show an error message explaining why the user cannot proceed.

**(HTML/CSS – PRIMENG)**

**(Code snippets)**

**Accessibility for disabled buttons (H3)**

* Disabled buttons block users without informing them what is wrong, so avoid using [disabled], especially for form submissions.
* Disabled buttons generally have poor color contrast and can confuse users who are visually impaired.
* Use [aria-disabled=”true”] to make the button disabled and visible to screen readers. This helps the button receive focus on tab and notifies the screen reader user of its existence.
* The [aria-disabled] attribute will require both styling in CSS and JavaScript/TypeScript to toggle whether the element is enabled or disabled. [Mozilla article on the aria-disabled attribute](https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA/Attributes/aria-disabled)

*Last updated xx/xx/xxxx*