Bootstrap 5 Documentation

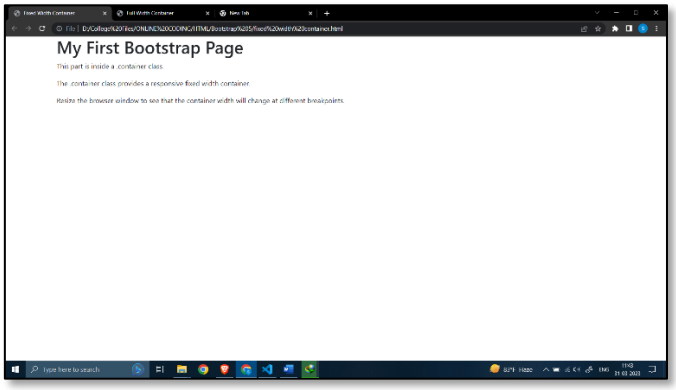
Containers

Containers are used to pad the content inside of them, and there are two container classes available:

- The **.container** class provides a responsive fixed width container
- The **.container-fluid** class provides a full width container, spanning the entire width of the viewport

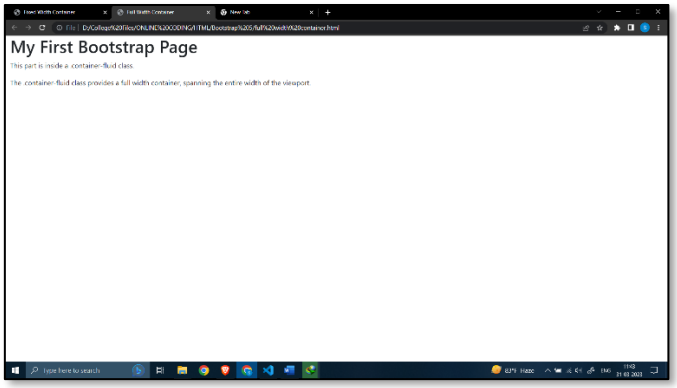
Fixed Container

Use the **.container** class to create a responsive, fixed-width container. Note that its width (max-width) will change on different screen sizes:



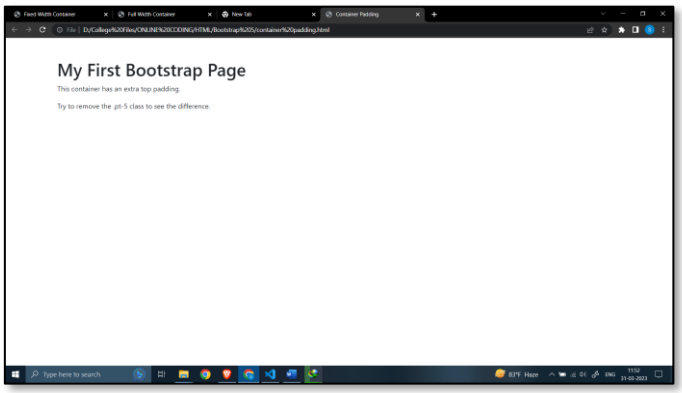
Fluid Container

Use the .container-fluid class to create a full width container, that will always span the entire width of the screen (width is always 100%):

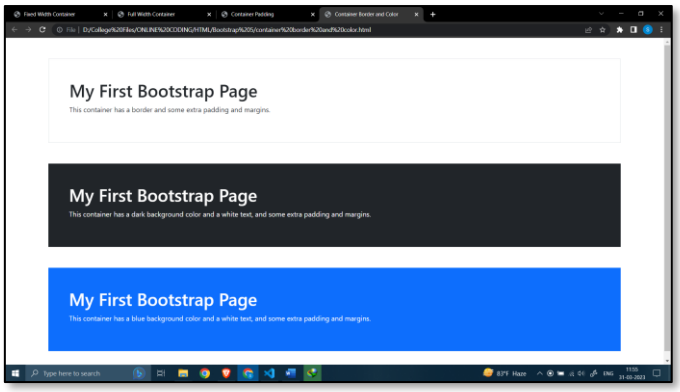


Container Padding

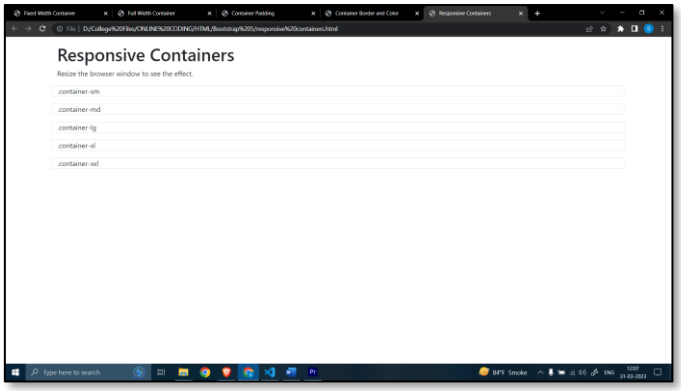
By default, containers have left and right padding, with no top or bottom padding. Therefore, we often use spacing utilities, such as extra padding and margins to make them look even better:



Other utilities, such as borders and colors, are also often used together with containers:



You can also use the **.container-sm/md/lg/xl** classes to determine when the container should be responsive. The max-width of the container will change on different screen sizes/viewports:



Grid System

Bootstrap's grid system is built with flexbox and allows up to 12 columns across the page. If you do not want to use all 12 columns individually, you can group the columns together to create wider columns. The grid system is responsive, and the columns will re-arrange automatically depending on the screen size. Make sure that the sum adds up to 12 or fewer (it is not required that you use all 12 available columns).

span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1
span 4			span 4			span 4					
span 4			span 8								
span 6						span 6					
span 12											

The Bootstrap 5 grid system has six classes:

- **.col** (extra small devices - screen width less than 576px)
- **.col-sm** (small devices - screen width equal to or greater than 576px)
- **.col-md** (medium devices - screen width equal to or greater than 768px)
- **.col-lg** (large devices - screen width equal to or greater than 992px)
- **.col-xl** (xlarge devices - screen width equal to or greater than 1200px)
- **.col-xxl** (xxlarge devices - screen width equal to or greater than 1400px)

The classes above can be combined to create more dynamic and flexible layouts. Each class scales up, so if you want to set the same widths for sm and md, you only need to specify sm.

The following examples demonstrate different applications of rows

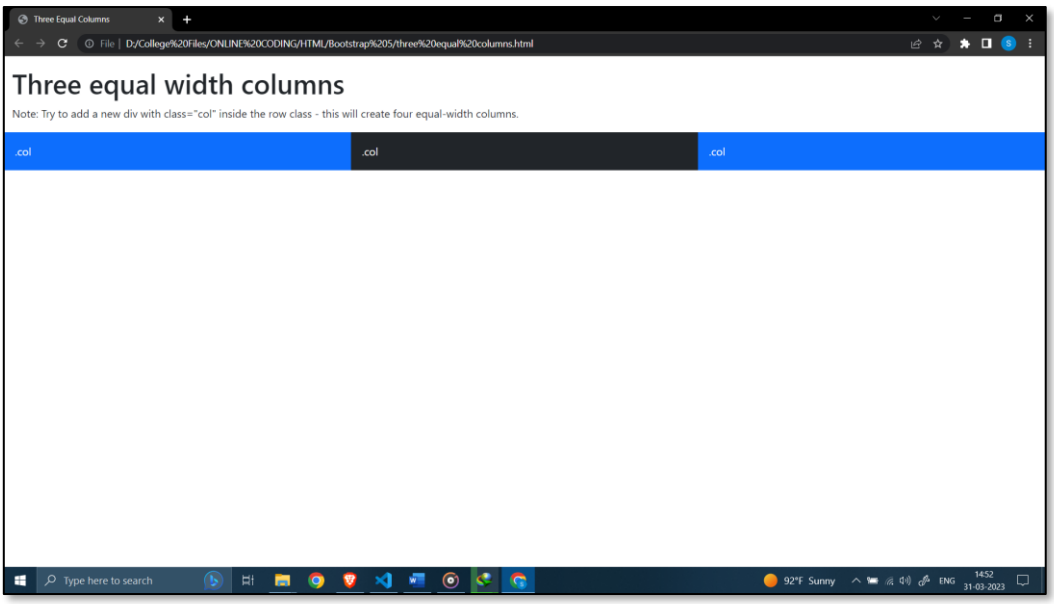
```
<!-- Control the column width, and how they should appear on different devices -->

<div class="row">

    <div class="col-*-*"></div>
    <div class="col-*-*"></div>

</div>

<div class="row">

    <div class="col-*-*"></div>
    <div class="col-*-*"></div>
    <div class="col-*-*"></div>

</div>

<!-- Or let Bootstrap automatically handle the layout -->
<div class="row">

    <div class="col"></div>
    <div class="col"></div>
    <div class="col"></div>

</div>
```

First example: create a row ( ). Then, add the desired number of columns (tags with appropriate .col-*-* classes). The first star (*) represents the responsiveness: sm, md, lg, xl or xxl, while the second star represents a number, which should add up to 12 for each row.

Second example: instead of adding a number to each col, let bootstrap handle the layout, to create equal width columns: two "col" elements = 50% width to each col, while three cols = 33.33% width to each col. Four cols = 25% width, etc. You can also use .col-sm|md|lg|xl|xxl to make the columns responsive.

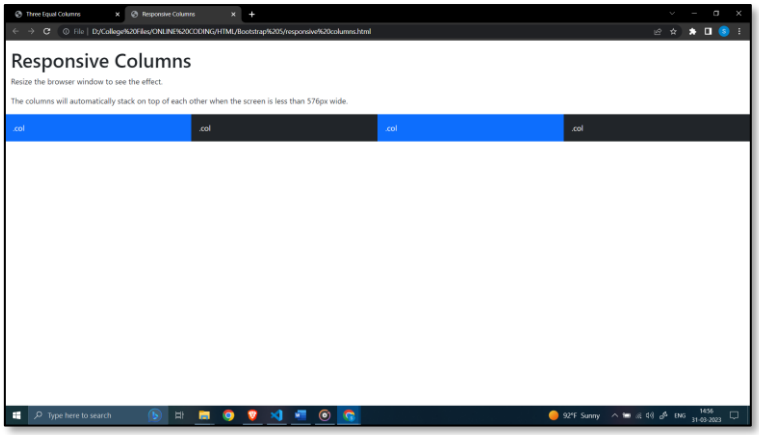
Three Equal Columns

The following example shows how to create three equal-width columns, on all devices and screen widths:



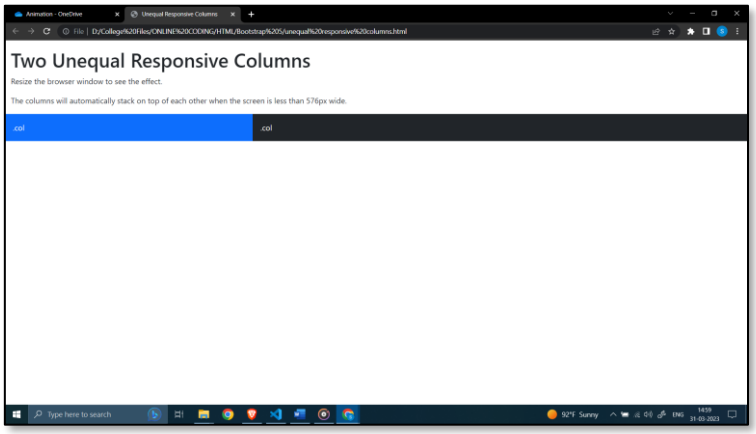
Responsive Columns

The following example shows how to create four equal-width columns starting at tablets and scaling to extra-large desktops. On mobile phones or screens that are less than 576px wide, the columns will automatically stack on top of each other:



Two Unequal Responsive Columns

The following example shows how to get two various-width columns starting at tablets and scaling to large extra desktops:

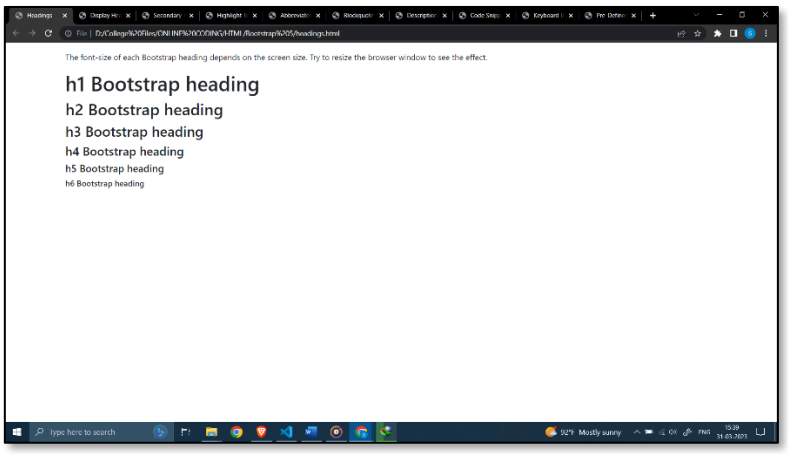


Typography

Bootstrap 5 uses a default font-size of 1rem (16px by default), and its line-height is 1.5. In addition, all <p> elements have margin-top: 0 and margin-bottom: 1rem (16px by default).

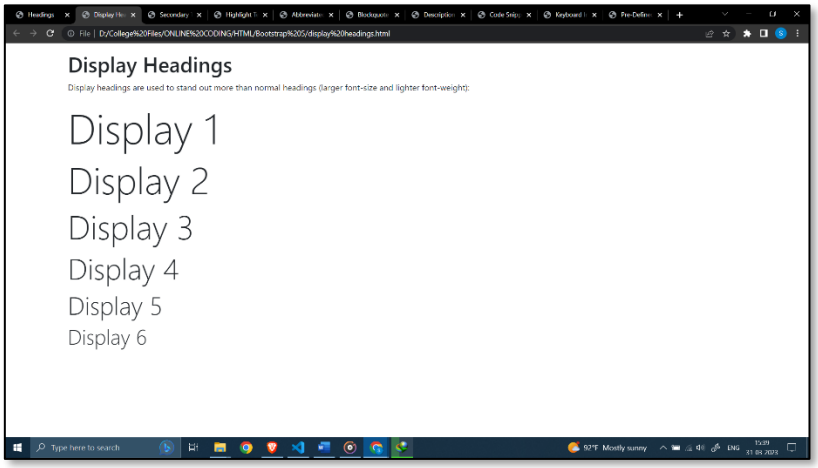
Headings

Bootstrap 5 styles HTML headings (<h1> to <h6>) with a bolder font-weight and a responsive font-size. You can also use .h1 to .h6 classes on other elements to make them behave as headings if you want:



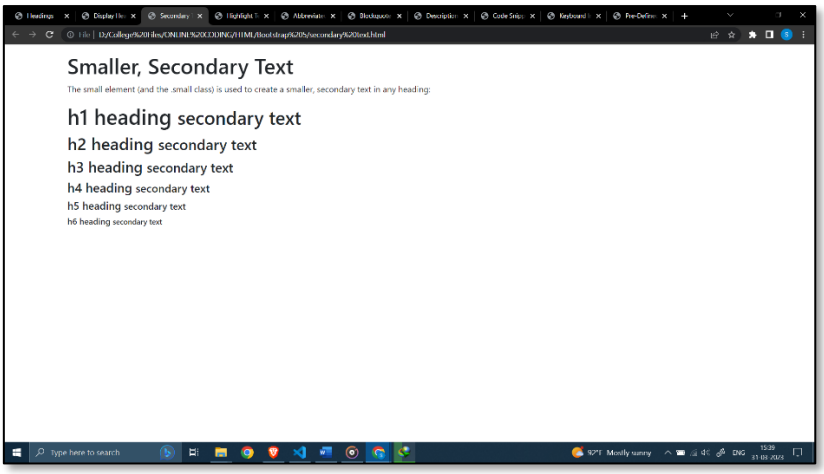
Display Headings

Display headings are used to stand out more than normal headings (larger font-size and lighter font-weight), and there are six classes to choose from which are **.display-1** to **.display-6**:



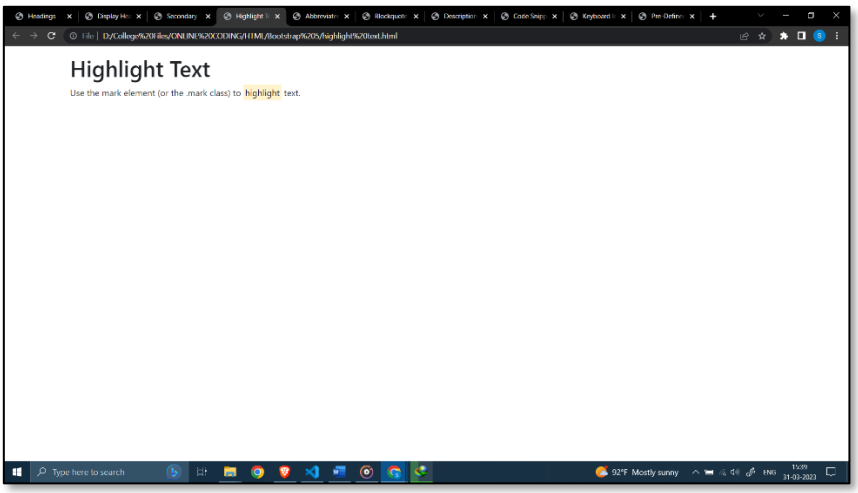
<small>

In Bootstrap 5 the HTML <small> element (and the .small class) is used to create a smaller, secondary text in any heading:



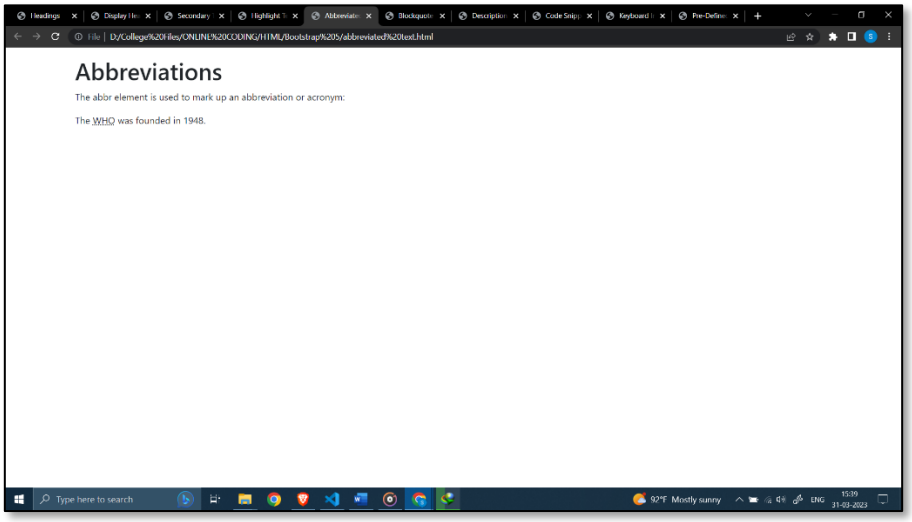
<mark>

Bootstrap 5 will style <mark> and .mark with a yellow background color and some padding. Use the mark element to highlight text:



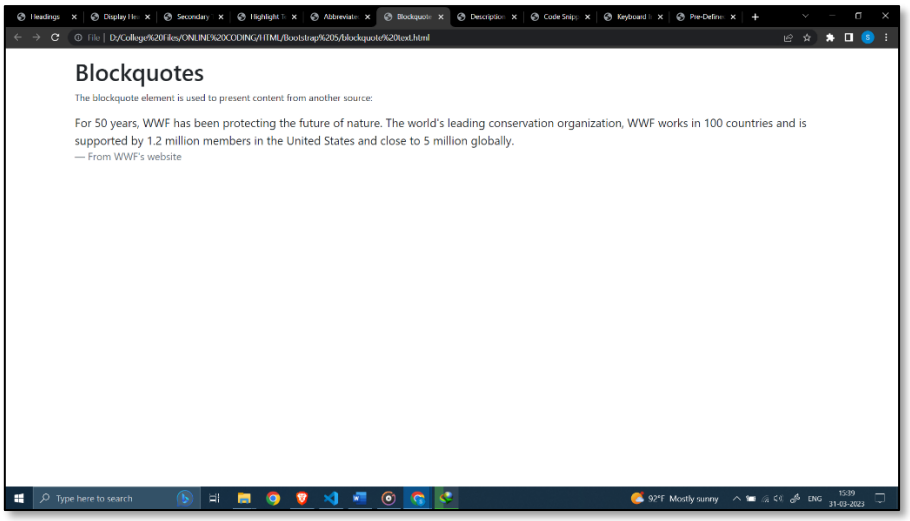
<abbr>

Bootstrap 5 will style the HTML <abbr> element with a dotted border bottom and a cursor with question mark on hover:



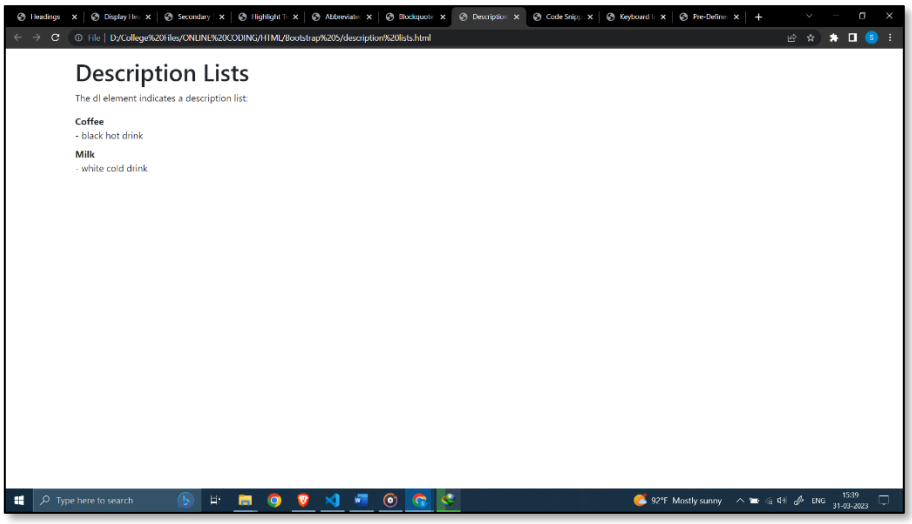
<blockquote>

Add the .blockquote class to a <blockquote> when quoting blocks of content from another source. And when naming a source, like "from WWF's website", use the .blockquote-footer class:



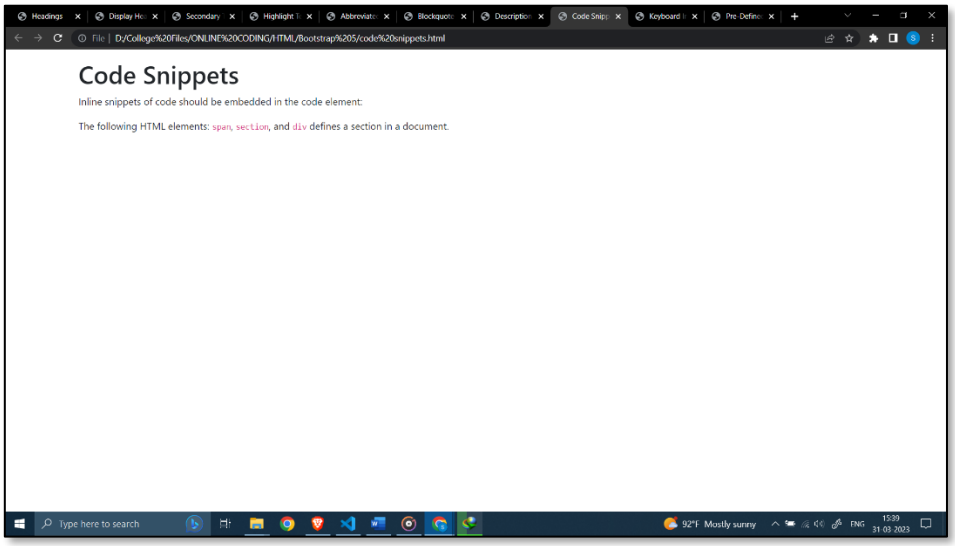
<dl>

Bootstrap 5 will style the HTML <dl> element in the following way:



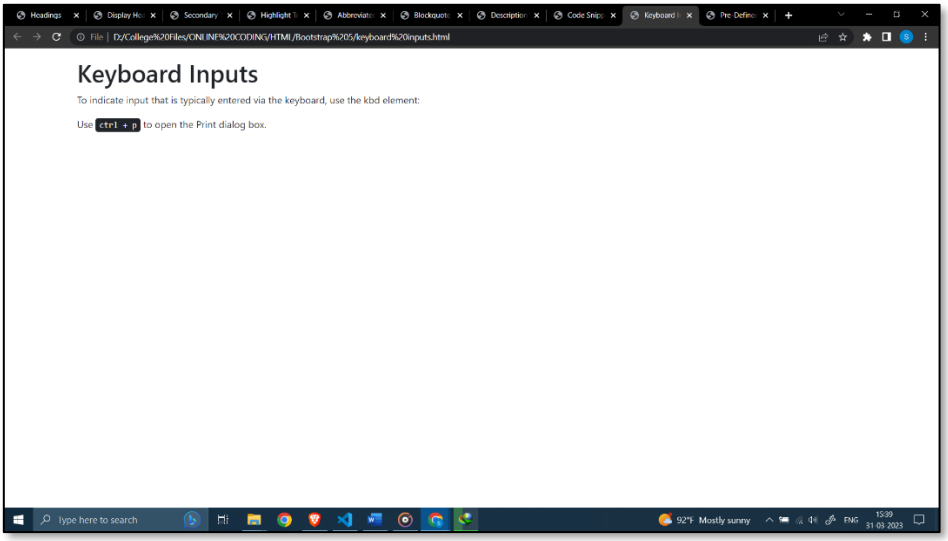
<code>

Bootstrap 5 will style the HTML <code> element in the following way:



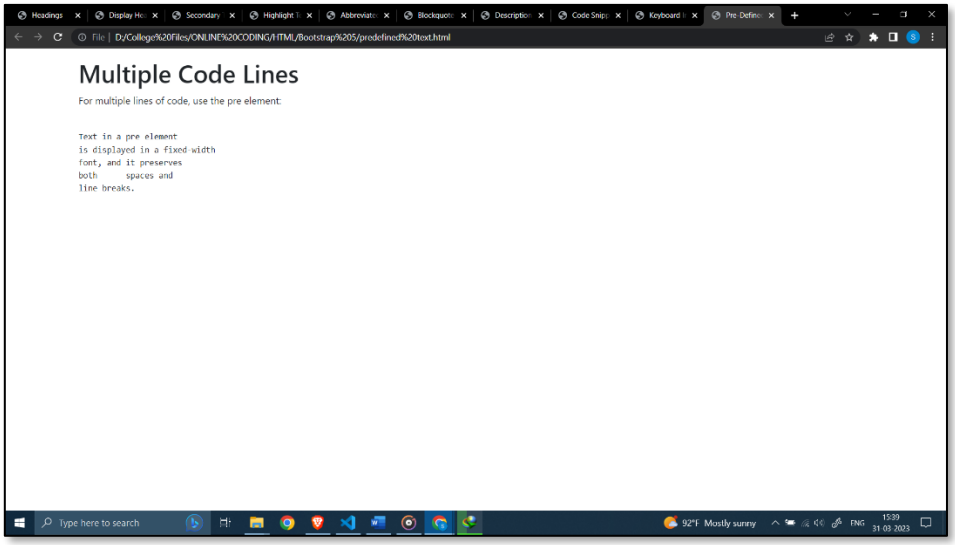
<kbd>

Bootstrap 5 will style the HTML <kbd> element in the following way:



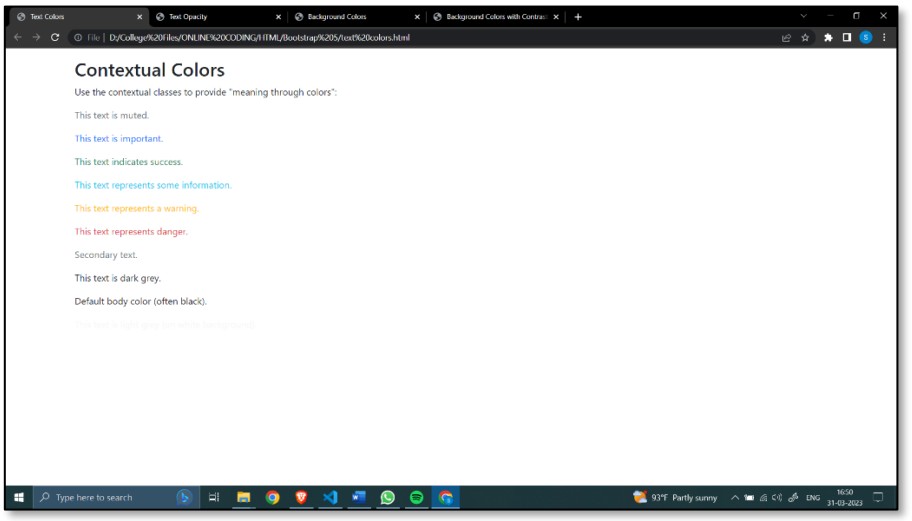
<pre>

Bootstrap 5 will style the HTML <pre> element in the following way:

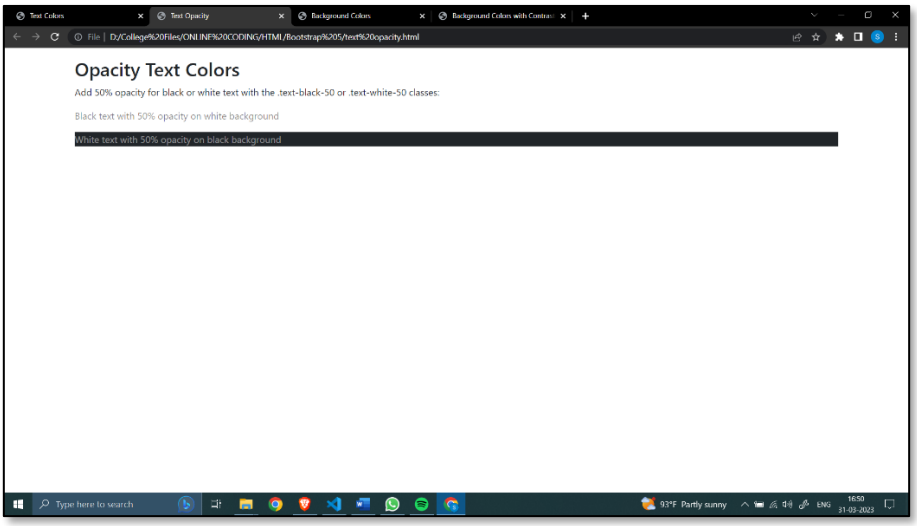


Colors

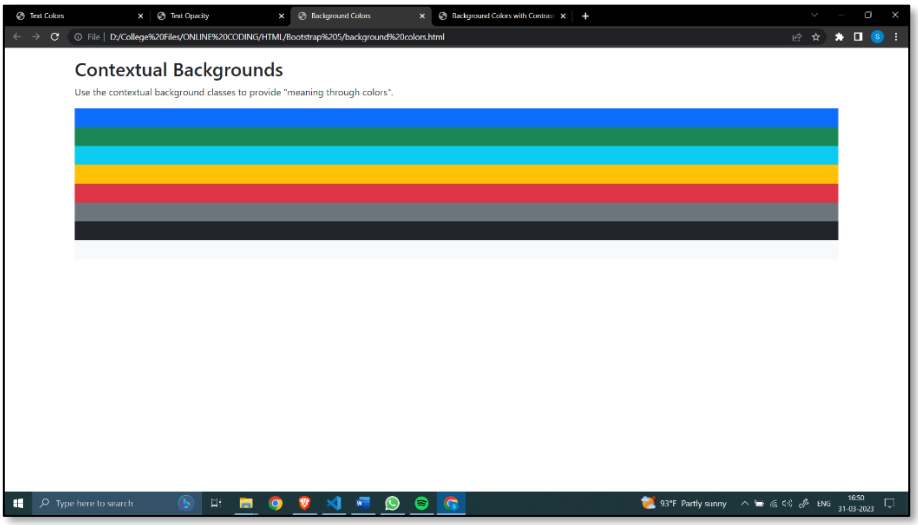
Bootstrap 5 has some contextual classes that can be used to provide meaning through colors. The classes for text colors are: .text-muted, .text-primary, .text-success, .text-info, .text-warning, .text-danger, .text-secondary, .text-white, .text-dark, .text-body (default body color/often black) and .text-light:



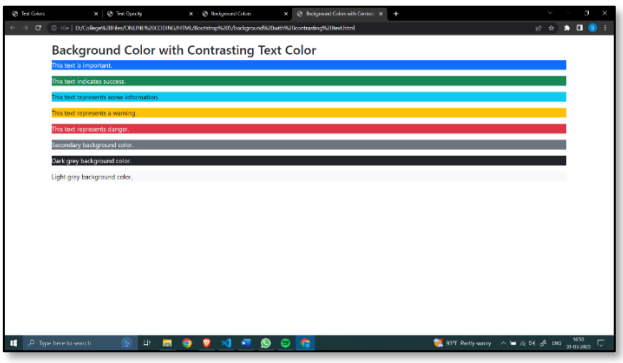
You can also add 50% opacity for black or white text with the .text-black-50 or .text-white-50 classes:



The classes for background colors are: .bg-primary, .bg-success, .bg-info, .bg-warning, .bg-danger, .bg-secondary, .bg-dark and .bg-light. The **.bg-color** classes above does not work well with text, or atleast then you have to specify a proper **.text-color** class to get the right text color for each background:

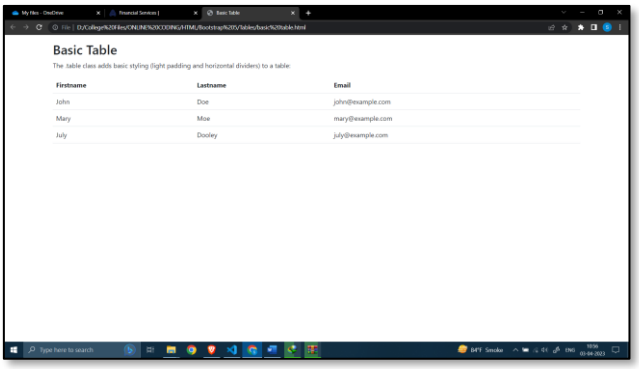


However, you can use the **.text-bg-color** classes and Bootstrap will automatically handle the appropriate text color for each background color:



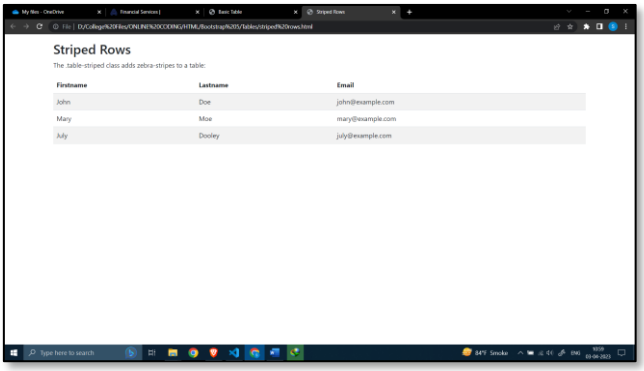
Tables

A basic Bootstrap 5 table has a light padding and horizontal dividers. The **.table** class adds basic styling to a table:



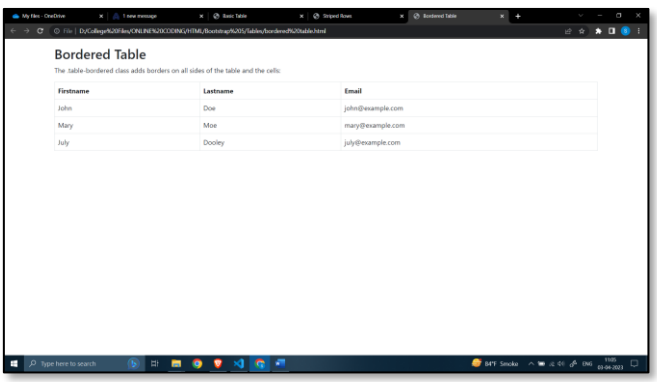
Striped Rows

The **.table-striped** class adds zebra-stripes to a table:



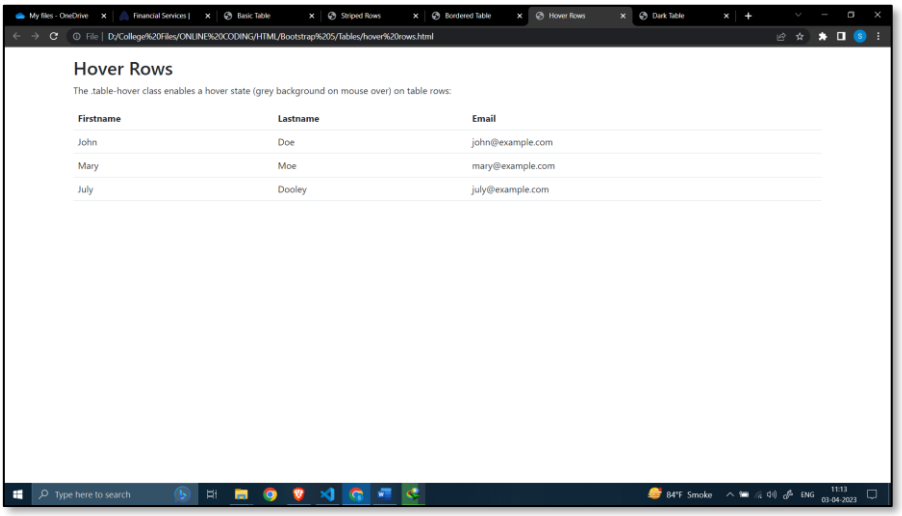
Bordered Table

The **.table-bordered** class adds borders on all sides of the table and cells:



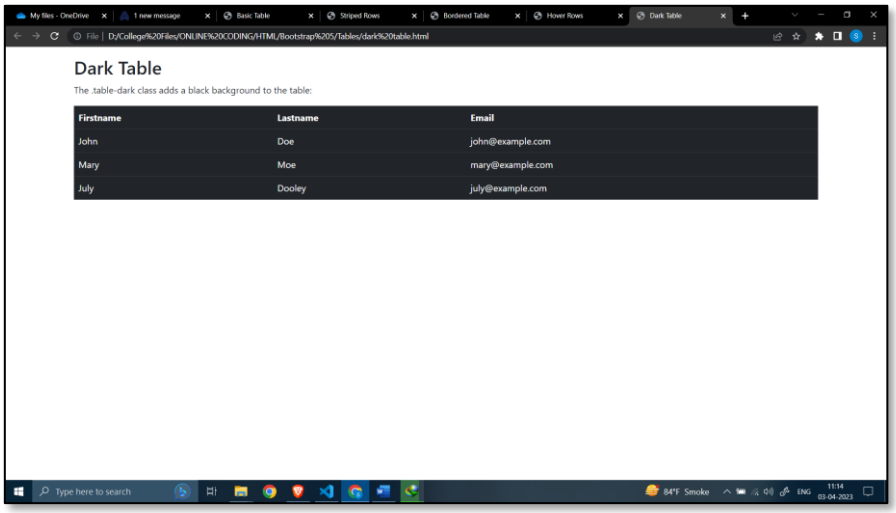
Hover Rows

The **.table-hover** class adds a hover effect (grey background color) on table rows:



Black/Dark Table

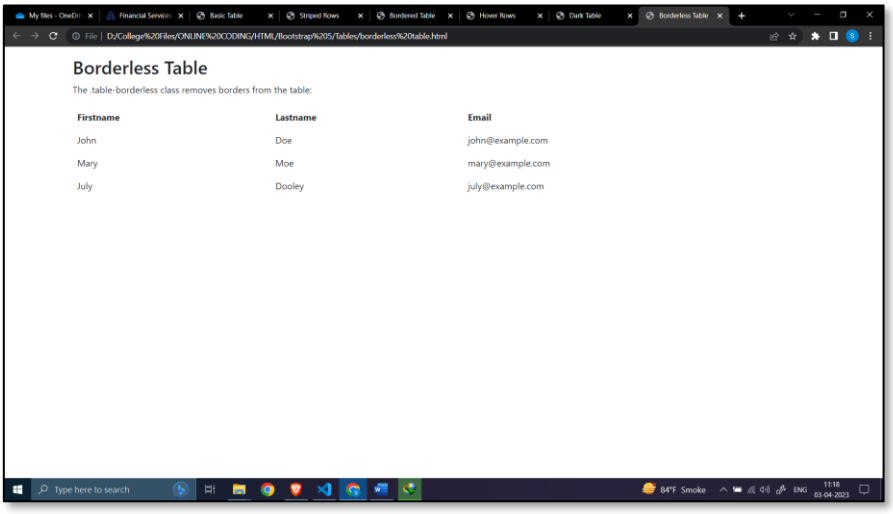
The **.table-dark** class adds a black background to the table:



We can also combine the **.table-dark** with other classes like **.table-striped** or **.table-hover** to create a dark striped or hoverable dark table respectively.

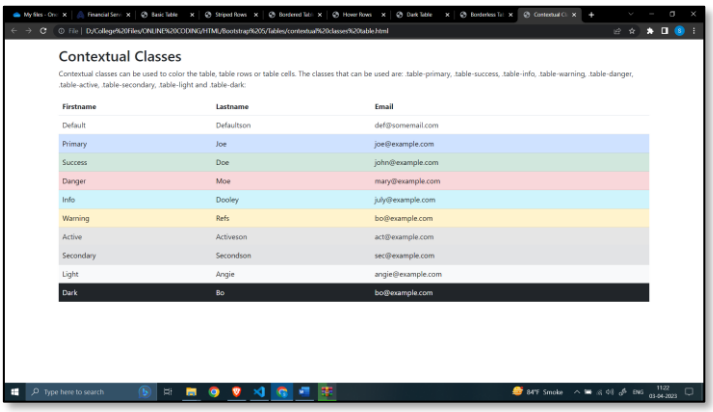
Borderless Table

The **.table-borderless** class removes borders from the table:



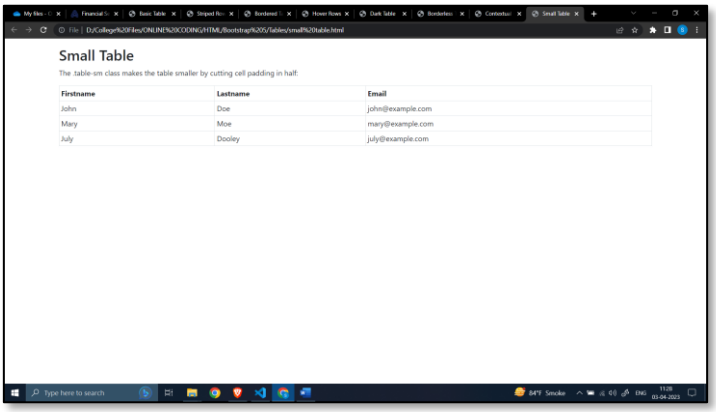
Contextual Classes

Contextual classes can be used to color the whole table (<table>), the table rows (<tr>) or table cells (<td>).



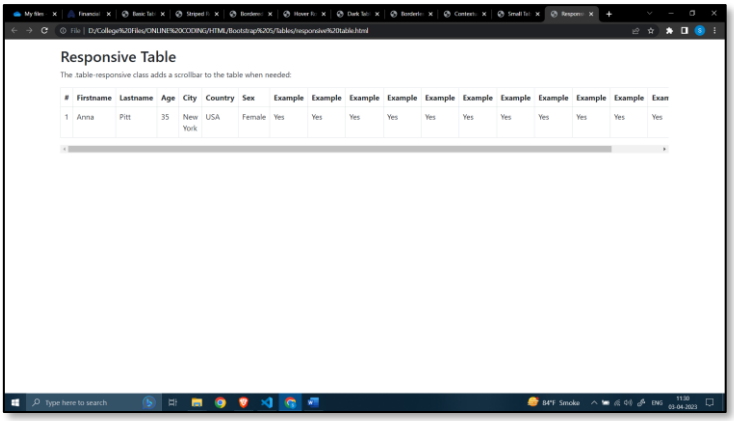
Small Table

The **.table-sm** class makes the table smaller by cutting cell padding in half:



Responsive Table

The **.table-responsive** class adds a scrollbar to the table when needed (when it is too big horizontally):



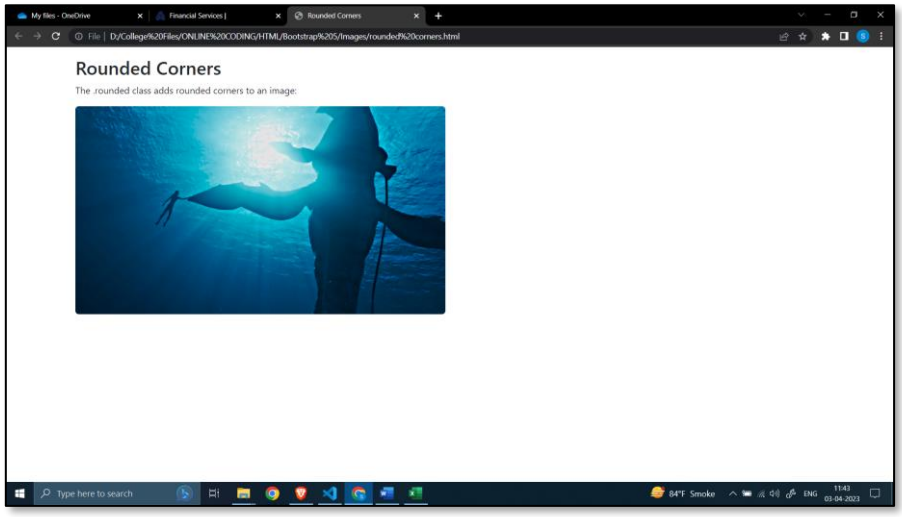
You can also decide when the table should get a scrollbar, depending on the screen width:

Class	Screen Width
.table-responsive-sm	< 576px
.table-responsive-md	< 768px
.table-responsive-lg	< 992px
.table-responsive-xl	< 1200px
.table-responsive-xxl	< 1400px

Images

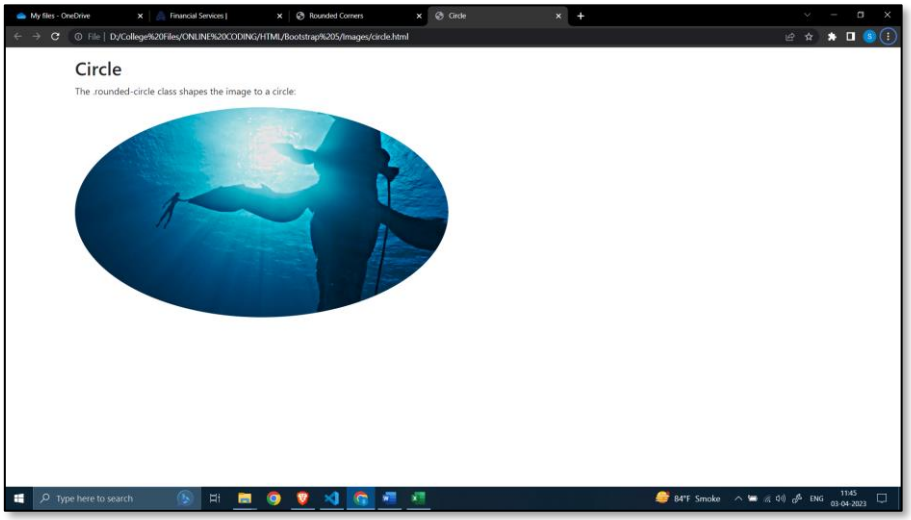
Rounded Corners

The **.rounded** class adds rounded corners to an image:



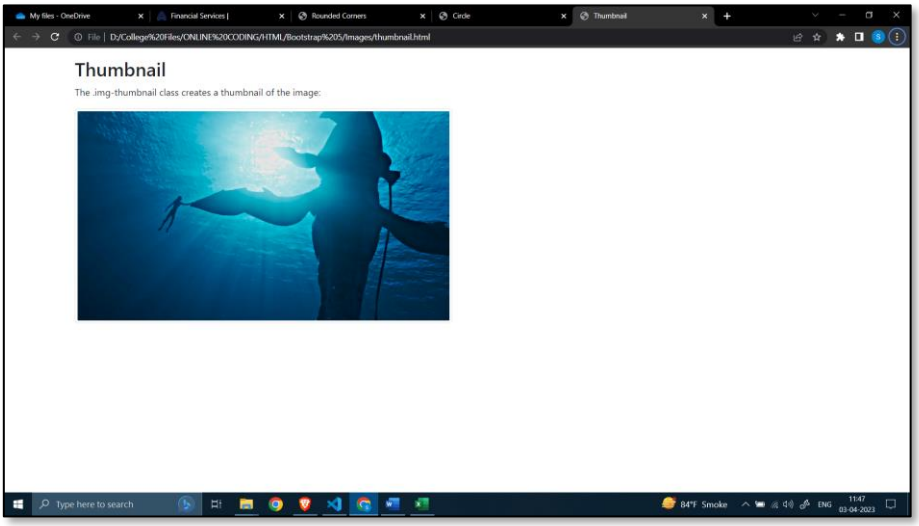
Circle

The **.rounded-circle** class shapes the image to a circle:



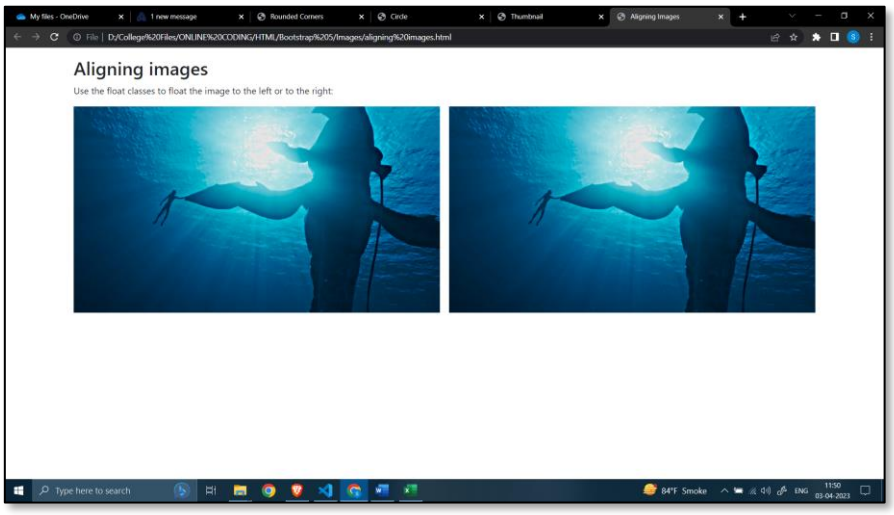
Thumbnail

The **.img-thumbnail** class shapes the image to a thumbnail (bordered):



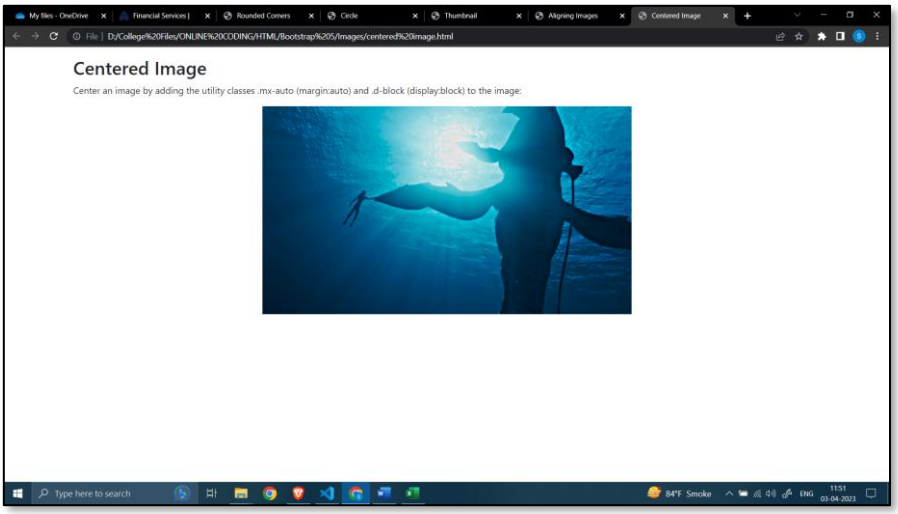
Aligning Images

Float an image to the left with the **.float-start** class or to the right with **.float-end**:



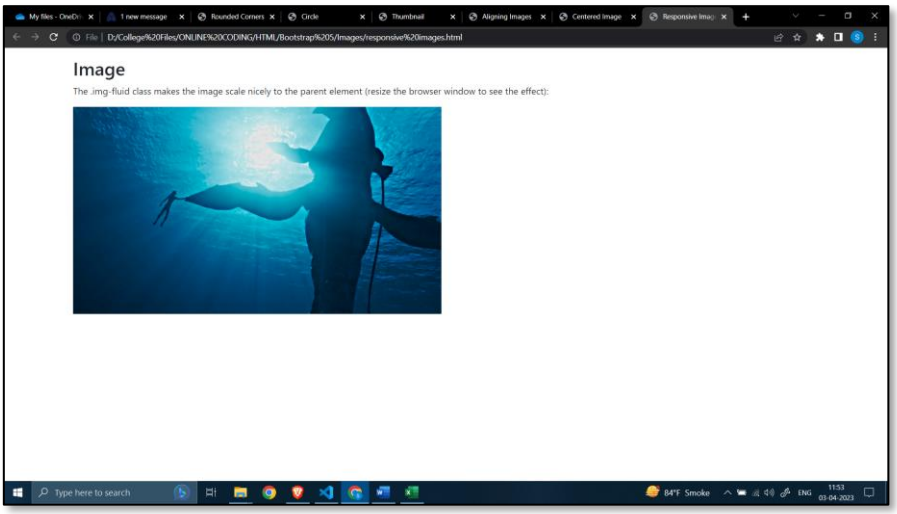
Centered Image

Center an image by adding the utility classes **.mx-auto** (margin:auto) and **.d-block** (display:block) to the image:



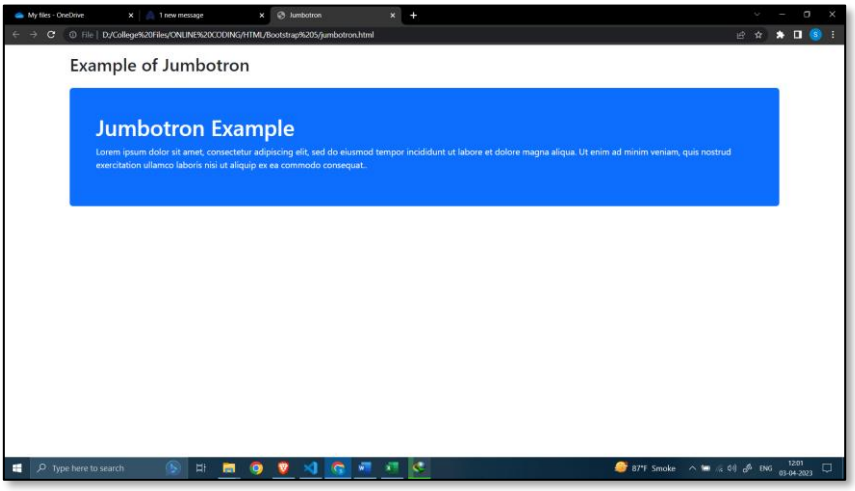
Responsive Images

Images come in all sizes. So do screens. Responsive images automatically adjust to fit the size of the screen. Create responsive images by adding an **.img-fluid** class to the <img> tag. The image will then scale nicely to the parent element and it applies max-width: 100%; and height: auto; to the image:



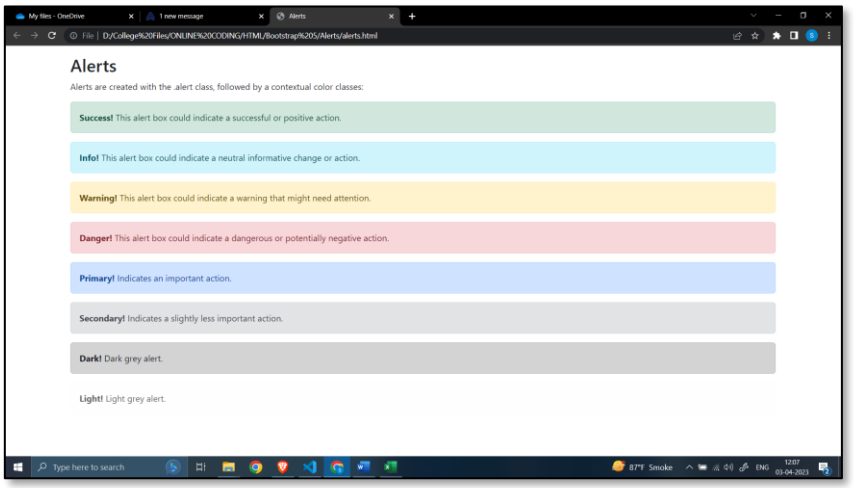
Jumbotron

A jumbotron was introduced in Bootstrap 3 as a big padded box for calling extra attention to some special content or information. Jumbotrons are no longer supported in Bootstrap 5. However, you can use a element and add special helper classes together with a color class to achieve the same effect:



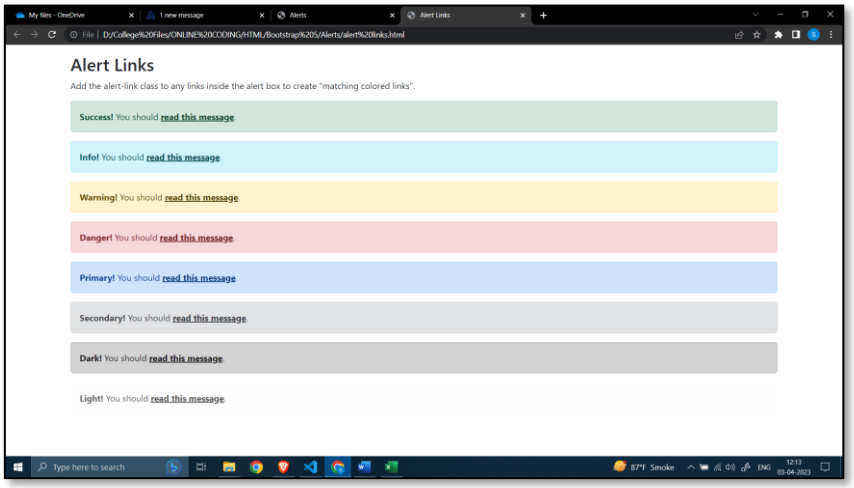
Alerts

Bootstrap 5 provides an easy way to create predefined alert messages. Alerts are created with the **.alert** class, followed by one of the contextual classes:



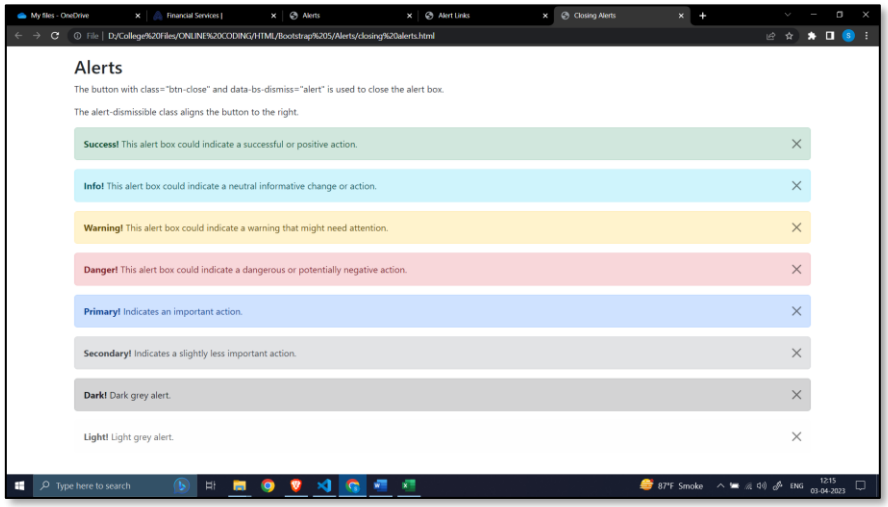
Alert Links

Add the **.alert-link** class to any links inside the alert box to create matching colored links:



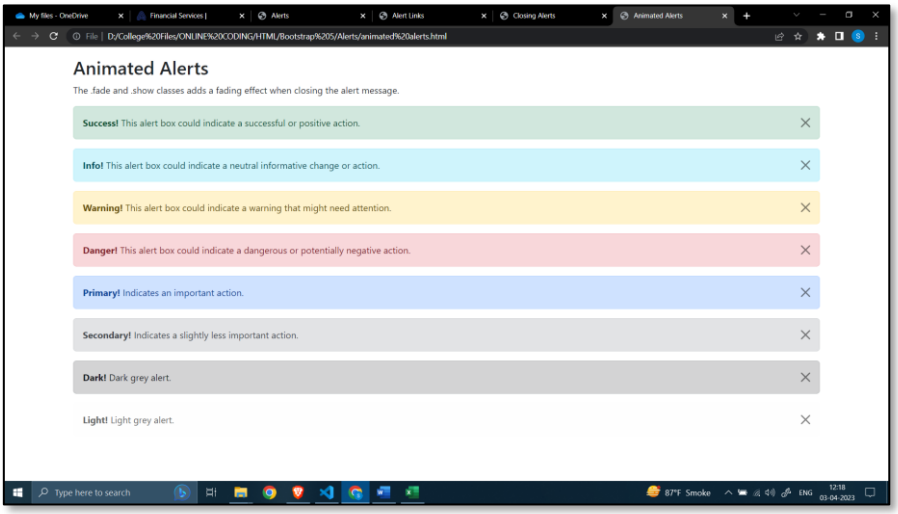
Closing Alerts

To close the alert message, add a **.alert-dismissible** class to the alert container. Then add **class="btn-close"** and **data-bs-dismiss="alert"** to a link or a button element (when you click on this the alert box will disappear).



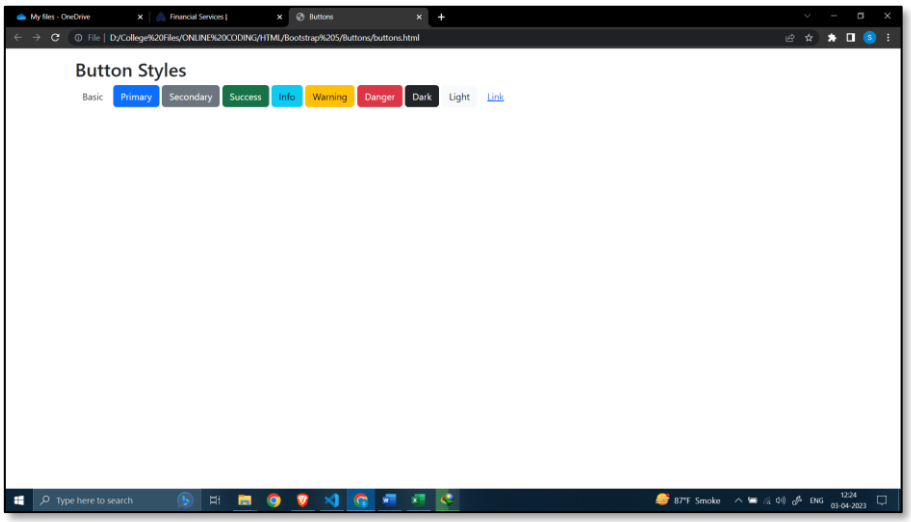
Animated Alerts

The **.fade** and **.show** classes adds a fading effect when closing the alert message:

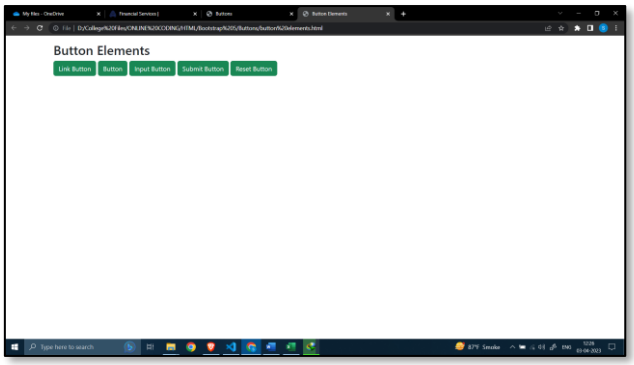


Buttons

Bootstrap 5 provides different styles of buttons:

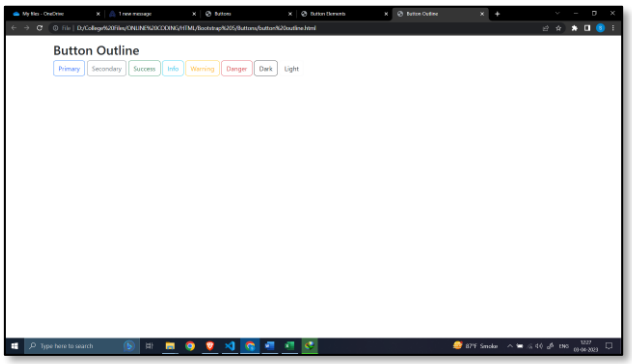


The button classes can be used on <a>, <button>, or <input> elements:



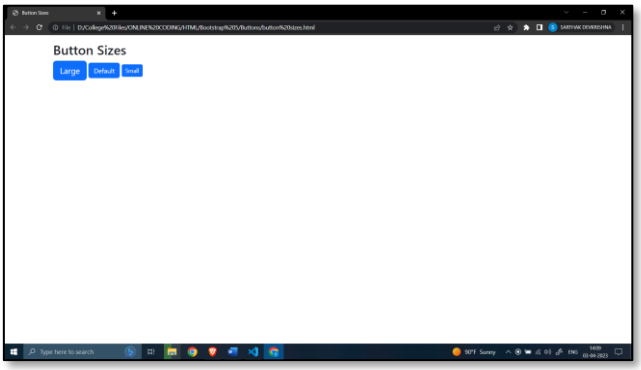
Button Outline

Bootstrap 5 also provides eight outline/bordered buttons. Move the mouse over them to see an additional "hover" effect:



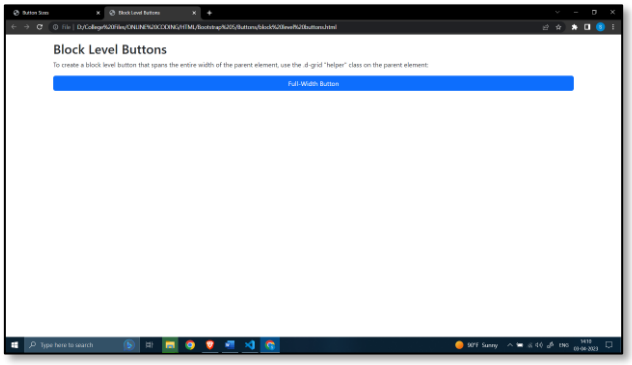
Button Sizes

Use the **.btn-lg** class for large buttons or **.btn-sm** class for small buttons:

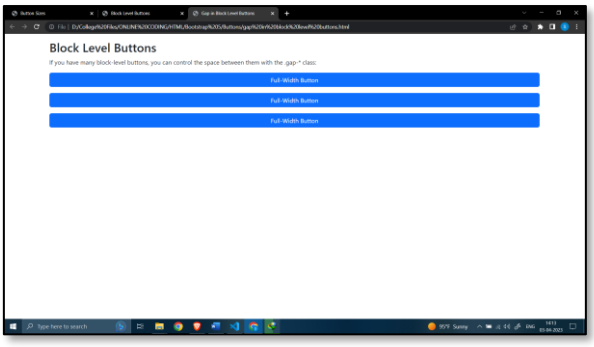


Block Level Buttons

To create a block level button that spans the entire width of the parent element, use the **.d-grid** helper class on the parent element:

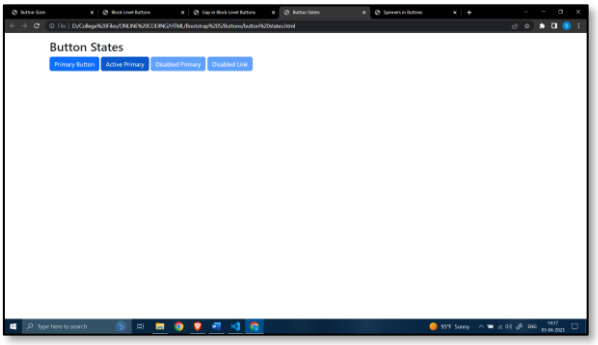


If you have many block-level buttons, you can control the space between them with the **.gap-*** class:



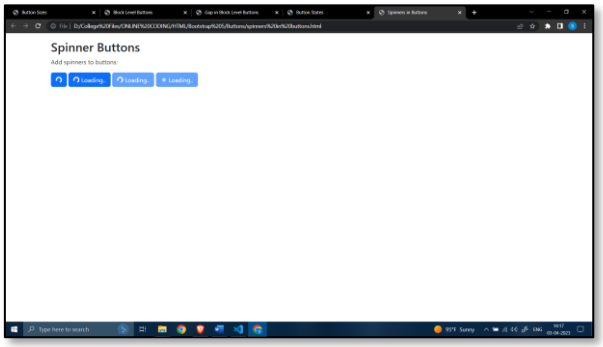
Active/Disabled Buttons

A button can be set to an active (appear pressed) or a disabled (unclickable) state. The class **.active** makes a button appear pressed, and the disabled attribute makes a button unclickable. Note that <a> elements do not support the disabled attribute and must therefore use the **.disabled** class to make it visually appear disabled.



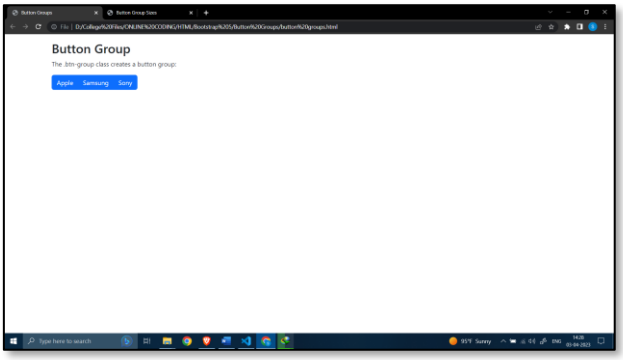
Spinner Buttons

You can also add spinners to a button as below:

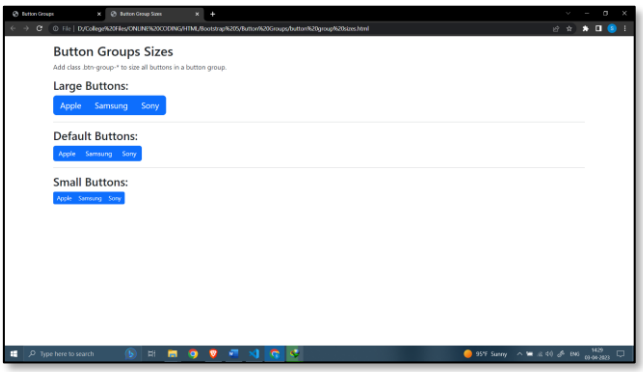


Button Groups

Bootstrap 5 allows you to group a series of buttons together (on a single line) in a button group. Use a element with class **.btn-group** to create a button group:

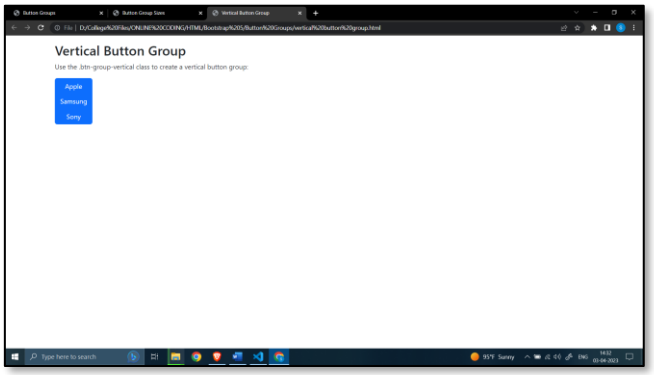


Instead of applying button sizes to every button in a group, use class **.btn-group-lg** for a large button group or the **.btn-group-sm** for a small button group:



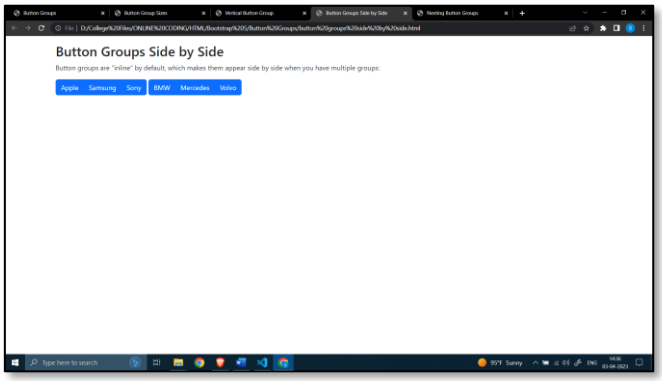
Vertical Button Groups

Bootstrap 5 also supports vertical button groups. Use the class **.btn-group-vertical** to create a vertical button group:



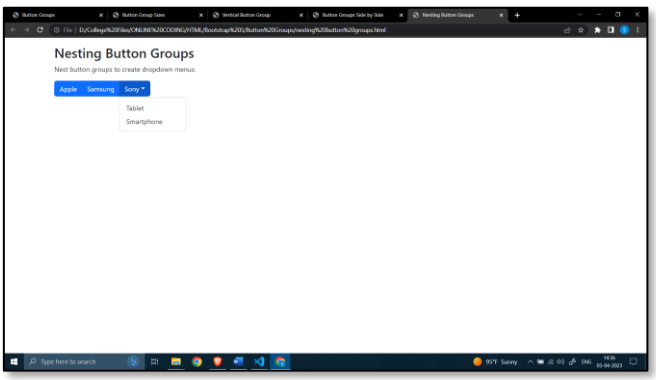
Button Groups Side by Side

Button groups are "inline" by default, which makes them appear side by side when you have multiple groups:



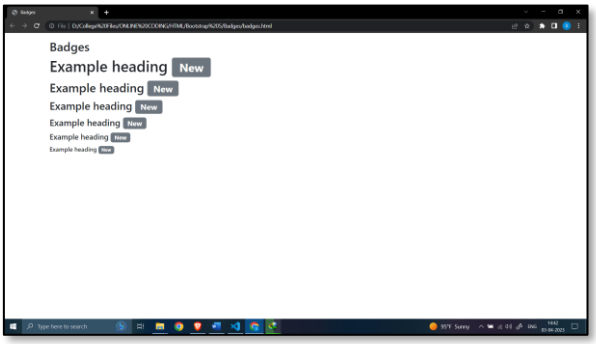
Nesting Button Groups & Dropdown Menus

Nest button groups to create dropdown menus:



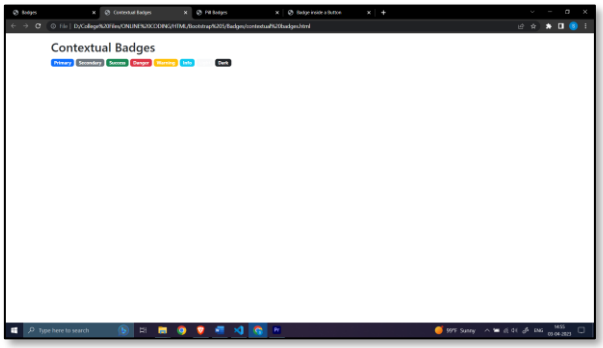
Badges

Badges are used to add additional information to any content. Use the **.badge** class together with a contextual class (like **.bg-secondary**) within elements to create rectangular badges. Note that badges scale to match the size of the parent element (if any):



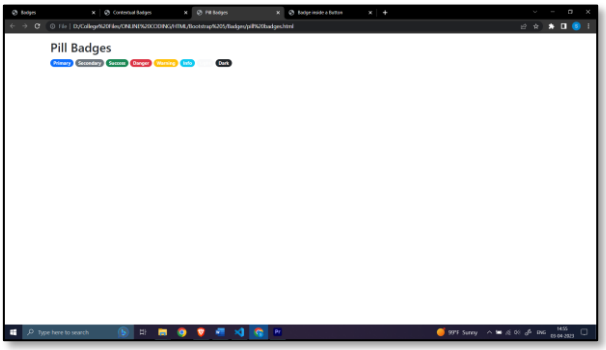
Contextual Badges

Use any of the contextual classes (**.bg-***) to change the color of a badge:



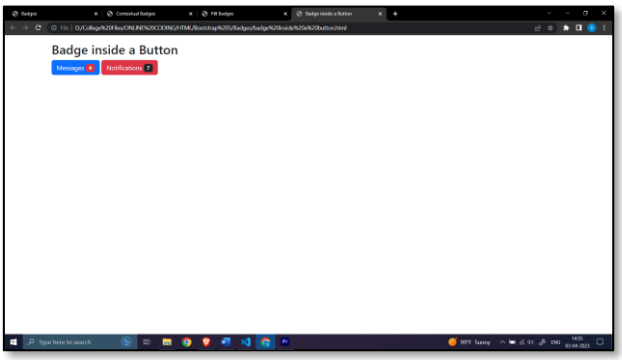
Pill Badges

Use the **.rounded-pill** class to make the badges more round:



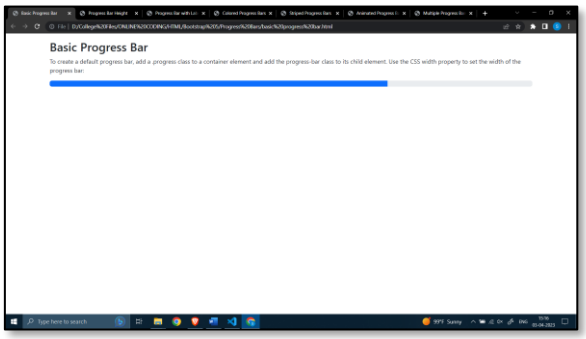
Badge inside an Element

An example of using a badge inside a button:



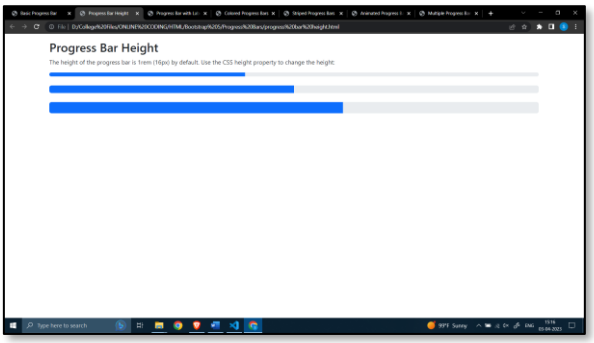
Progress Bars

A progress bar can be used to show how far a user is in a process. To create a default progress bar, add a **.progress** class to a container element and add the .progress-bar class to its child element. Use the CSS width property to set the width of the progress bar:



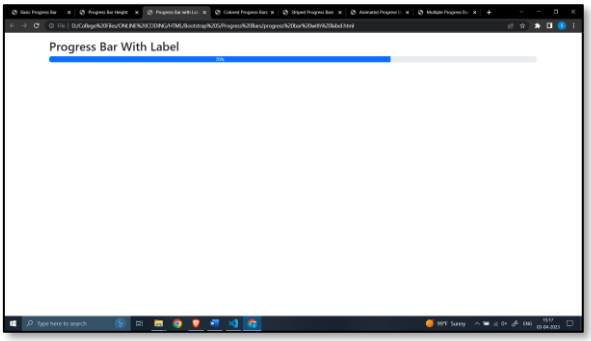
Progress Bar Height

The height of the progress bar is 1rem (usually 16px) by default. Use the CSS height property to change it. Note that you must set the same height for the progress container and the progress bar:



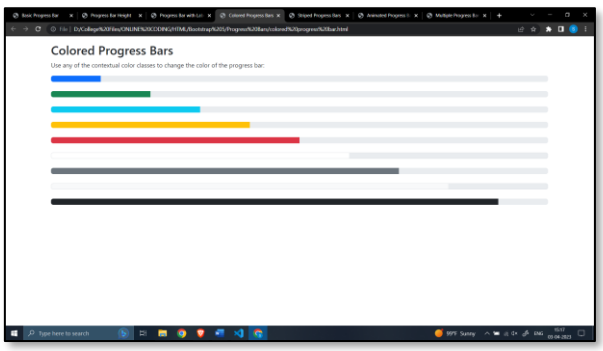
Progress Bar Labels

Add text inside the progress bar to show the visible percentage:



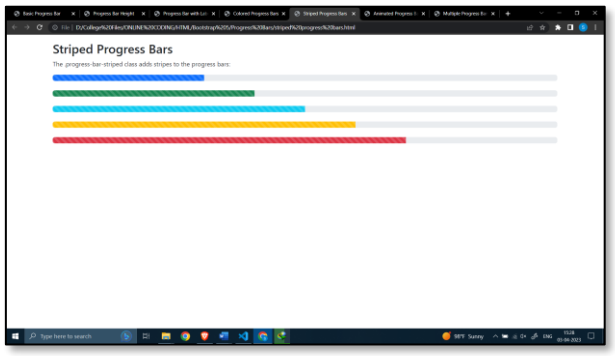
Colored Progress Bars

By default, the progress bar is blue (primary). Use any of the contextual background classes to change its color:



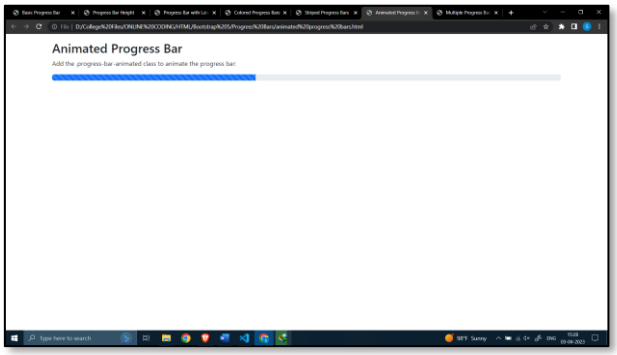
Striped Progress Bars

Use the **.progress-bar-striped** class to add stripes to the progress bars:



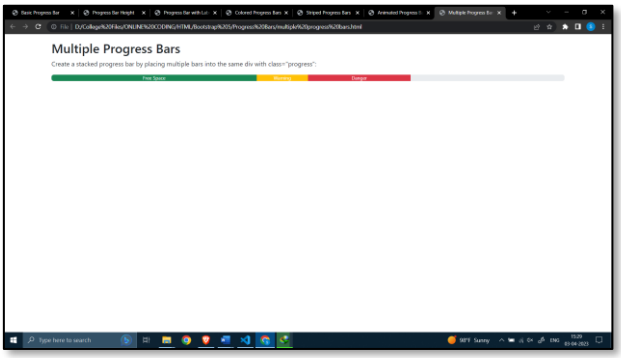
Animated Progress Bar

Add the **.progress-bar-animated** class to animate the progress bar:



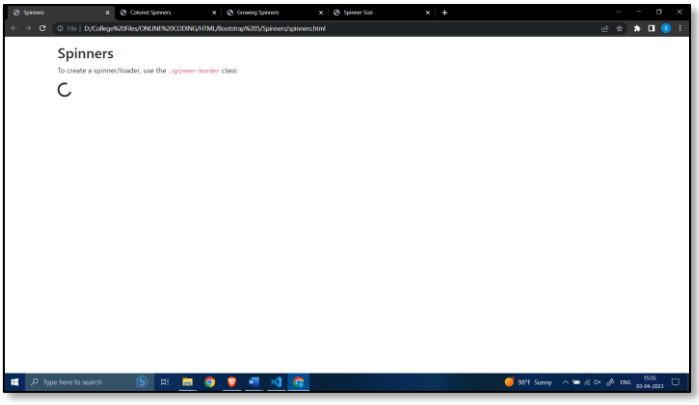
Multiple Progress Bars

Progress bars can also be stacked as below:



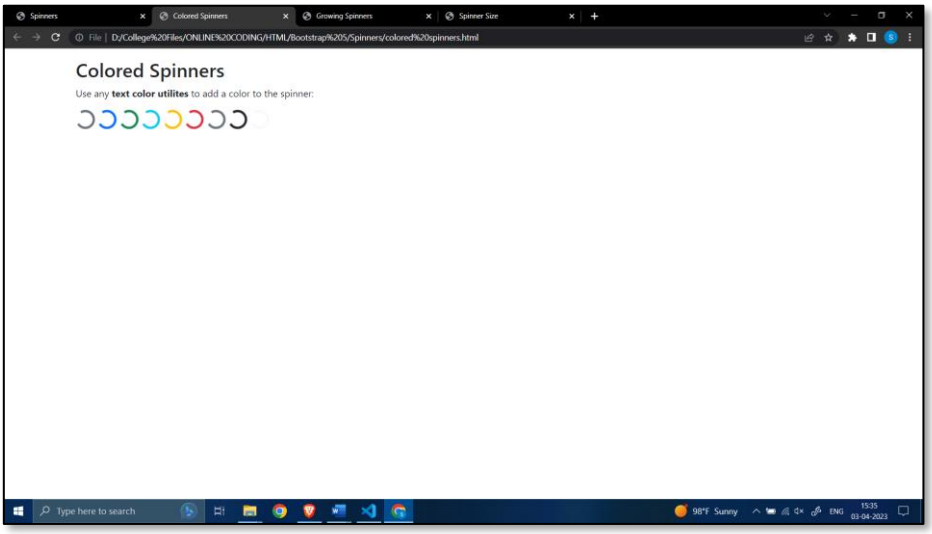
Spinners

To create a spinner/loader, use the **.spinner-border** class:



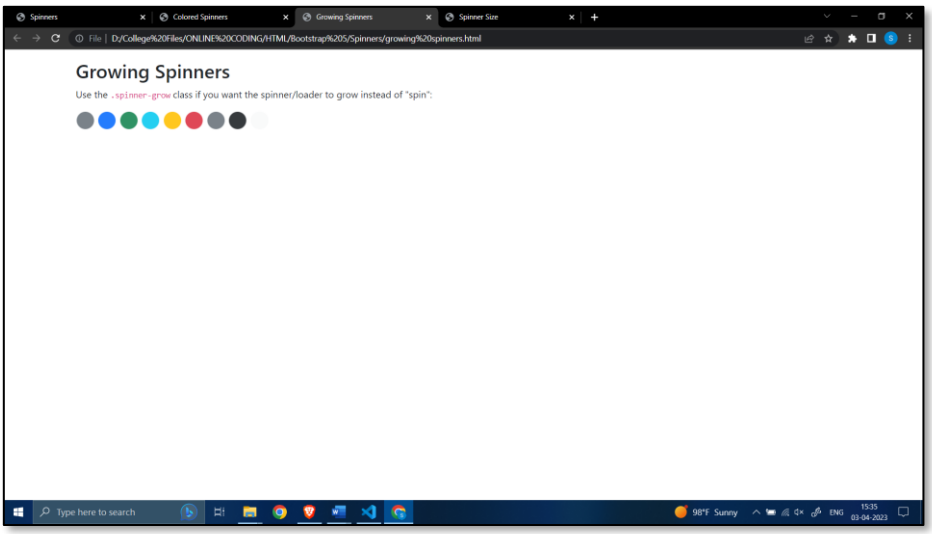
Colored Spinners

Use any text color utilites to add a color to the spinner:



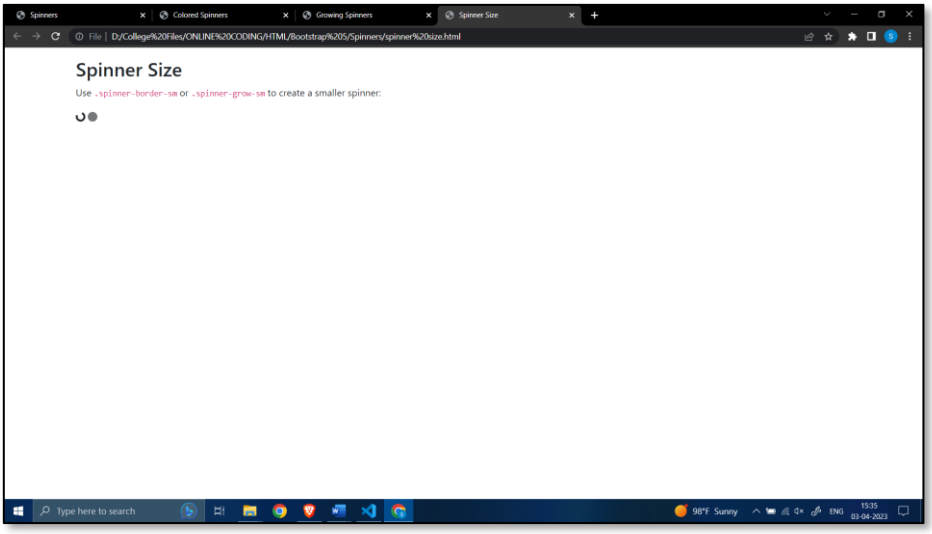
Growing Spinners

Use the **.spinner-grow** class if you want the spinner/loader to grow instead of "spin":



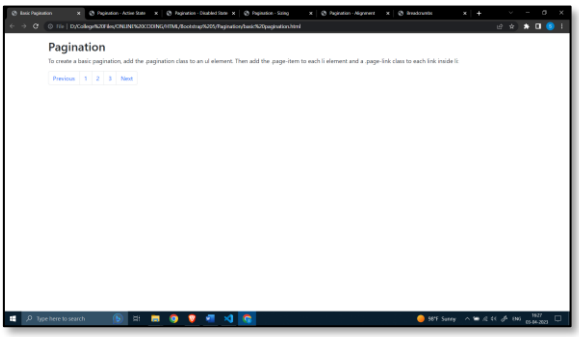
Spinner Size

Use **.spinner-border-sm** or **.spinner-grow-sm** to create a smaller spinner:



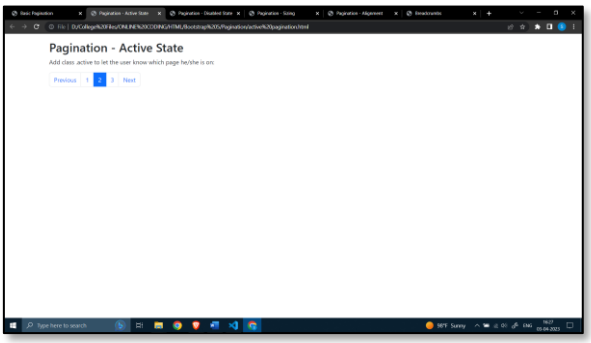
Pagination

If you have a web site with lots of pages, you may wish to add some sort of pagination to each page. To create a basic pagination, add the **.pagination** class to an <ul> element. Then add the **.page-item** to each <li> element and a **.page-link** class to each link inside <li>:



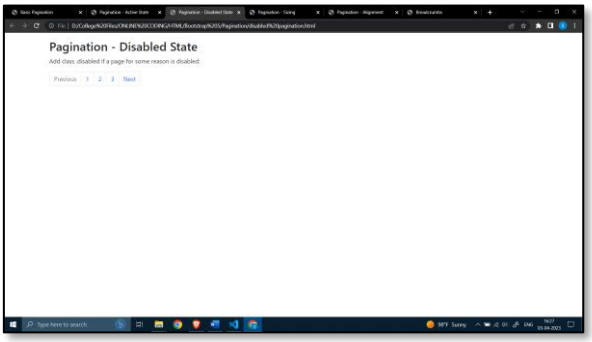
Active State

The **.active** class is used to "highlight" the current page:



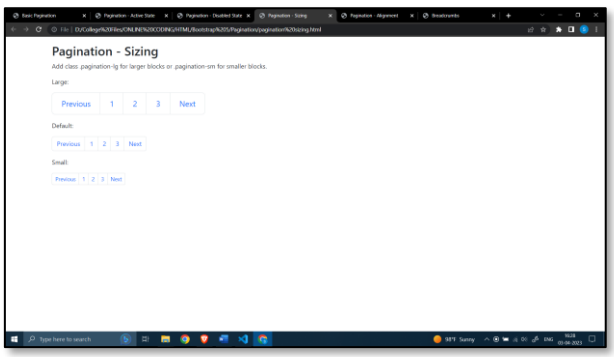
Disabled State

The **.disabled** class is used for un-clickable links:



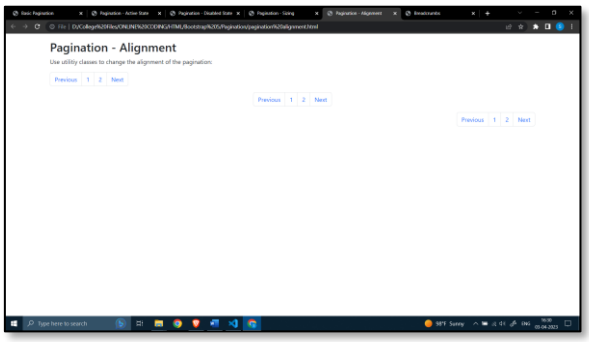
Pagination Sizing

Pagination blocks can also be sized to a larger or a smaller size. Add class **.pagination-lg** for larger blocks or **.pagination-sm** for smaller blocks:



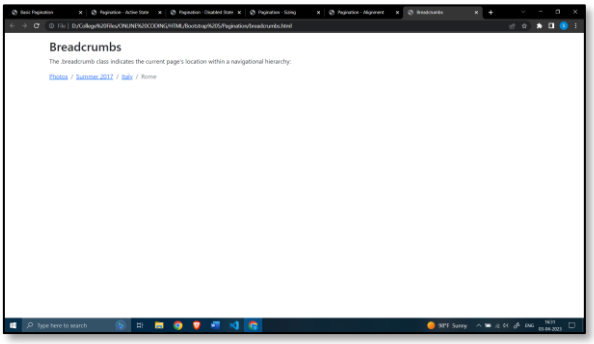
Pagination Alignment

Use utility classes to change the alignment of the pagination:



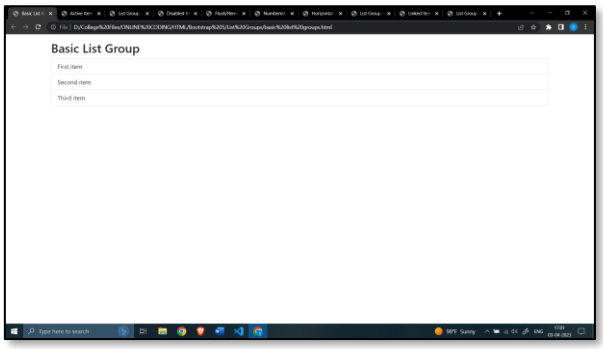
Breadcrumbs

Another form for pagination, is breadcrumbs. The **.breadcrumb** and **.breadcrumb-item** classes indicates the current page's location within a navigational hierarchy:



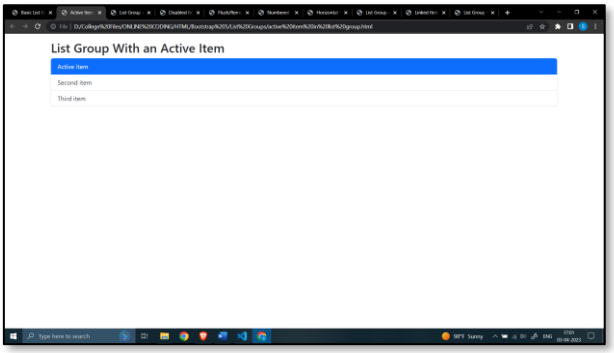
List Groups

The most basic list group is an unordered list with list items. To create a basic list group, use an <ul> element with class **.list-group**, and <li> elements with class **.list-group-item**:



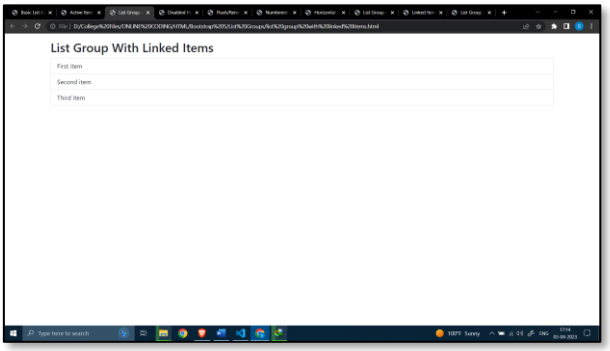
Active State

Use the **.active** class to highlight the current item:



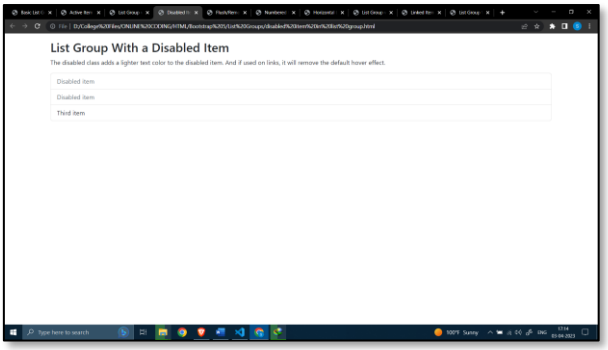
List Group With Linked Items

To create a list group with linked items, use instead of <ul> and <a> instead of <li>. Optionally, add the **.list-group-item-action** class if you want a grey background color on hover:



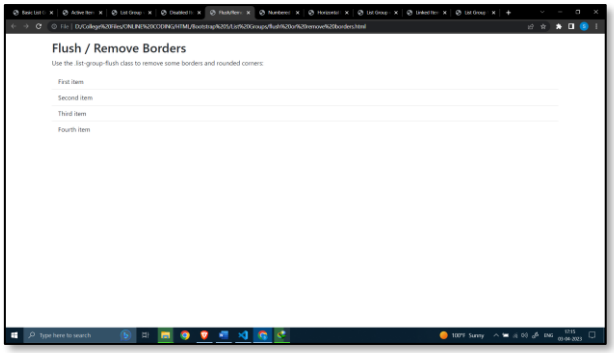
Disabled Item

The **.disabled** class adds a lighter text color to the disabled item. And when used on links, it will remove the hover effect:



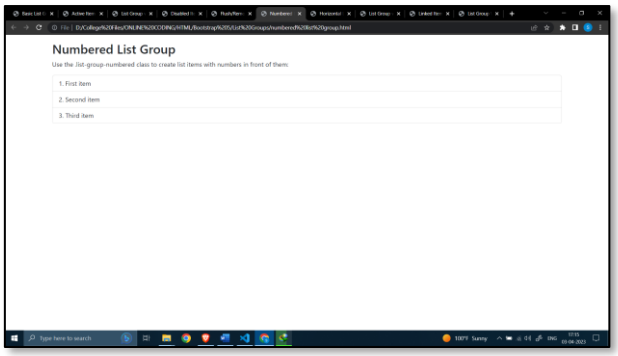
Flush / Remove Borders

Use the **.list-group-flush** class to remove some borders and rounded corners:



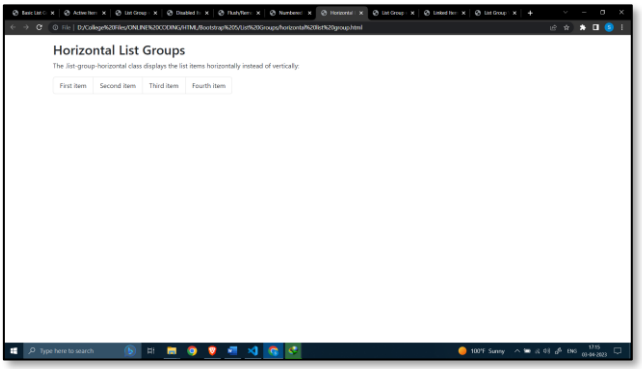
Numbered List Groups

Use the **.list-group-numbered** class to create list items with numbers in front of them:



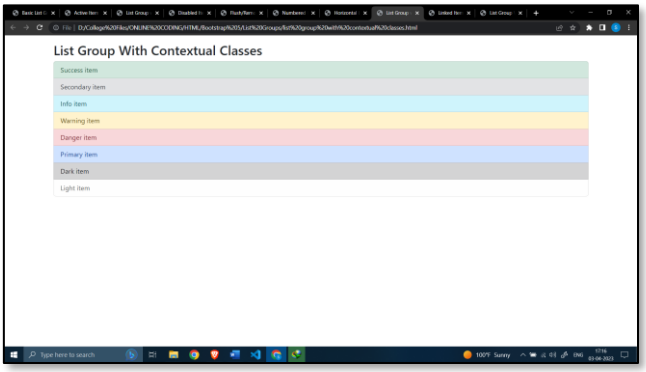
Horizontal List Groups

If you want the list items to display horizontally instead of vertically (side by side instead of on top of each other), add the **.list-group-horizontal** class to **.list-group**:

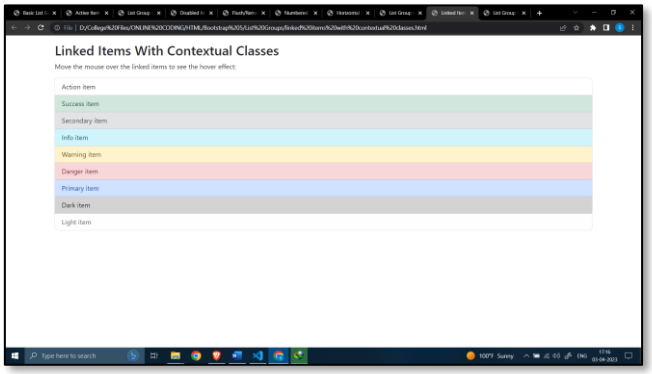


Contextual Classes

Contcontextual classes can be used to add color to the list items:

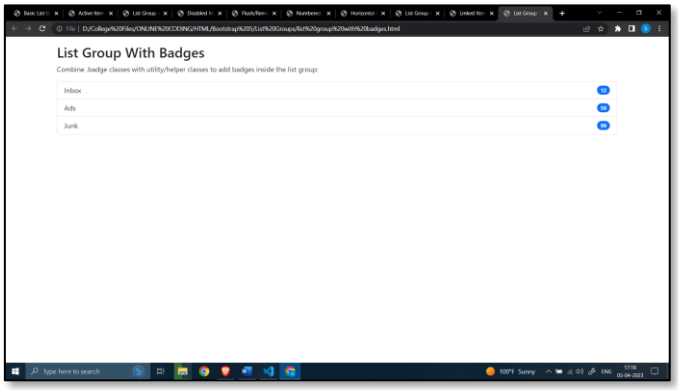


Link items with Contextual Classes



List Group with Badges

Combine **.badge** classes with utility/helper classes to add badges inside the list group:

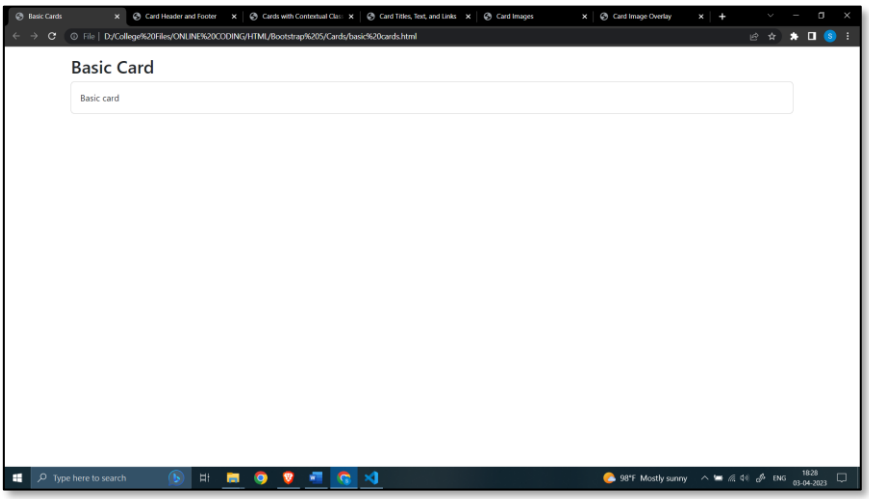


Cards

A card in Bootstrap 5 is a bordered box with some padding around its content. It includes options for headers, footers, content, colors, etc.

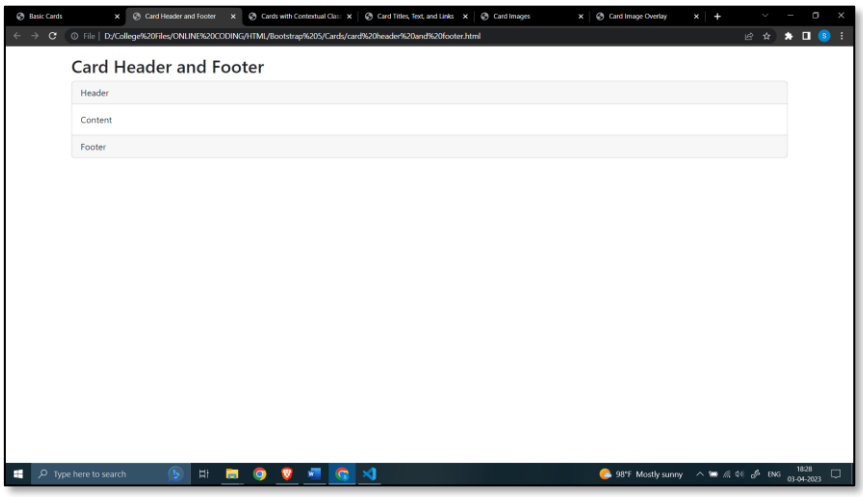
Basic Card

A basic card is created with the **.card** class, and content inside the card has a **.card-body** class:



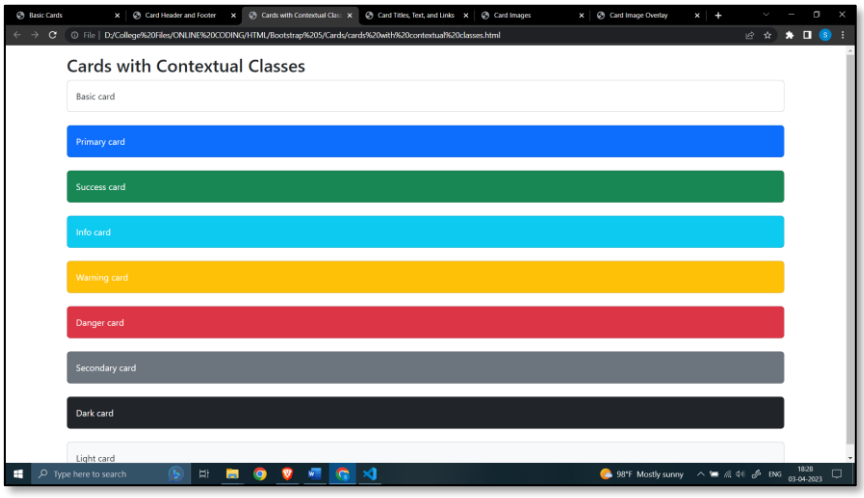
Header and Footer

The **.card-header** class adds a heading to the card and the .card-footer class adds a footer to the card:



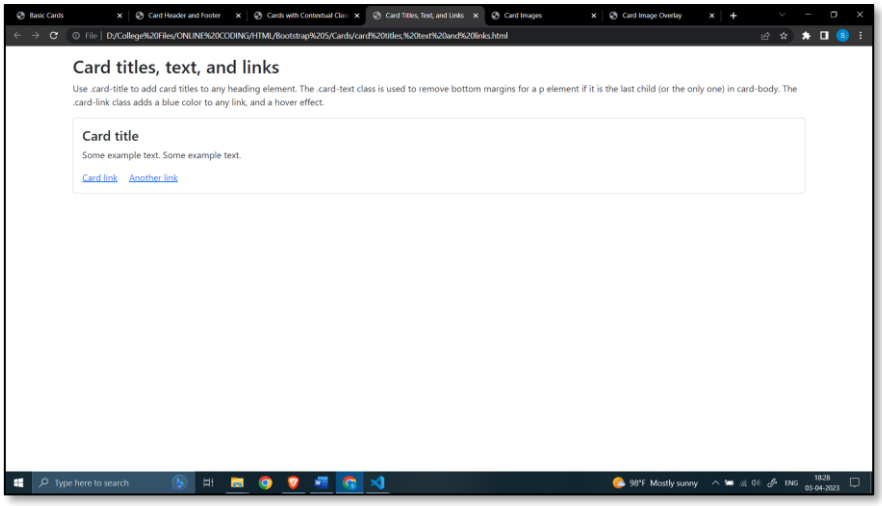
Contextual Cards

To add a background color the card, use contextual classes:



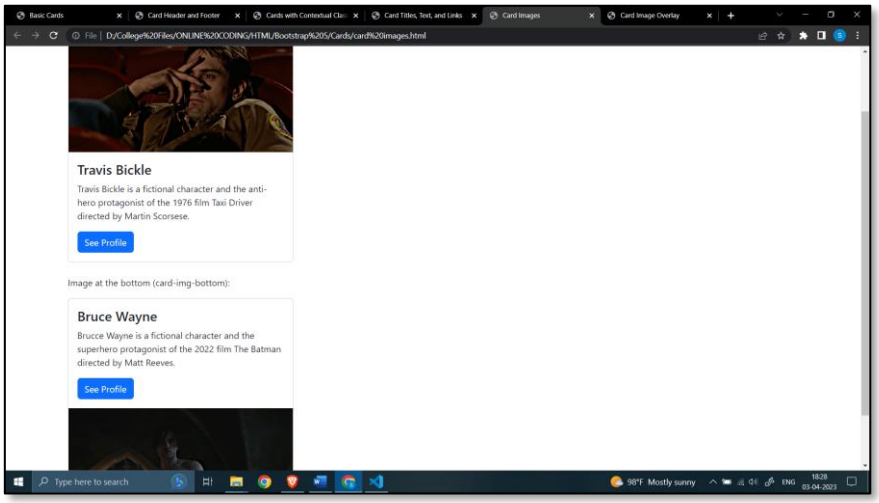
Titles, text, and links

Use **.card-title** to add card titles to any heading element. The **.card-text** class is used to remove bottom margins for a <p> element if it is the last child (or the only one) inside .card-body. The **.card-link** class adds a blue color to any link, and a hover effect.



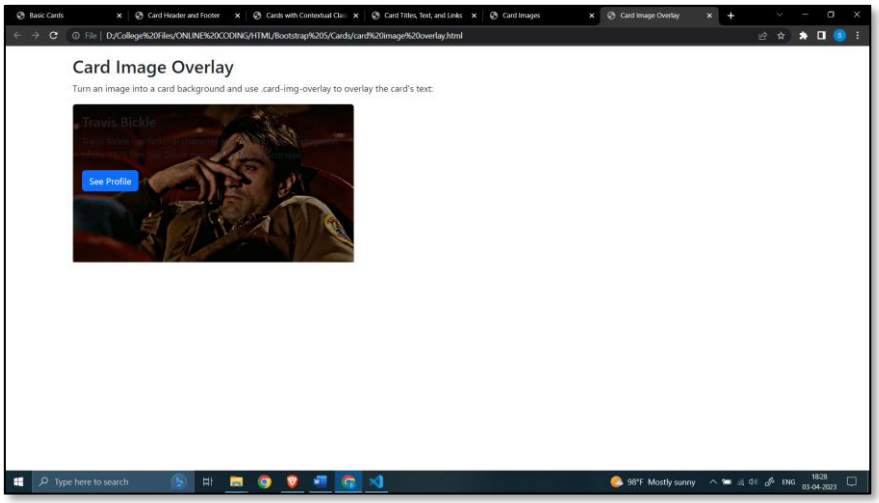
Card Images

Add **.card-img-top** or **.card-img-bottom** to an <img> to place the image at the top or at the bottom inside the card. Note that we have added the image outside of the **.card-body** to span the entire width:



Card Image Overlays

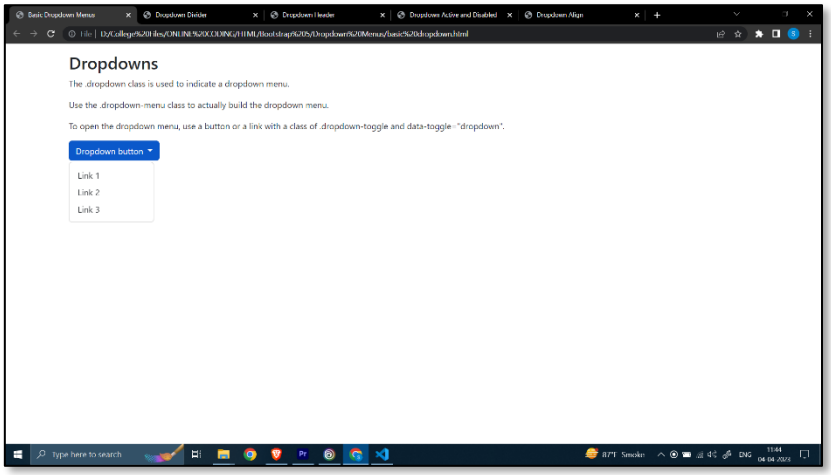
Turn an image into a card background and use **.card-img-overlay** to add text on top of the image:



Dropdown Menus

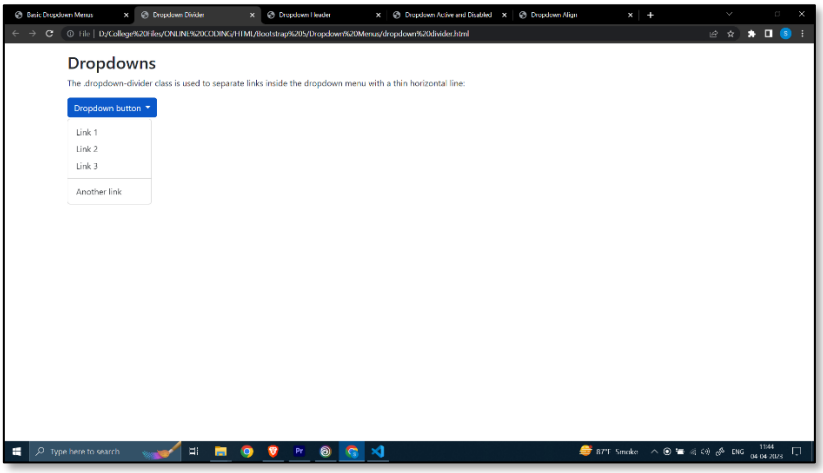
A dropdown menu is a toggleable menu that allows the user to choose one value from a predefined list.

The **.dropdown** class indicates a dropdown menu. To open the dropdown menu, use a button or a link with a class of **.dropdown-toggle** and the **data-bs-toggle="dropdown"** attribute. Add the **.dropdown-menu** class to a element to actually build the dropdown menu. Then add the **.dropdown-item** class to each element (links or buttons) inside the dropdown menu:



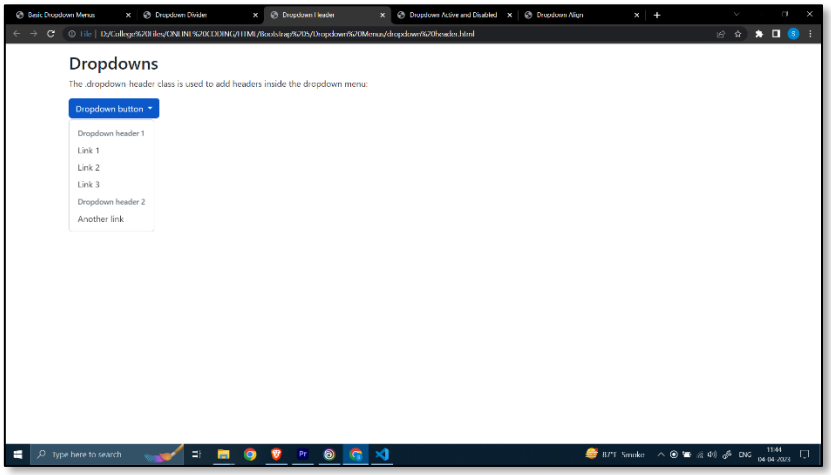
Dropdown Divider

The .dropdown-divider class is used to separate links inside the dropdown menu with a thin horizontal border:



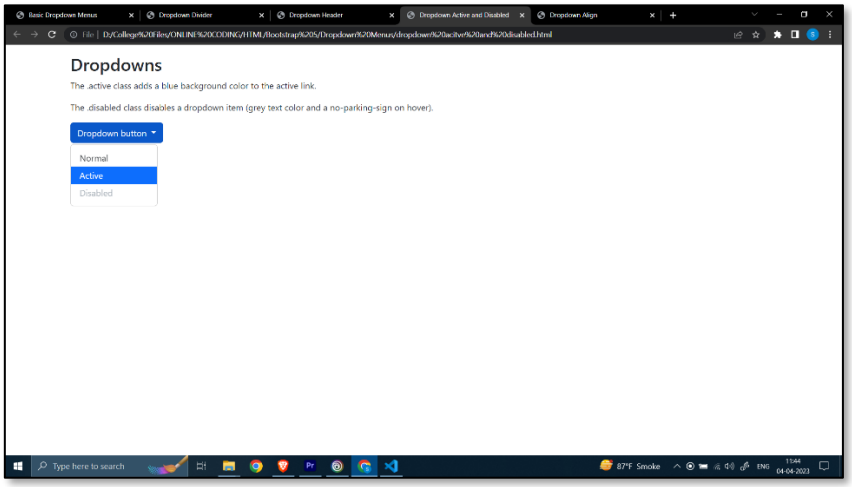
Dropdown Header

The .dropdown-header class is used to add headers inside the dropdown menu:



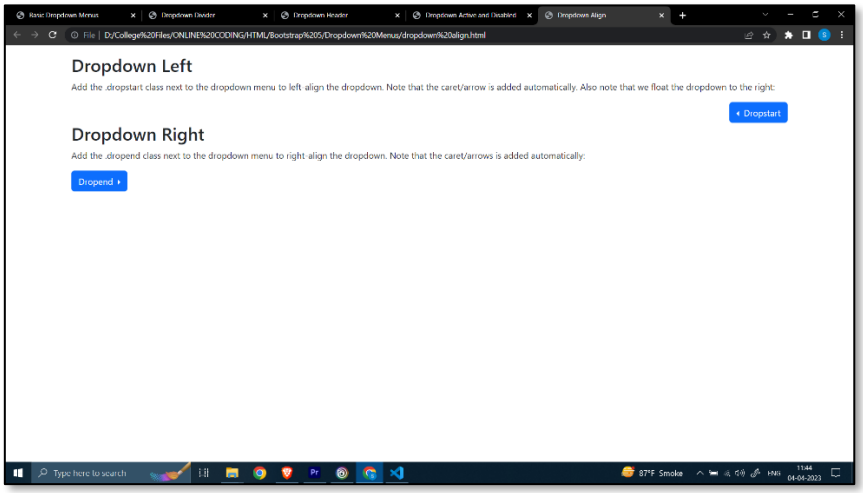
Disabled and Active items

Highlight a specific dropdown item with the **.active** class (adds a blue background color). To disable an item in the dropdown menu, use the **.disabled** class (gets a light-grey text color and a "no-parking-sign" icon on hover):



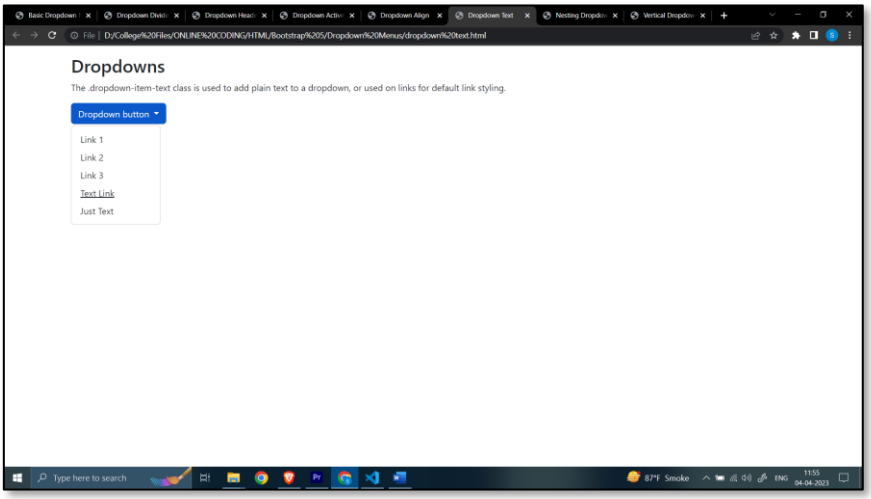
Dropdown Position

You can also create a "dropend" or "dropstart" menu, by adding the **.dropend** or **.dropstart** class to the dropdown element. Note that the caret/arrow is added automatically. To right-align the dropdown menu, add the **.dropdown-menu-end** class to the element with .dropdown-menu. If you want the dropdown menu to expand upwards instead of downwards, change the element with class="dropdown" to "dropup":

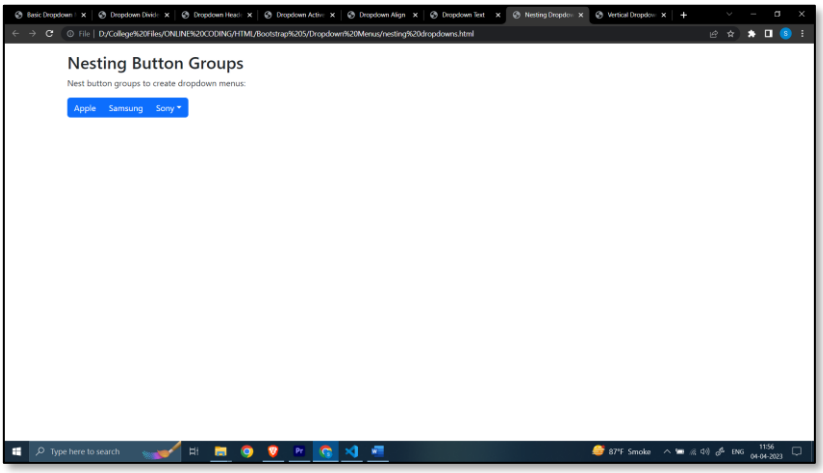


Dropdown Text

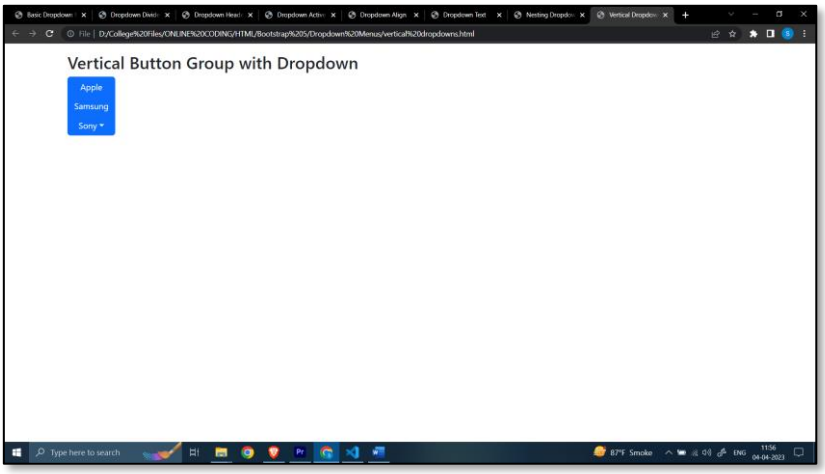
The **.dropdown-item-text** class is used to add plain text to a dropdown item, or used on links for default link styling.



Grouped Buttons with a Dropdown



Vertical Button Group w/ Dropdown

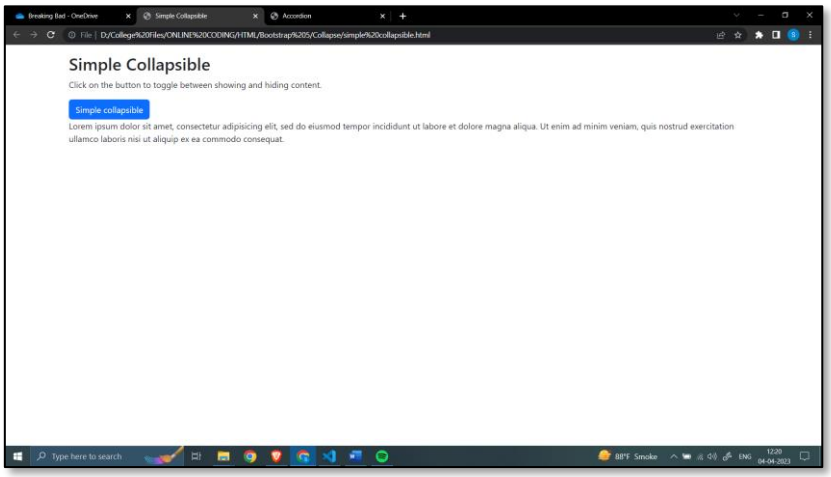


Collapsible

Basic Collapsible

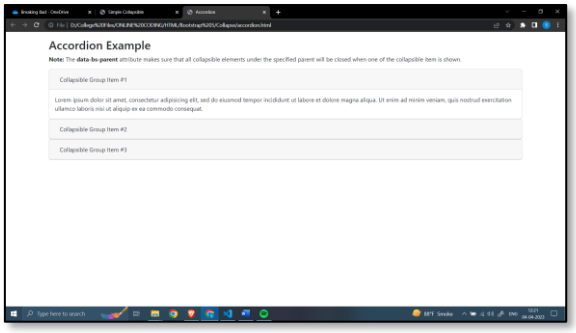
Collapsibles are useful when you want to hide and show large amount of content. The **.collapse** class indicates a collapsible element (a in our example); this is the content that will be shown or hidden with a click of a button. To control (show/hide) the collapsible content, add the data-bs-toggle="collapse" attribute to an <a> or a <button> element. Then add the data-bs-target="#id" attribute to connect the button with the collapsible content ( ).

Note: For <a> elements, you can use the href attribute instead of the data-bs-target attribute. By default, the collapsible content is hidden. However, you can add the .show class to show the content by default.



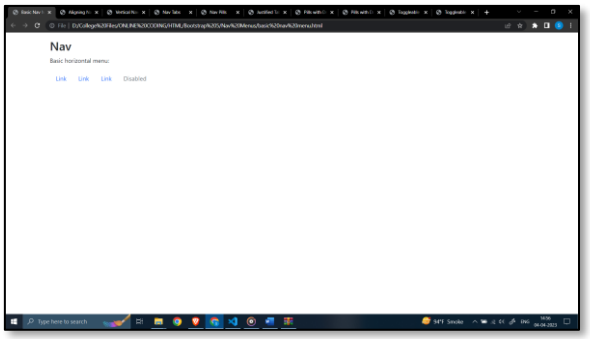
Accordion

The following example shows a simple accordion by extending the card component. Use the data-bs-parent attribute to make sure that all collapsible elements under the specified parent will be closed when one of the collapsible item is shown.



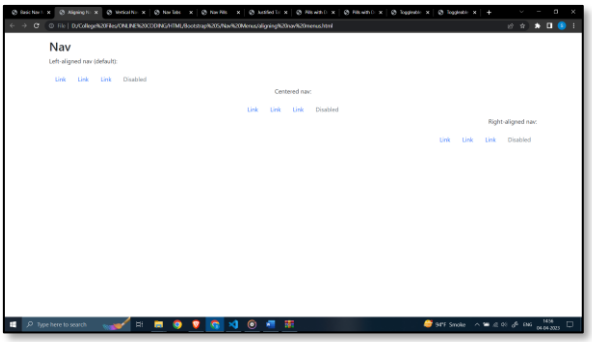
Nav Menus

If you want to create a simple horizontal menu, add the .nav class to a <ul> element, followed by .nav-item for each <li> and add the .nav-link class to their links:



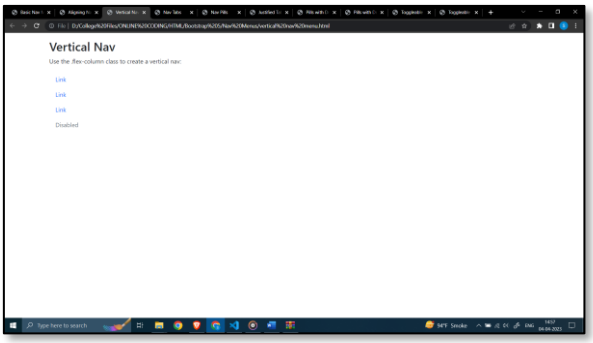
Aligned Nav

Add the .justify-content-center class to center the nav, and the .justify-content-end class to right-align the nav.



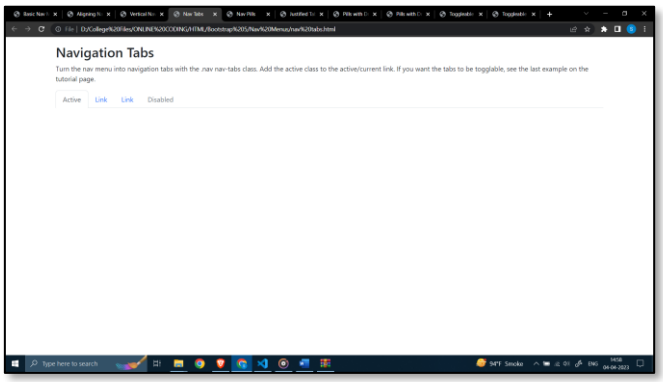
Vertical Nav

Add the .flex-column class to create a vertical nav:



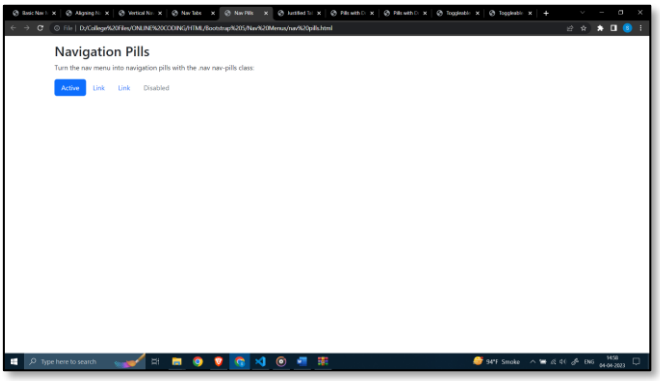
Tabs

Turn the nav menu into navigation tabs with the .nav-tabs class. Add the .active class to the active/current link.



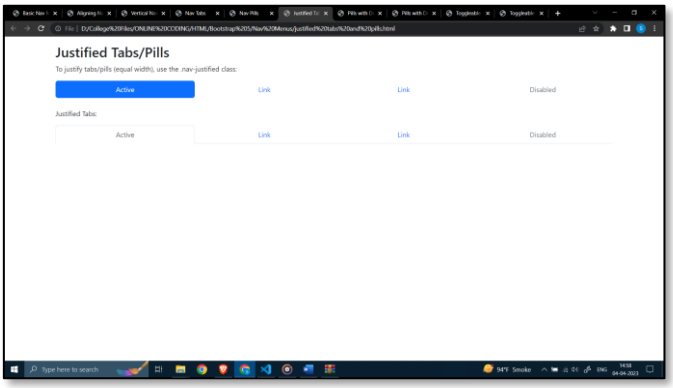
Pills

Turn the nav menu into navigation pills with the .nav-pills class.

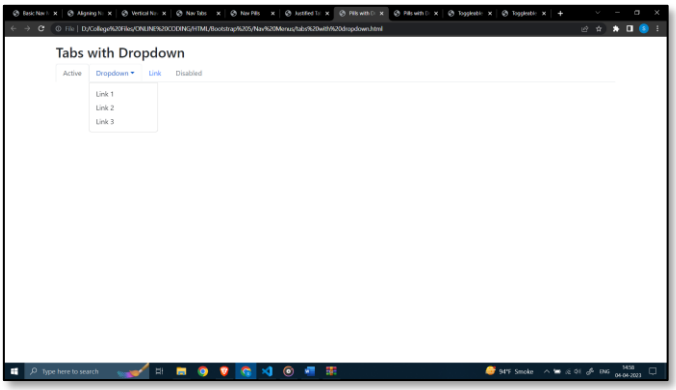


Justified Tabs/Pills

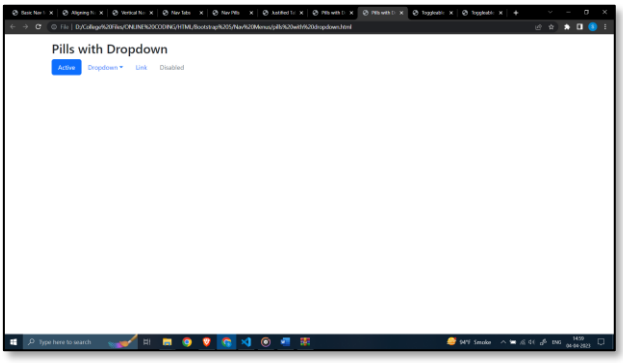
Justify the tabs/pills with the .nav-justified class (equal width):



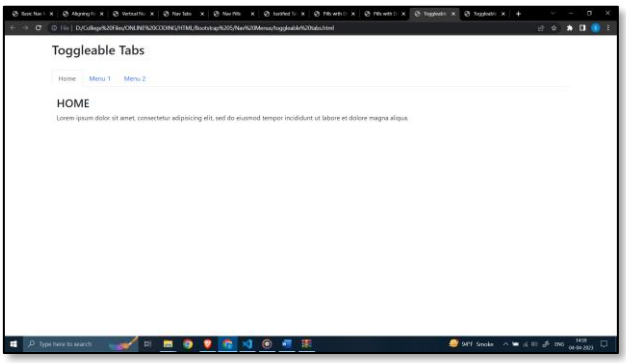
Tabs with Dropdown



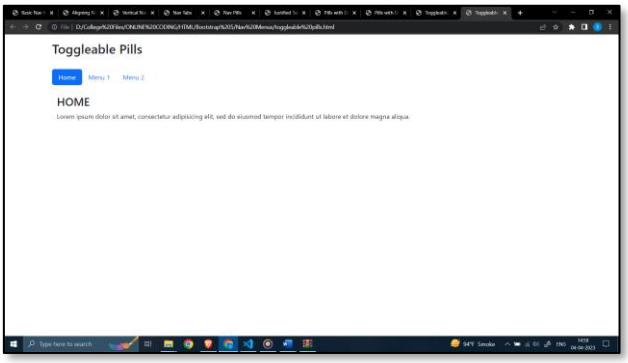
Pills with Dropdown



Toggleable Tabs

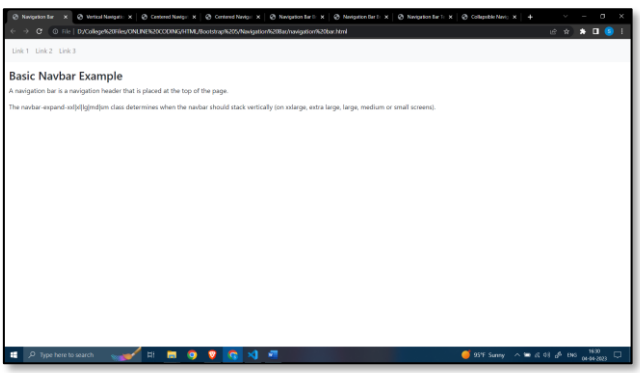


Toggleable Pills



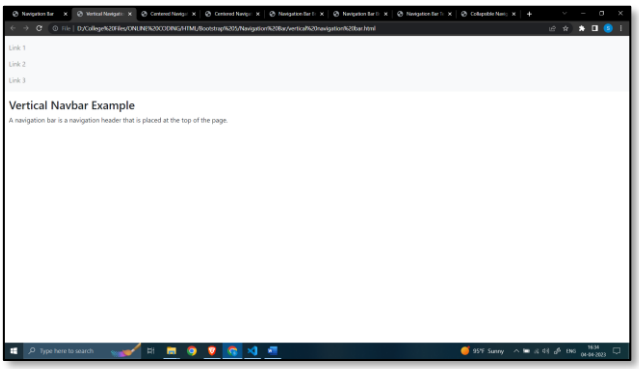
Navigation Bar

A navigation bar is a navigation header that is placed at the top of the page. With Bootstrap, a navigation bar can extend or collapse, depending on the screen size. A standard navigation bar is created with the **.navbar** class, followed by a responsive collapsing class: **.navbar-expand-xxl|xl|lg|md|sm** (stacks the navbar vertically on xxlarge, extra large, large, medium or small screens). To add links inside the navbar, use either an <ul> element (or a) with class="navbar-nav". Then add <li> elements with a **.nav-item** class followed by an <a> element with a **.nav-link** class:



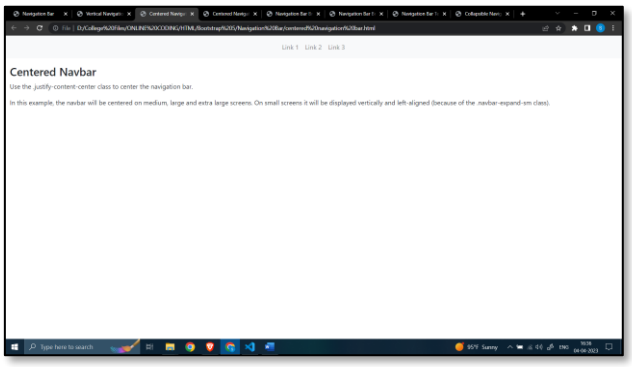
Vertical Navbar

Remove the **.navbar-expand-*** class to create a navigation bar that will always be vertical:



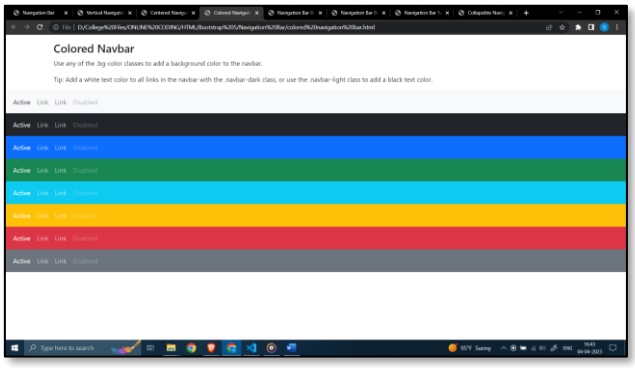
Centered Navbar

Add the **.justify-content-center** class to center the navigation bar:



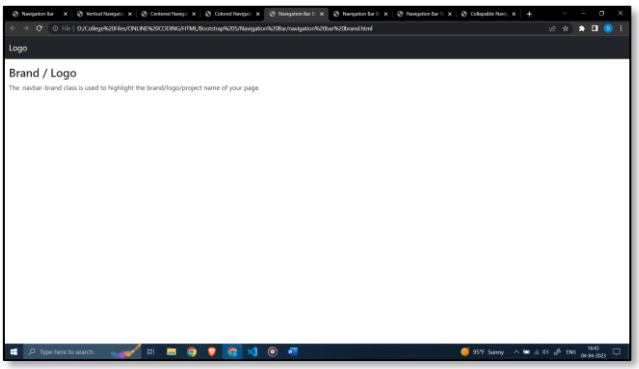
Colored Navbar

Use any of the **.bg-color** classes to change the background color of the navbar

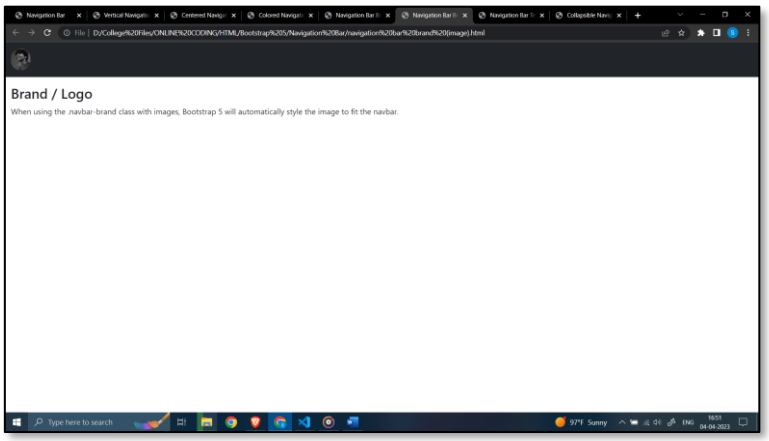


Brand / Logo

The .navbar-brand class is used to highlight the brand/logo/project name of your page:

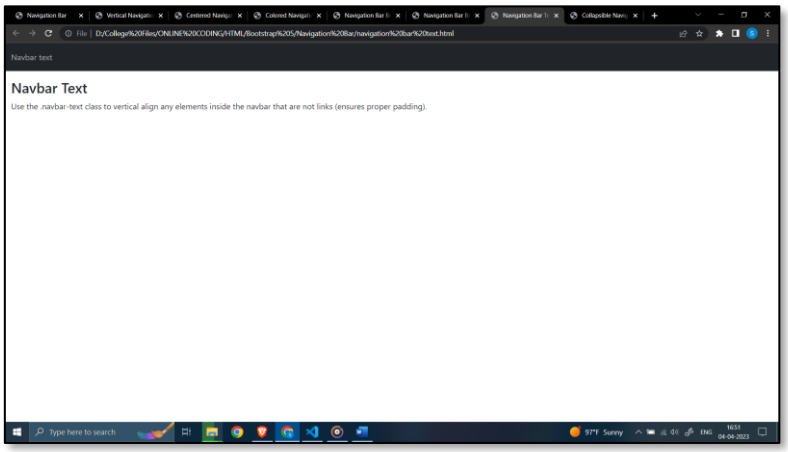


When using the .navbar-brand class with images, Bootstrap 5 will automatically style the image to fit the navbar vertically.



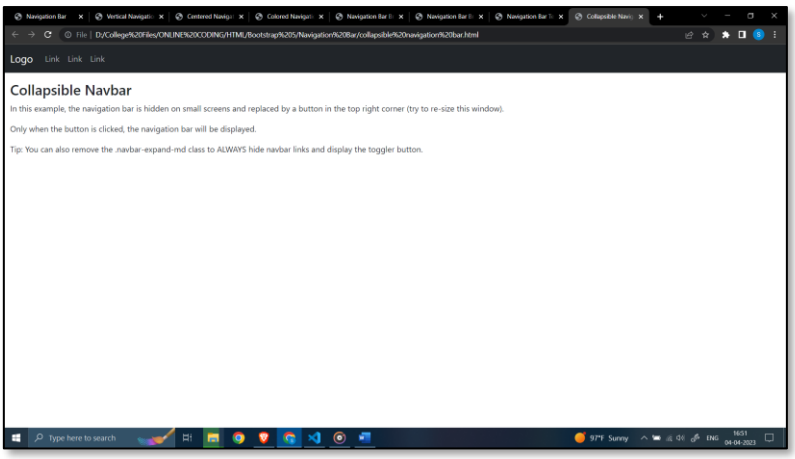
Navbar text

Use the **.navbar-text** class to vertical align any elements inside the navbar that are not links (ensures proper padding and text color).



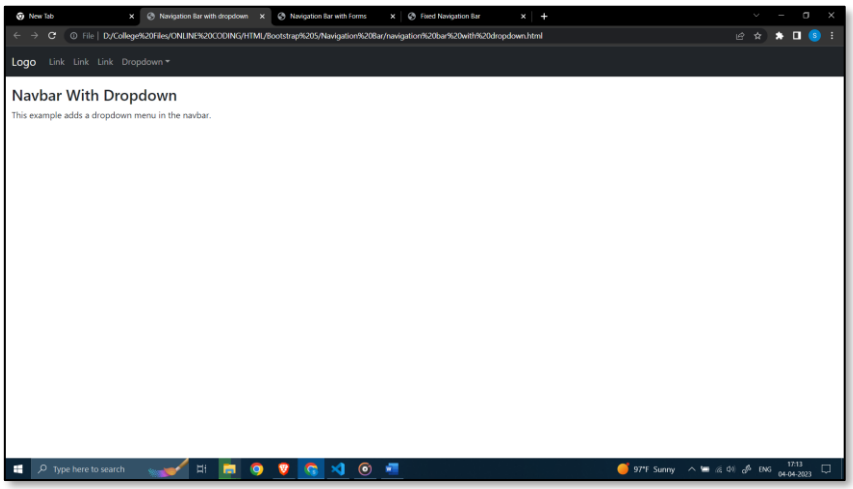
Collapsible Navbar

Very often, especially on small screens, you want to hide the navigation links and replace them with a button that should reveal them when clicked on. To create a collapsible navigation bar, use a button with class="navbar-toggler", data-bs-toggle="collapse" and data-bs-target="#thetarget". Then wrap the navbar content (links, etc) inside a element with class="collapse navbar-collapse", followed by an id that matches the data-bs-target of the button: "thetarget".



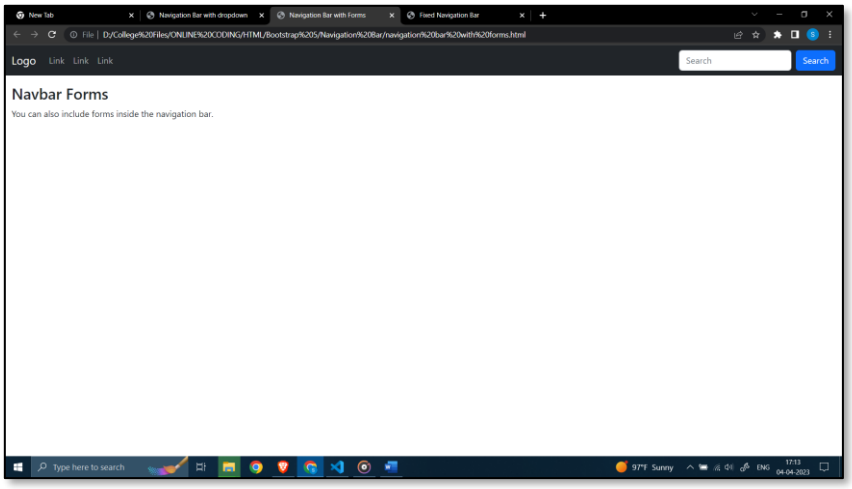
Navbar Dropdown

Navbars can also hold dropdown menus:



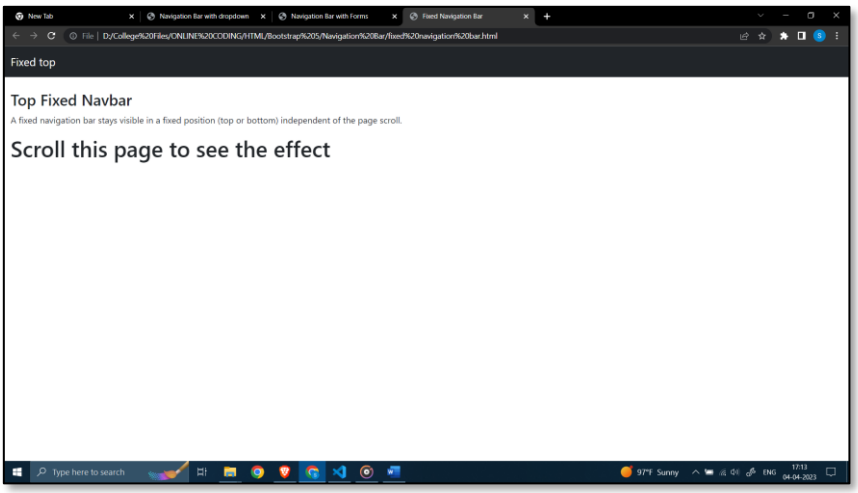
Navbar Forms and Buttons

You can also include forms inside the navigation bar:



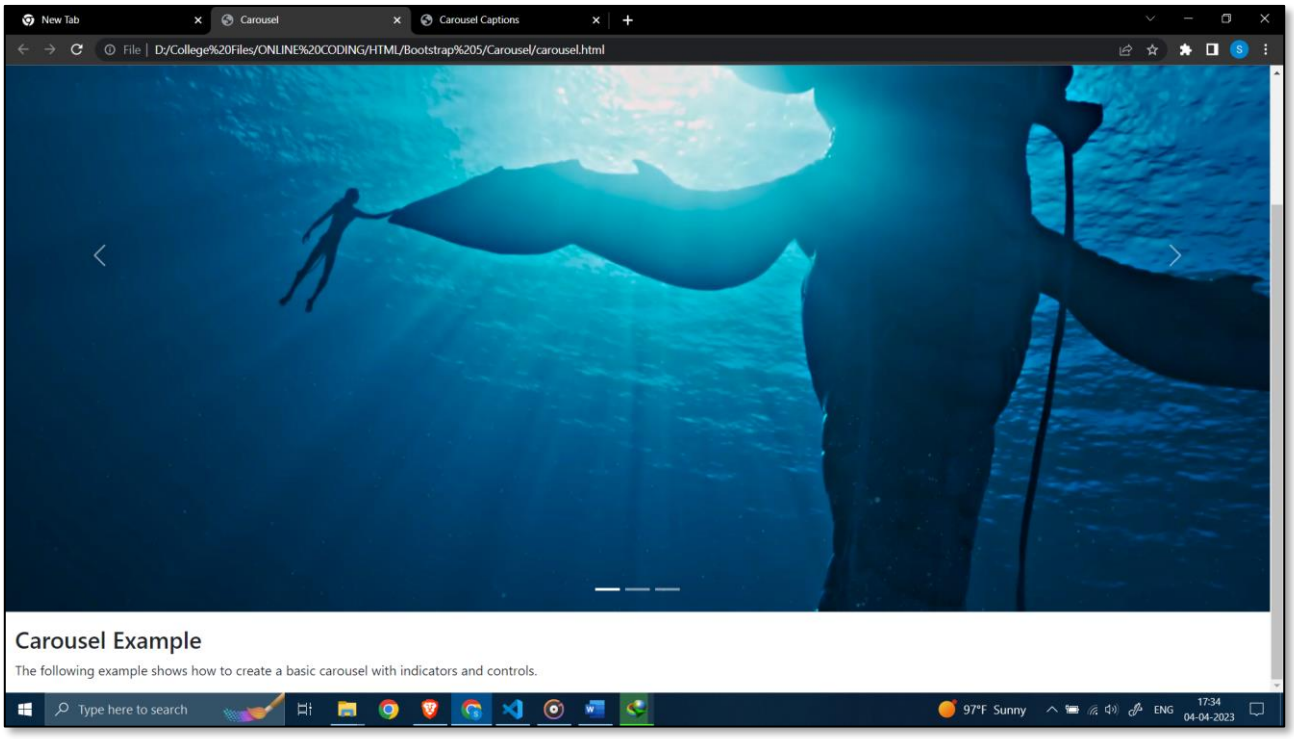
Fixed Navbar

The navigation bar can also be fixed at the top or at the bottom of the page. A fixed navigation bar stays visible in a fixed position (top or bottom) independent of the page scroll. The **.fixed-top** class makes the navigation bar fixed at the top. Use the **.fixed-bottom** class to make the navbar stay at the bottom of the page. Use the **.sticky-top** class to make the navbar fixed/stay at the top of the page when you scroll past it.



Carousel

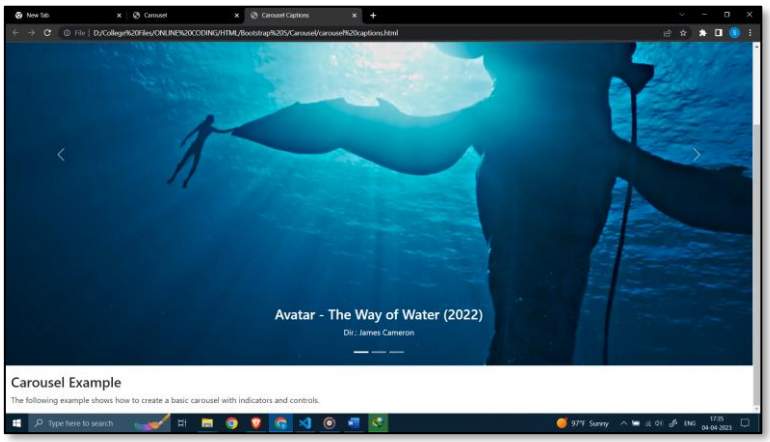
A Carousel is a slideshow for cycling through elements. The following example shows how to create a basic carousel with indicators and controls:



A description of what each class from the example above do:

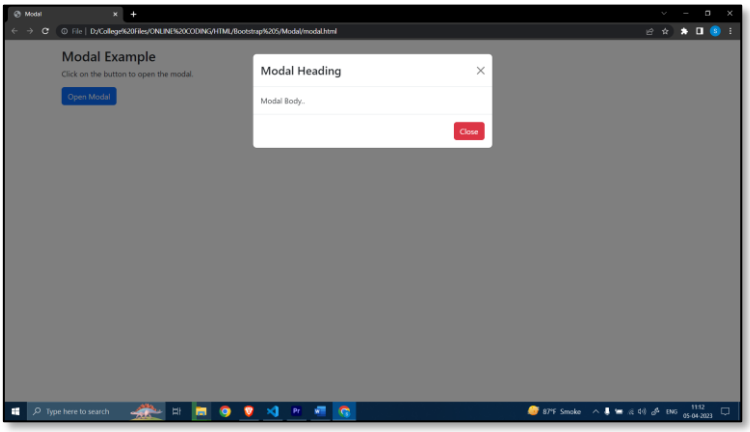
Class	Description
.carousel	Creates a carousel
.carousel-indicators	Adds indicators for the carousel. These are the little dots at the bottom of each slide (which indicates how many slides there are in the carousel, and which slide the user are currently viewing)
.carousel-inner	Adds slides to the carousel
.carousel-item	Specifies the content of each slide
.carousel-control-prev	Adds a left (previous) button to the carousel, which allows the user to go back between the slides
.carousel-control-next	Adds a right (next) button to the carousel, which allows the user to go forward between the slides
.carousel-control-prev-icon	Used together with .carousel-control-prev to create a "previous" button
.carousel-control-next-icon	Used together with .carousel-control-next to create a "next" button
.slide	Adds a CSS transition and animation effect when sliding from one item to the next. Remove this class if you do not want this effect

Add elements inside within each to create a caption for each slide:



Modals

The Modal component is a dialog box/popup window that is displayed on top of the current page. The following example shows how to create a basic modal:



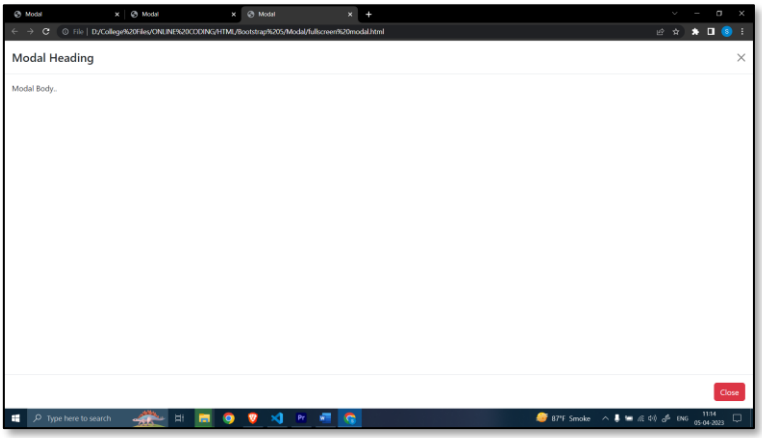
Use the **modal fade** class to add a fading effect when opening and closing the modal.

Modal Size

Change the size of the modal by adding the **.modal-sm** class for small modals (max-width 300px), **.modal-lg** class for large modals (max-width 800px), or **.modal-xl** for extra large modals (max-width 1140px). Default is 500px max-width. Add the size class to the element with class **.modal-dialog**.

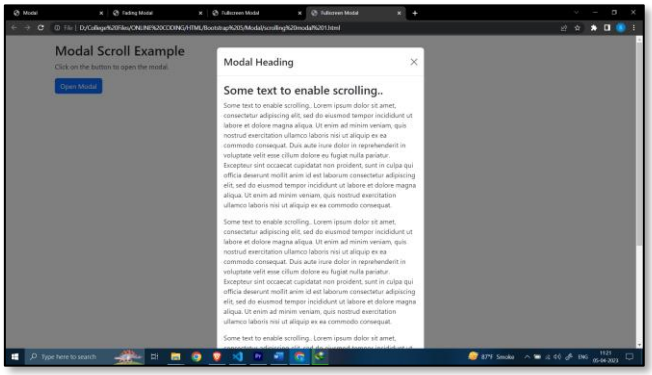
Fullscreen Modals

If you want the modal to span the whole width and height of the page, use the **.modal-fullscreen** class. You can also control when the modal should be in fullscreen, with the **.modal-fullscreen-*-*** classes:

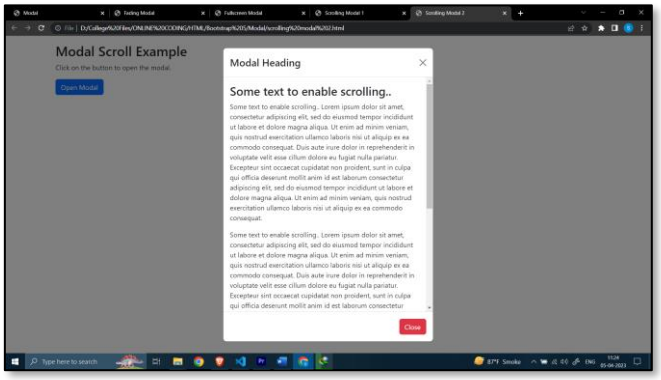


Scrolling Modal

When you have a lot of content inside the modal, a scrollbar is added to the page. See the examples below to understand it:

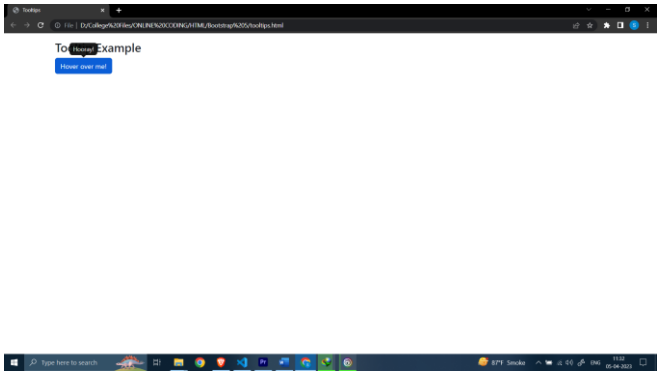


However, it is possible to only scroll inside the modal, instead of the page itself, by adding .modal-dialog-scrollable to .modal-dialog:



Tooltips

The Tooltip component is small pop-up box that appears when the user moves the mouse pointer over an element. To create a tooltip, add the data-bs-toggle="tooltip" attribute to an element. Use the title attribute to specify the text that should be displayed inside the tooltip. Tooltips must be initialized with JavaScript to work:

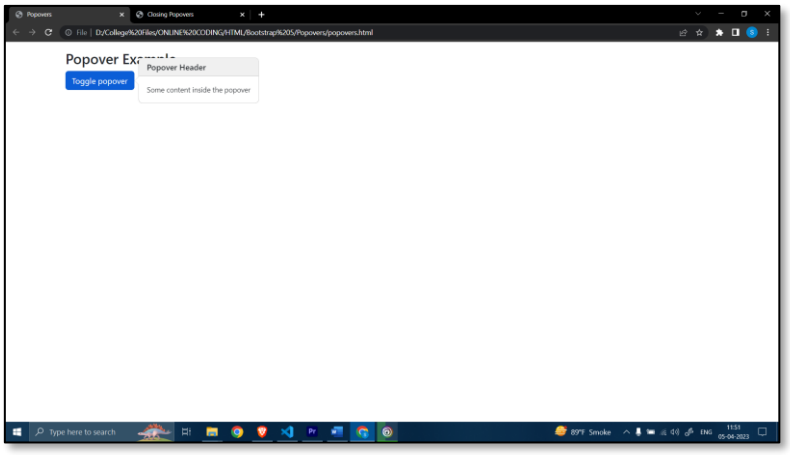


By default, the tooltip will appear on top of the element. Use the data-bs-placement attribute to set the position of the tooltip on top, bottom, left or the right side of the element:

```
<a href="#" data-bs-toggle="tooltip" data-bs-placement="top" title="Hooray!">Hover</a>

<a href="#" data-bs-toggle="tooltip" data-bs-placement="bottom" title="Hooray!">Hover</a>

<a href="#" data-bs-toggle="tooltip" data-bs-placement="left" title="Hooray!">Hover</a>

<a href="#" data-bs-toggle="tooltip" data-bs-placement="right" title="Hooray!">Hover</a>
```

Popovers

The Popover component is similar to tooltips; it is a pop-up box that appears when the user clicks on an element. The difference is that the popover can contain much more content. To create a popover, add the data-bs-toggle="popover" attribute to an element. Use the title attribute to specify the header text of the popover, and use the data-bs-content attribute to specify the text that should be displayed inside the popover's body. Popovers must be initialized with JavaScript to work:

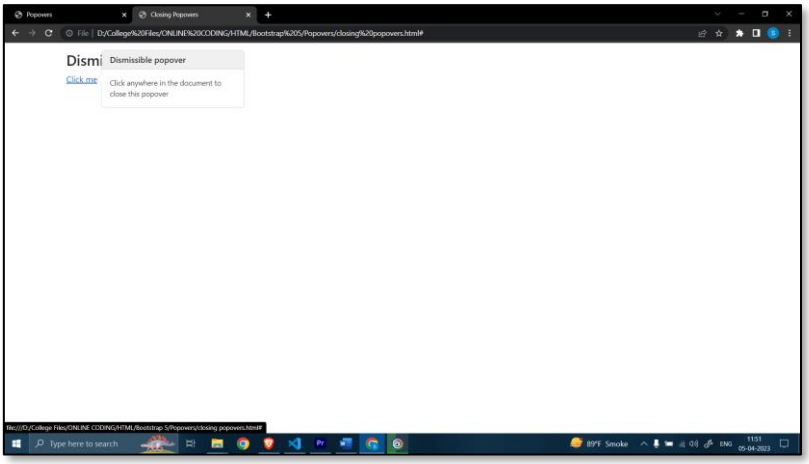


By default, the popover will appear on the right side of the element. Use the data-bs-placement attribute to set the position of the popover on top, bottom, left or the right side of the element. The placement attributes do not work as you expect if it is not enough room for them. For example: if you use the top placement at the top of a page (where it is no room for it), it will instead display the popover below the element or to the right (wherever it is room for it):

```
<a href="#" title="Header" data-bs-toggle="popover" data-bs-placement="top" data-content="Content">Top</a>

<a href="#" title="Header" data-bs-toggle="popover" data-bs-placement="bottom" data-content="Content">Bottom</a>

<a href="#" title="Header" data-bs-toggle="popover" data-bs-placement="left" data-content="Content">Left</a>

<a href="#" title="Header" data-bs-toggle="popover" data-bs-placement="right" data-content="Content">Right</a>
```

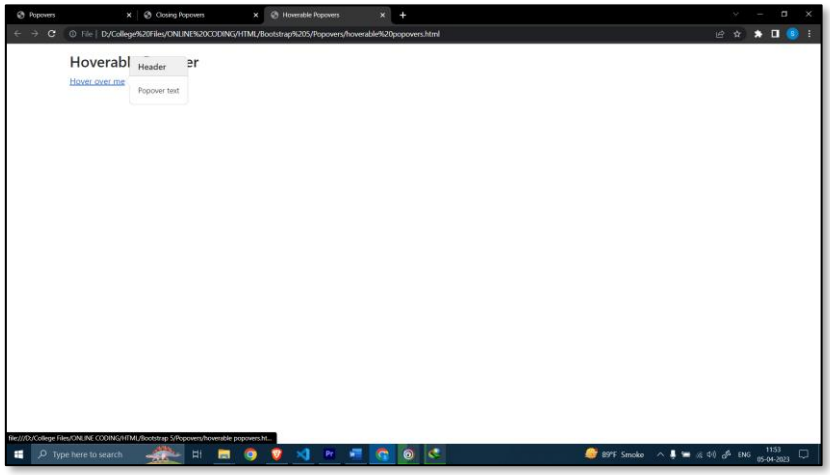
Closing Popovers

By default, the popover is closed when you click on the element again. However, you can use the data-bs-trigger="focus" attribute which will close the popover when clicking outside the element:



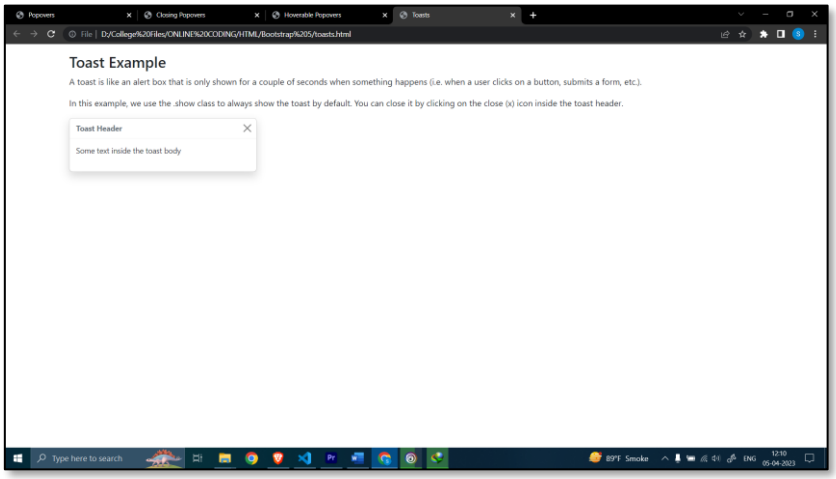
Hoverable Popovers

If you want the popover to be displayed when you move the mouse pointer over the element, use the data-bs-trigger attribute with a value of hover:



Toasts

The toast component is like an alert box that is only shown for a couple of seconds when something happens (i.e. when the user clicks on a button, submits a form, etc.). To create a toast, use the **.toast** class, add a **.toast-header** and a **.toast-body** inside of it. Toasts are hidden by default. Use the **.show** class if you want to display it. To close it, use a <button> element and add **data-bs-dismiss="toast"**:

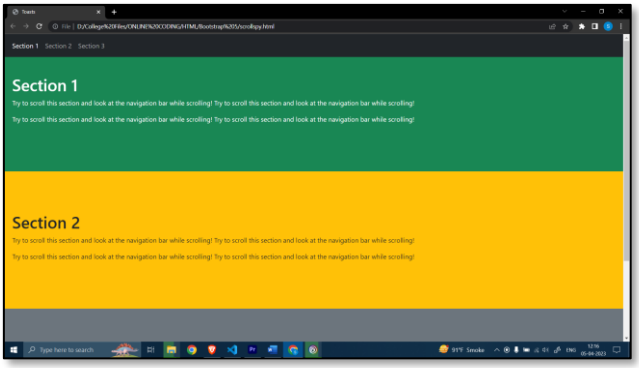


To show a toast with a click of a button, you must initialize it with JavaScript. Select the specified element and call the toast() method.

```
<script>

document.getElementById("toastbtn").onclick = function() {

  var toastElList = [].slice.call(document.querySelectorAll('.toast'))

  var toastList = toastElList.map(function(toastEl) {

    return new bootstrap.Toast(toastEl)

  })

  toastList.forEach(toast => toast.show())

}

</script>
```

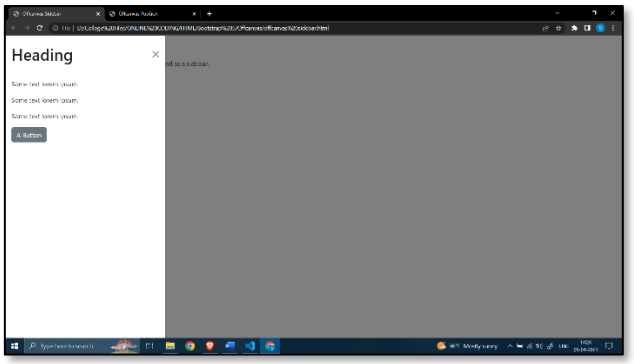
Scrollspy

Scrollspy is used to automatically update links in a navigation list based on scroll position. Add data-bs-spy="scroll" to the element that should be used as the scrollable area (often this is the <body> element). Then add the data-bs-target attribute with a value of the id or the class name of the navigation bar (.navbar). This is to make sure that the navbar is connected with the scrollable area. Note that scrollable elements must match the ID of the links inside the navbar's list items (matches <a href="#section1">). The optional data-bs-offset attribute specifies the number of pixels to offset from top when calculating the position of scroll. This is useful when you feel that the links inside the navbar changes the active state too soon or too early when jumping to the scrollable elements. Default is 10 pixels.



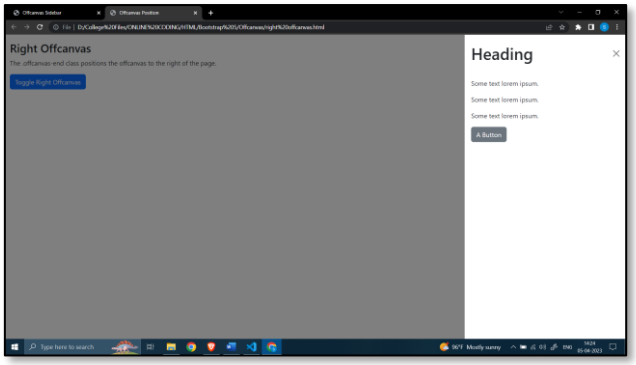
Offcanvas

Offcanvas is similar to modals (hidden by default and shown when activated), except that is often used as a sidebar navigation menu. The **.offcanvas** class creates the offcanvas sidebar. The **.offcanvas-start** class positions the offcanvas, and makes it 400px wide. See examples below for more positioning classes. The **.offcanvas-title** class ensures proper margins and line-height. Then, add your content inside the **.offcanvas-body** class. To open the offcanvas sidebar, you must use a <button> or an <a> element that points to the id of the **.offcanvas container**. To open the offcanvas sidebar with an <a> element, you can point to #demo with the href attribute, instead of data-bs-target attribute.



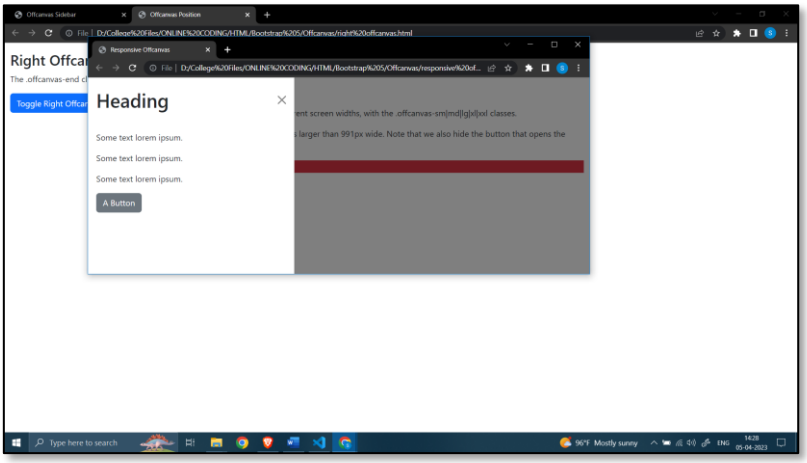
Offcanvas Position

Use the **.offcanvas-start|end|top|bottom** to position the offcanvas to the left, right, top or bottom:



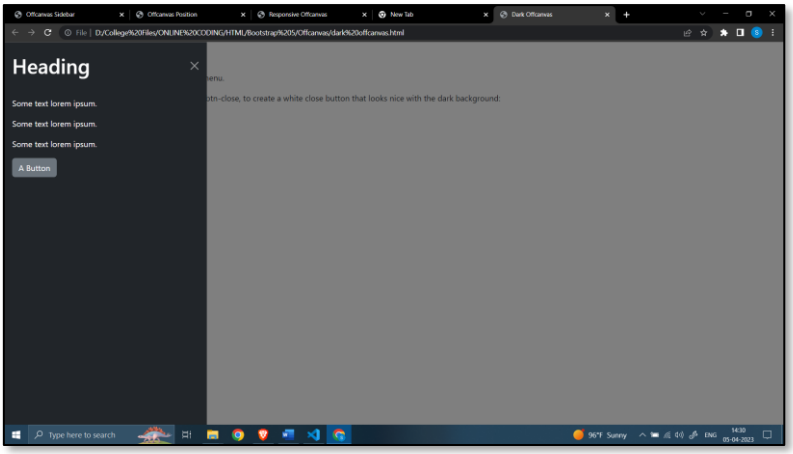
Responsive OffCanvas Menu

You can also control when you want to hide or show the offcanvas menu on different screen widths, with the **.offcanvas-sm|md|lg|xl|xxl** classes:



Dark OffCanvas Menu

Use the **.text-bg-dark** class to create a dark offcanvas menu. We have also added the **.btn-close-white** class to **.btn-close**, to create a white close button that looks nice with the dark background:

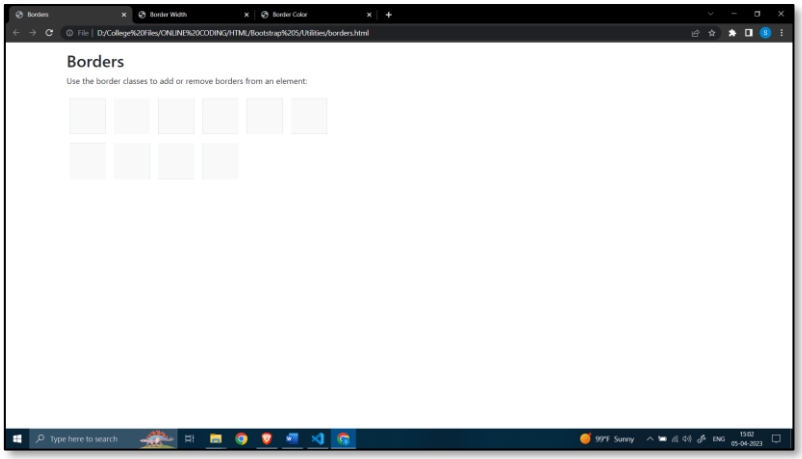


Utilities

Bootstrap 5 has a lot of utility/helper classes to quickly style elements without using any CSS code.

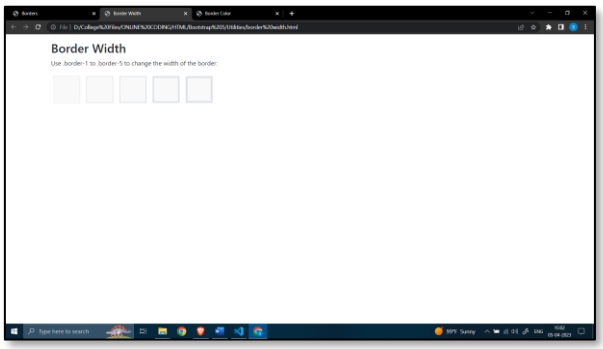
Borders

Use the border classes to add or remove borders from an element:



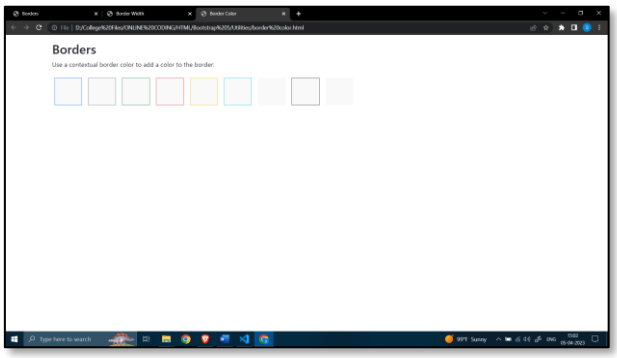
Border Width

Use **.border-1** to **.border-5** to change the width of the border:



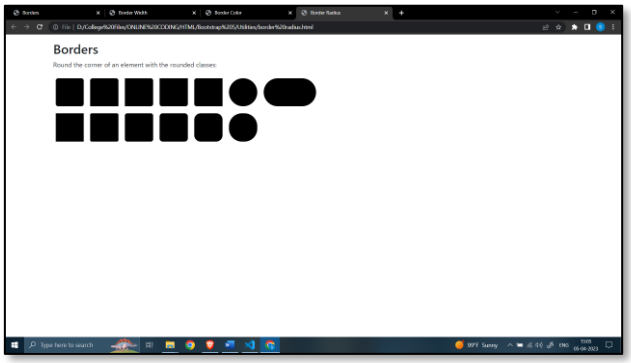
Border Color

Add a color to the border with any of the contextual border color classes:



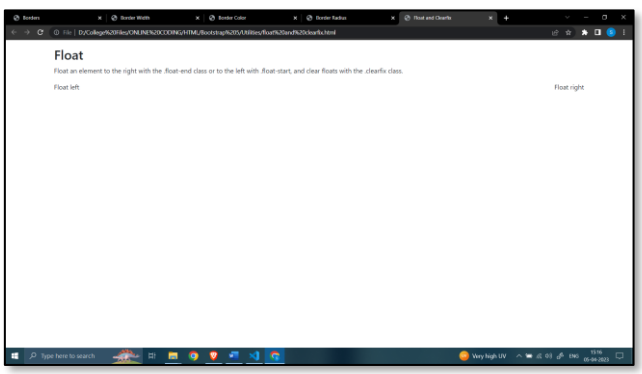
Border Radius

Add rounded corners to an element with the rounded classes:



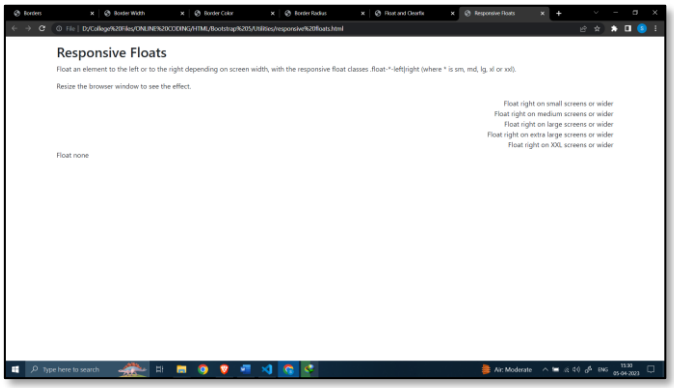
Float and Clearfix

Float an element to the right with the **.float-end** class or to the left with **.float-start**, and clear floats with the **.clearfix** class:



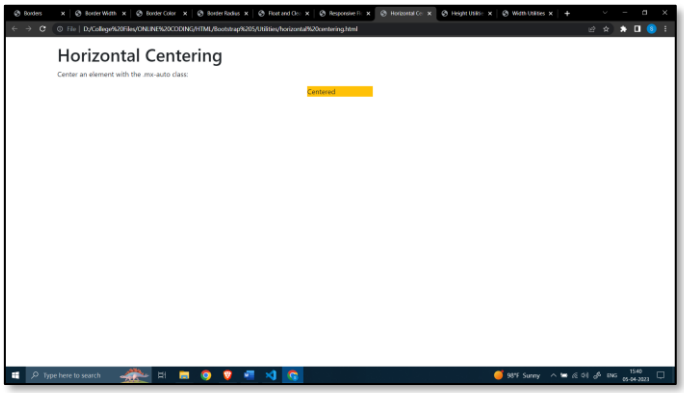
Responsive Floats

Float an element to the left or to the right depending on screen width, with the responsive float classes:



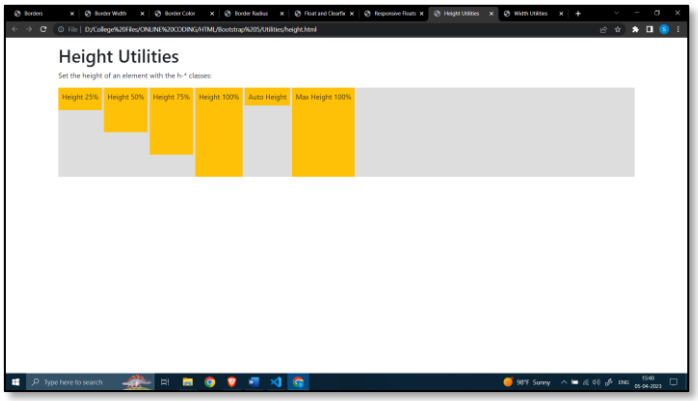
Center Align

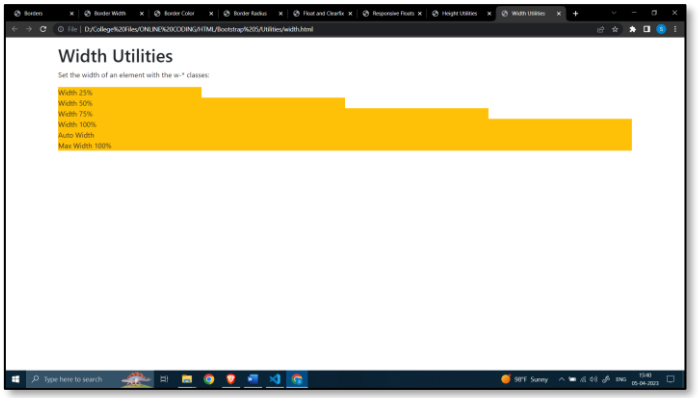
Center an element with the **.mx-auto** class (adds margin-left and margin-right: auto):



Height and Width

Set the height and width of an element with the **h-*** and **w-*** classes:





Spacing

Bootstrap 5 has a wide range of responsive margin and padding utility classes. They work for all breakpoints. The classes are used in the format:

{property}{sides}-{size} for xs and {property}{sides}-{breakpoint}-{size} for sm, md, lg, xl and xxl.

Where property is one of:

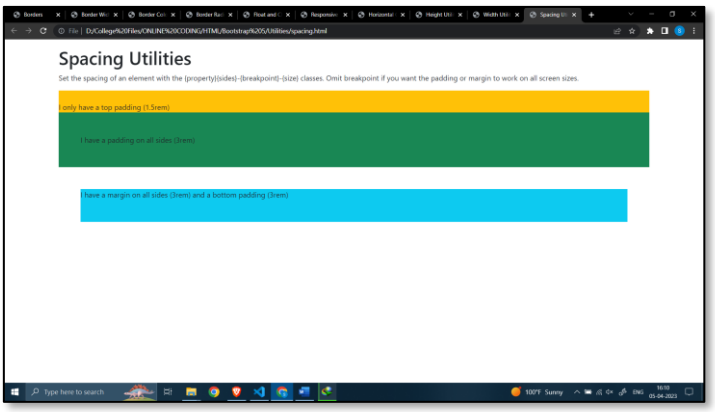
- m: sets margin
- p: sets padding

Where sides is one of:

- t: sets margin-top or padding-top
- b: sets margin-bottom or padding-bottom
- s: sets margin-left or padding-left
- e: sets margin-right or padding-right
- x: sets both padding-left and padding-right or margin-left and margin-right
- y: sets both padding-top and padding-bottom or margin-top and margin-bottom
- blank: sets a margin or padding on all 4 sides of the element

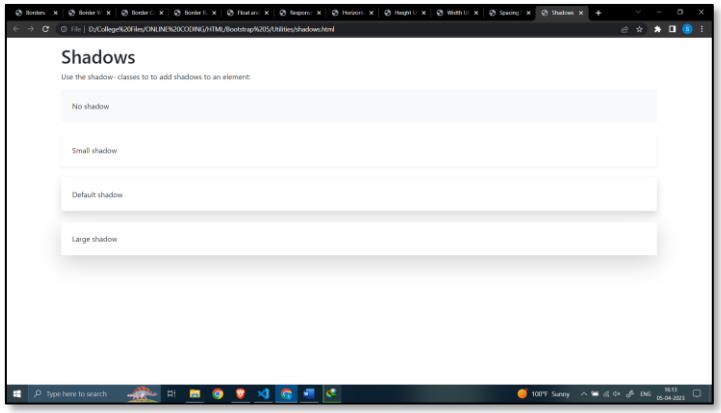
Where size is one of:

- 0: sets margin or padding to 0
- 1: sets margin or padding to .25rem
- 2: sets margin or padding to .5rem
- 3: sets margin or padding to 1rem
- 4: sets margin or padding to 1.5rem
- 5: sets margin or padding to 3rem
- auto: sets margin to auto



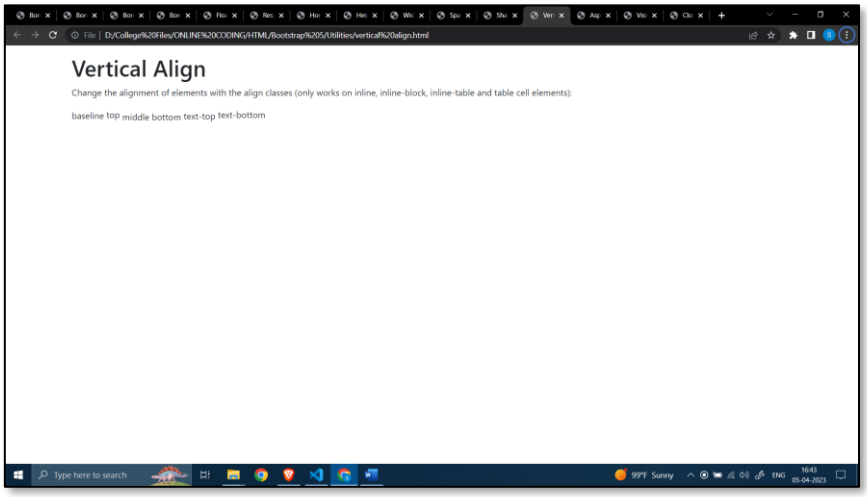
Shadow

Use the **shadow-*** classes to add shadows to an element:



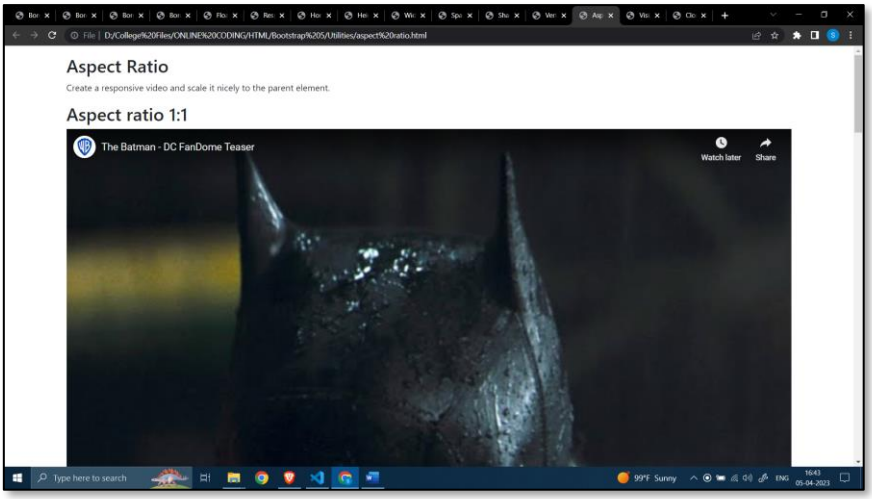
Vertical Align

Use the **align-*** classes to change the alignment of elements (only works on inline, inline-block, inline-table and table cell elements):



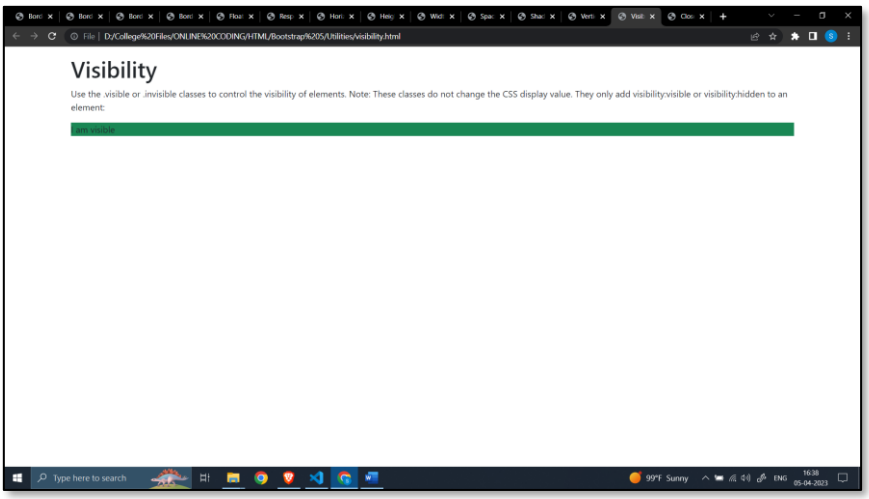
Aspect Ratio

Create responsive video or slideshows based on the width of the parent. Add the **.ratio** class together with an aspect ratio of your choice **.ratio-*** to a parent element, and add the embed (video or iframe) inside of it:



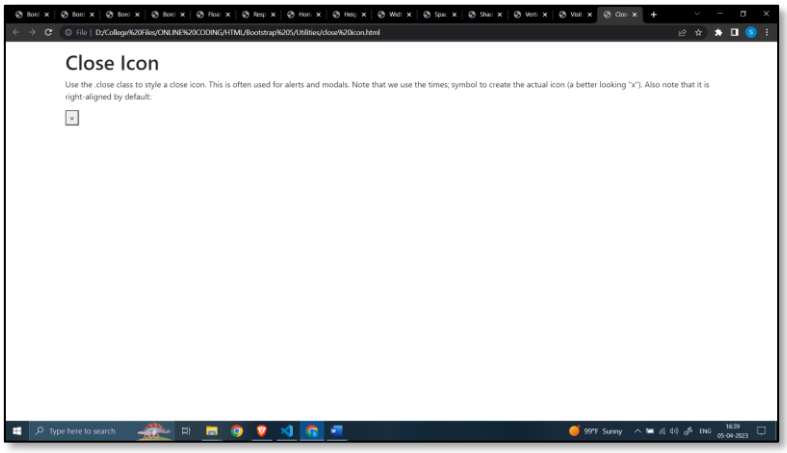
Visibility

Use the **.visible** or **.invisible** classes to control the visibility of elements. These classes do not change the CSS display value. They only add visibility:visible or visibility:hidden:



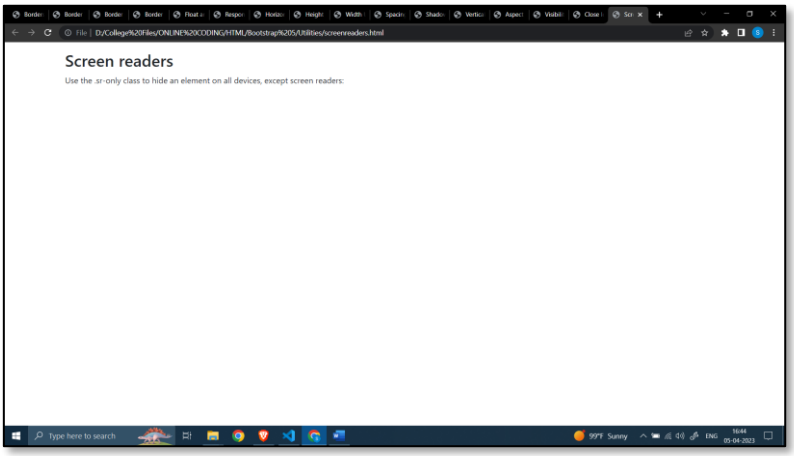
Close icon

Use the **.btn-close** class to style a close icon. This is often used for alerts and modals.



Screenreaders

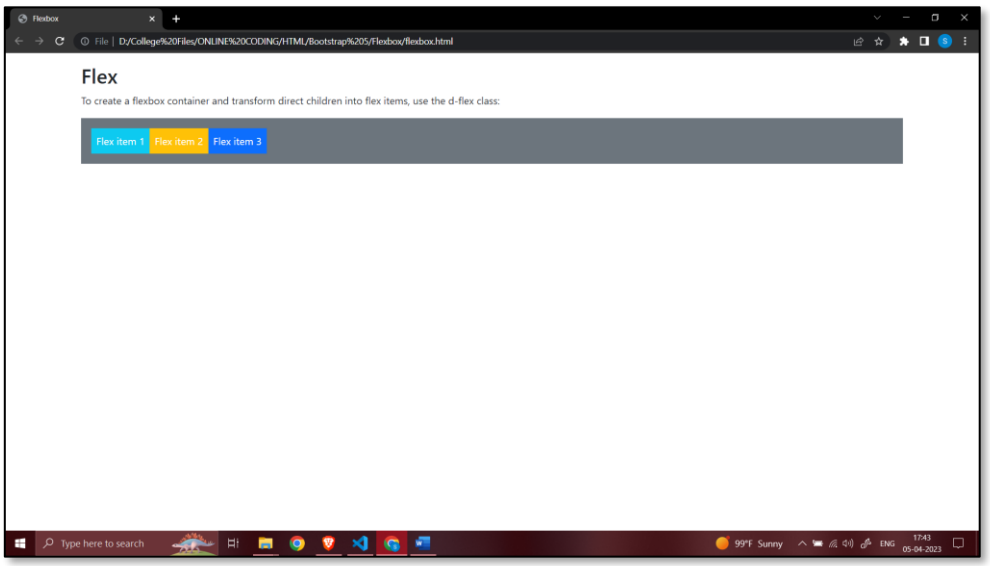
Use the **.visually-hidden** class to hide an element on all devices, except screen readers:



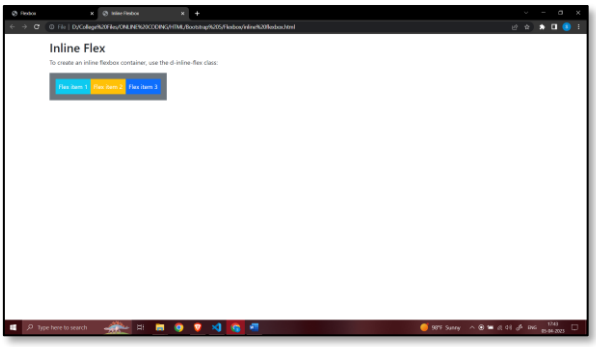
Flexbox

The biggest difference between Bootstrap 3 and Bootstrap 4 & 5 is that Bootstrap 5 now uses flexbox, instead of floats, to handle the layout. The Flexible Box Layout Module, makes it easier to design flexible responsive layout structure without using float or positioning.

To create a flexbox container and to transform direct children into flex items, use the **d-flex** class:

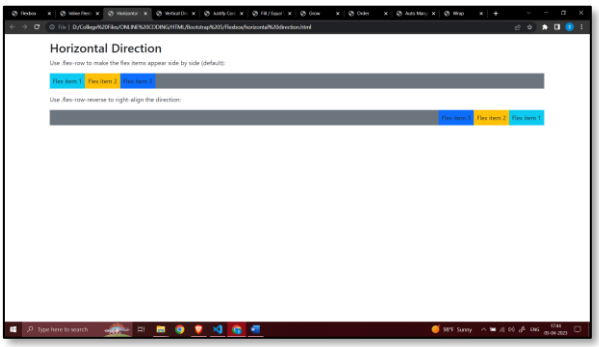


To create an inline flexbox container, use the **d-inline-flex** class:



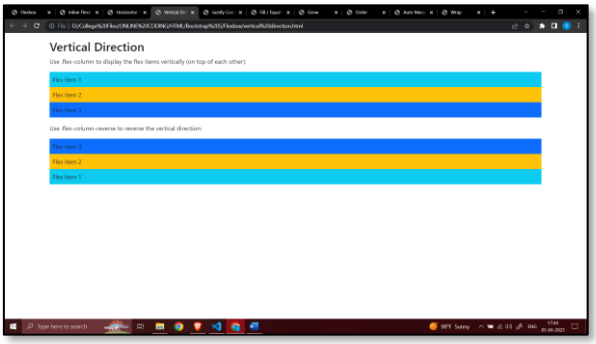
Horizontal Direction

Use **.flex-row** to display the flex items horizontally (side by side). This is default. Use **.flex-row-reverse** to right-align the horizontal direction:



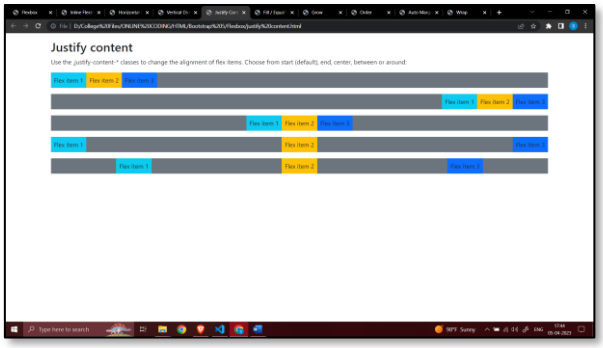
Vertical Direction

Use **.flex-column** to display the flex items vertically (on top of each other), or **.flex-column-reverse** to reverse the vertical direction:



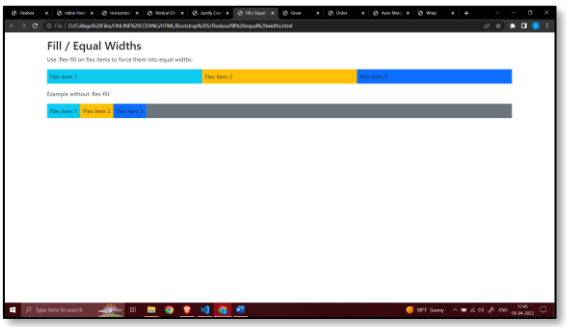
Justify Content

Use the **.justify-content-*** classes to change the alignment of flex items. Valid classes are start (default), end, center, between or around:



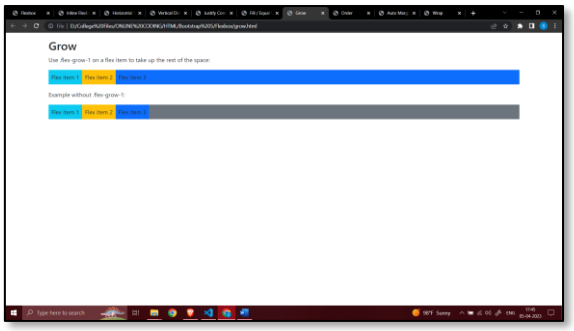
Fill/Equal Widths

Use **.flex-fill** on flex items to force them into equal widths:



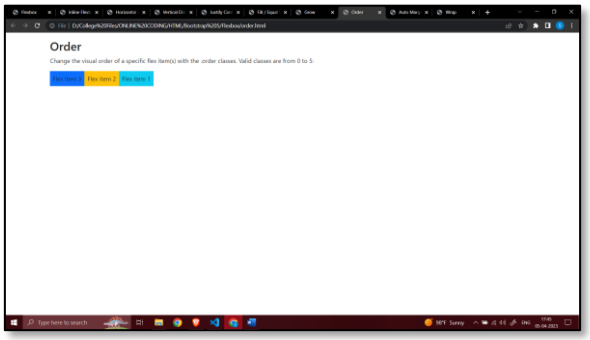
Grow

Use **.flex-grow-1** on a flex item to take up the rest of the space. In the example below, the first two flex items take up their necessary space, while the last item takes up the rest of the available space. Use **.flex-shrink-1** on a flex item to make it shrink if necessary.



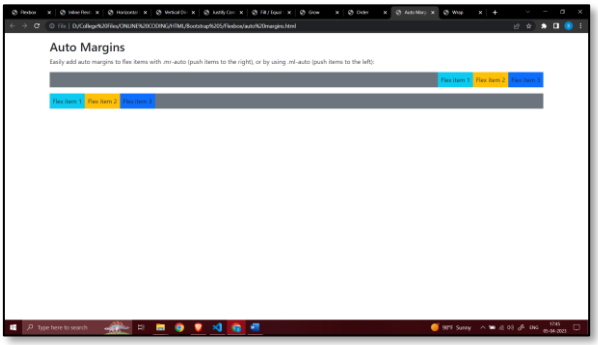
Order

Change the visual order of a specific flex item(s) with the **.order** classes. Valid classes are from 0 to 5, where the lowest number has highest priority (order-1 is shown before order-2, etc..):



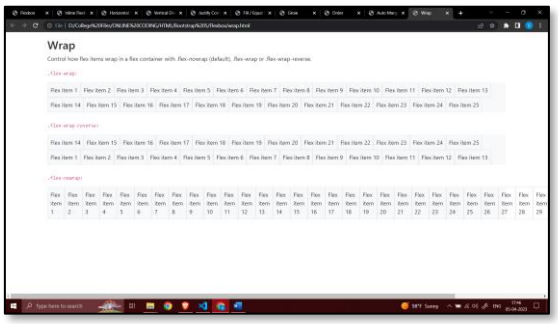
Auto Margins

Easily add auto margins to flex items with **.ms-auto** (push items to the right), or by using **.me-auto** (push items to the left):



Wrap

Control how flex items wrap in a flex container with **.flex-nowrap** (default), **.flex-wrap** or **.flex-wrap-reverse**.

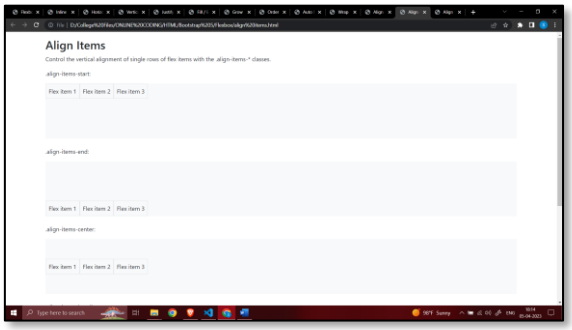


Align Content

Control the vertical alignment of gathered flex items with the .align-content-* classes. Valid classes are .align-content-start (default), .align-content-end, .align-content-center, .align-content-between, .align-content-around and .align-content-stretch. Note: These classes have no effect on single rows of flex items.

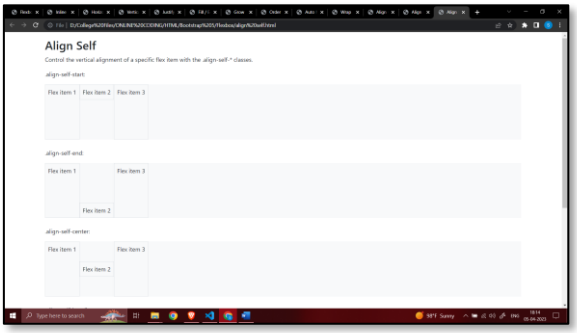
Align Items

Control the vertical alignment of single rows of flex items with the .align-items-* classes. Valid classes are .align-items-start, .align-items-end, .align-items-center, .align-items-baseline, and .align-items-stretch (default).



Align Self

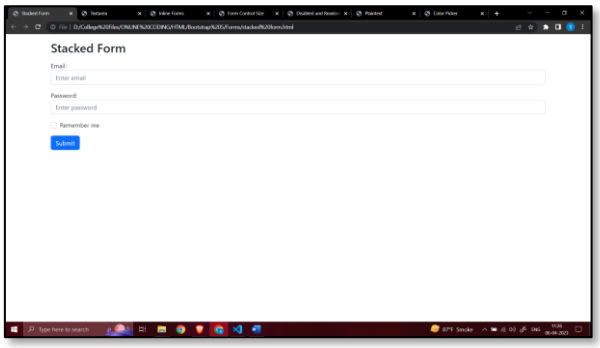
Control the vertical alignment of a specified flex item with the .align-self-* classes. Valid classes are .align-self-start, .align-self-end, .align-self-center, .align-self-baseline, and .align-self-stretch (default).



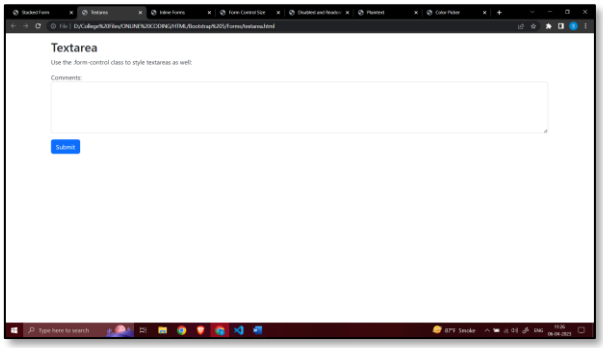
Forms

Stacked Form

All textual <input> and <textarea> elements with class **.form-control** get proper form styling. Also note that we add a **.form-label** class to each label element to ensure correct padding. Checkboxes have different markup. They are wrapped around a container element with **.form-check**, and labels have a class of **.form-check-label**, while checkboxes and radio buttons use **.form-check-input**.

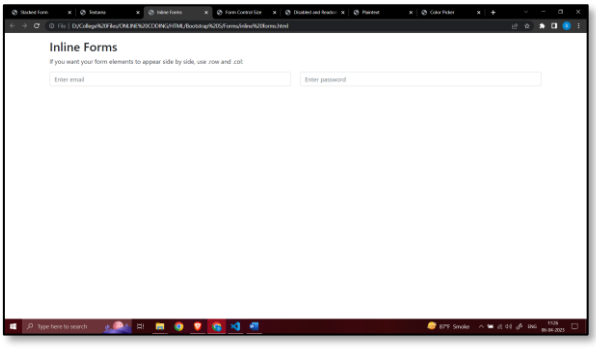


Textarea



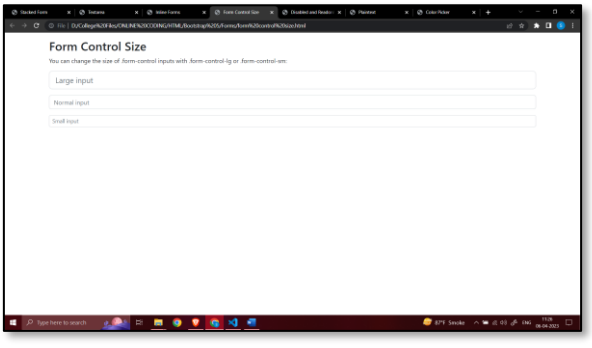
Inline Forms

If you want your form elements to appear side by side, use **.row** and **.col**:



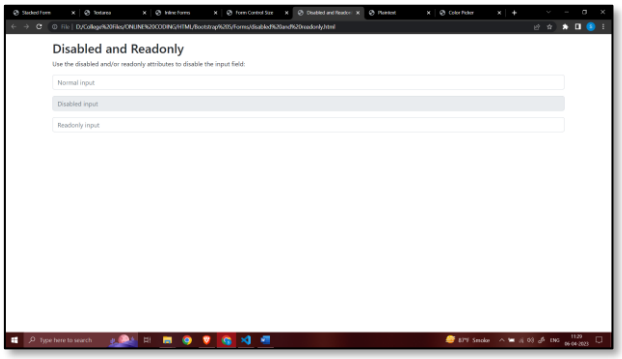
Form Control Size

You can change the size of **.form-control** inputs with **.form-control-lg** or **.form-control-sm**:



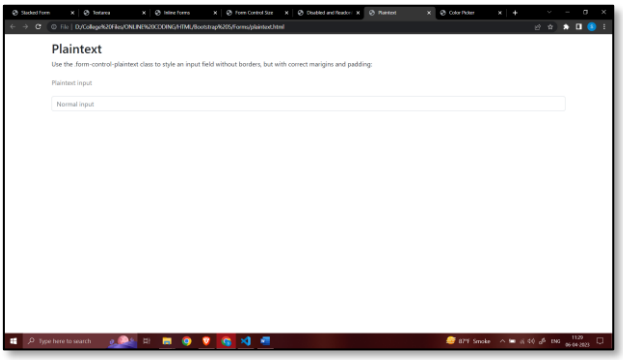
Disabled and Readonly

Use the disabled and/or readonly attributes to disable the input field:



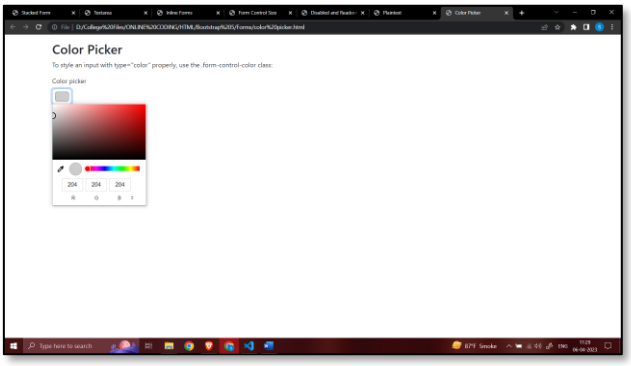
Plaintext input

Use the **.form-control-plaintext** class to style an input field without borders, but keep proper marigins and padding:



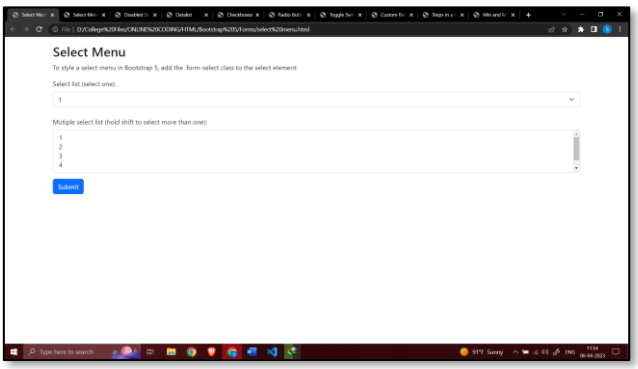
Color Picker

To style an input with type="color" properly, use the **.form-control-color** class:



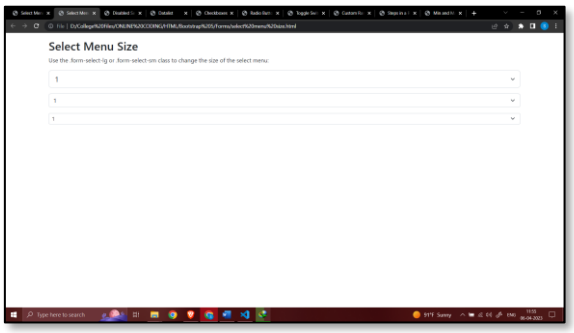
Select Menu

Select menus are used if you want to allow the user to pick from multiple options. To style a select menu in Bootstrap 5, add the **.form-select** class to the <select> element:



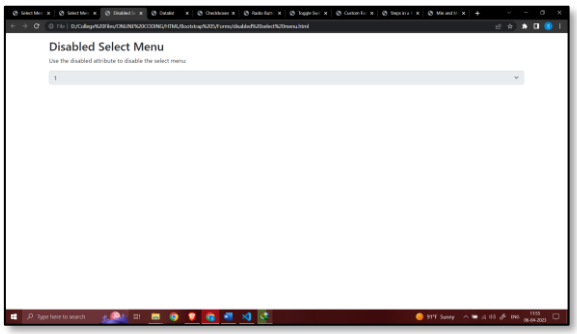
Select Menu Size

Use the **.form-select-lg** or **.form-select-sm** class to change the size of the select menu:



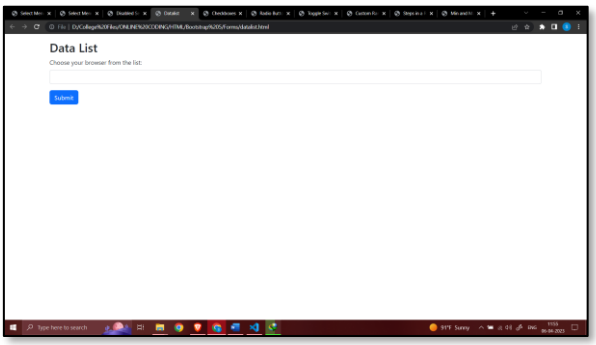
Disabled Select Menu

Use the disabled attribute to disable the select menu:



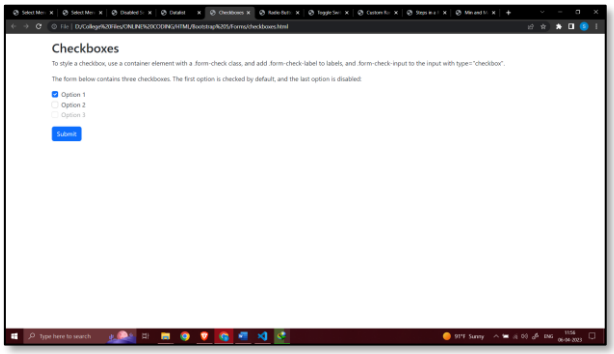
Data Lists

Bootstrap will also style data lists, which is a list of pre-defined options for an <input> element:



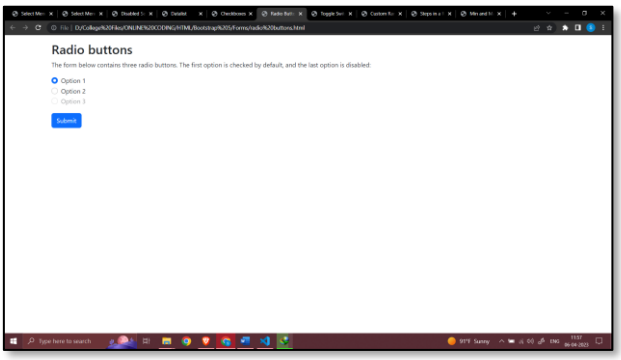
Checkboxes

Checkboxes are used if you want the user to select any number of options from a list of preset options. To style checkboxes, use a wrapper element with class="form-check" to ensure proper margins for labels and checkboxes. Then, add the **.form-check-label** class to label elements, and .form-check-input to style checkboxes properly inside the **.form-check container**. Use the checked attribute if you want the checkbox to be checked by default.



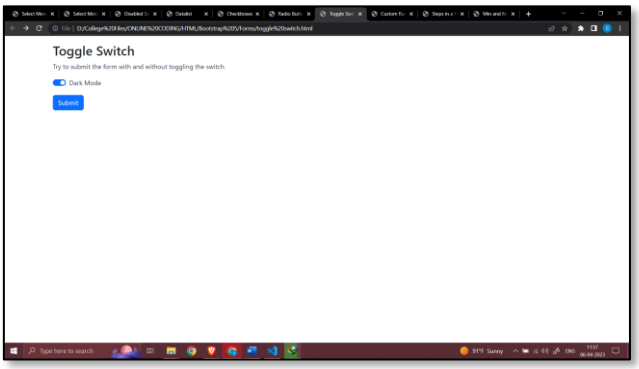
Radio buttons

Radio buttons are used if you want to limit the user to just one selection from a list of preset options.



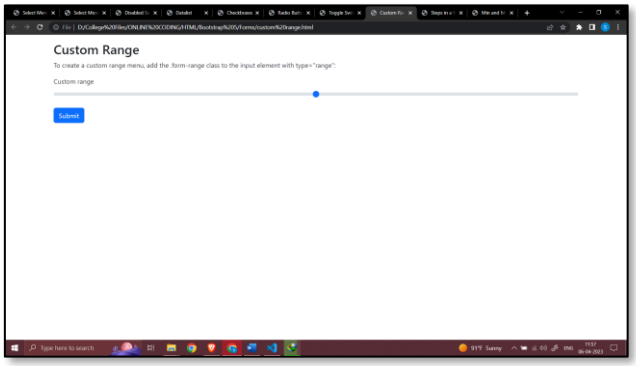
Toggle Switches

If you want your checkbox to be styled as a toggle switch, use the **.form-switch** class together with the **.form-check** container:



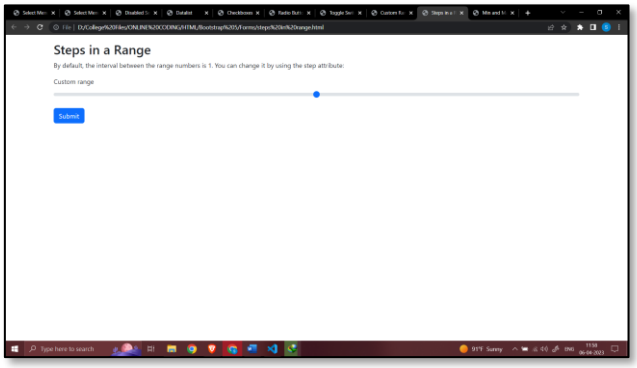
Custom Range

To style a range menu, add the **.form-range** class to the input element with type="range":



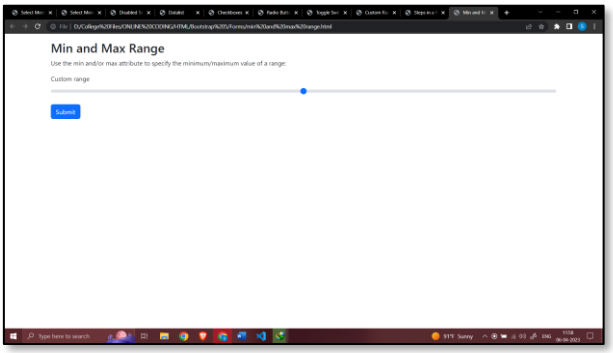
Steps

By default, the interval between the range numbers is 1. You can change it by using the step attribute:



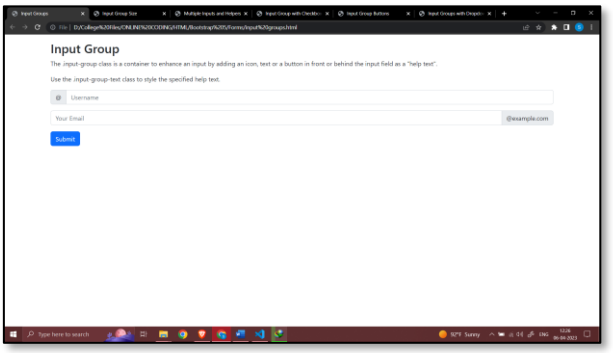
Min and Max

By default, the minimum value is 0 and maximum value is 100. You can use the min and/or max attribute change it:



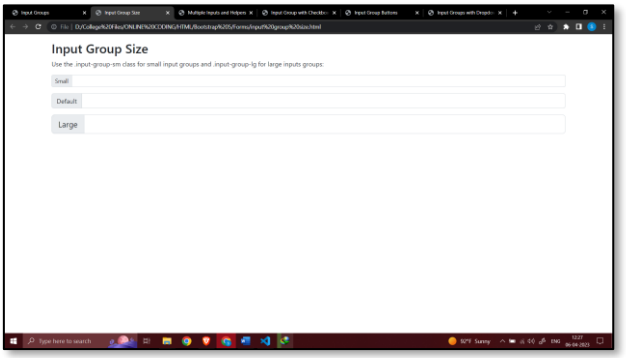
Input Groups

The **.input-group** class is a container to enhance an input by adding an icon, text or a button in front or behind the input field as a "help text". To style the specified help text, use the **.input-group-text** class:



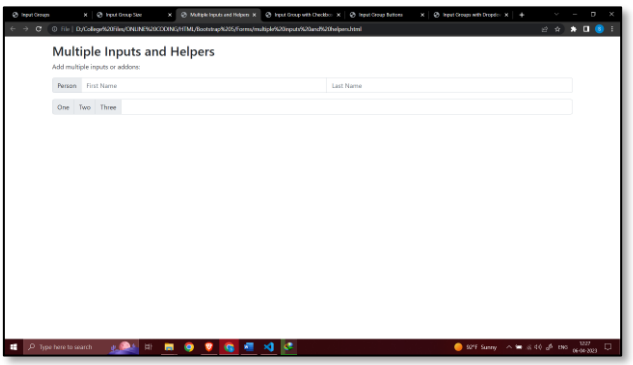
Input Group Size

Use the **.input-group-sm** class for small input groups and **.input-group-lg** for large inputs groups:



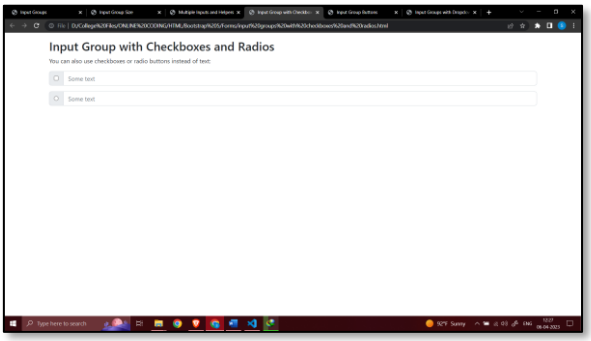
Multiple Inputs and Helpers

Add multiple inputs or addons:

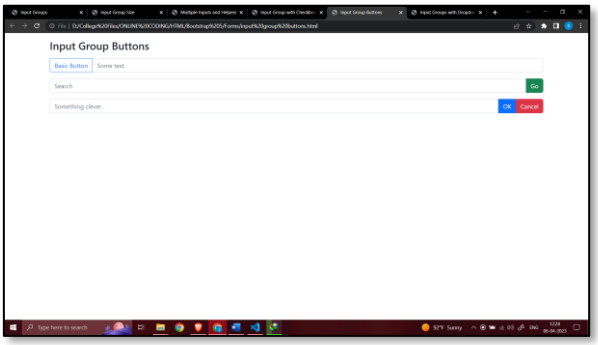


Input Group with Checkboxes and Radios

You can also use checkboxes or radio buttons instead of text:

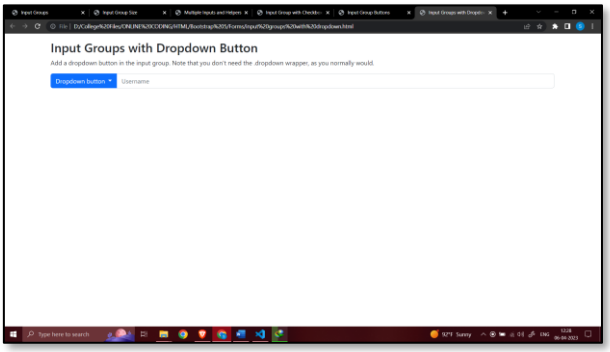


Input Group Buttons



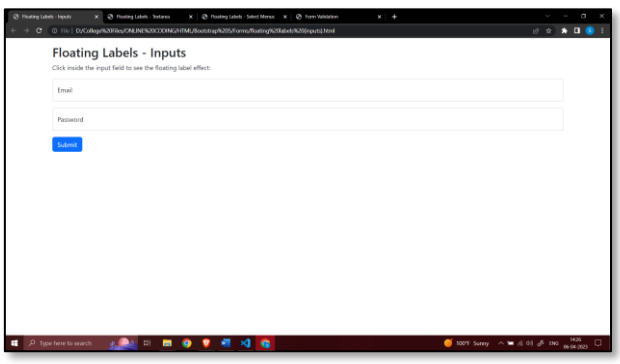
Input Group with Dropdown Button

Add a dropdown button in the input group. Note that you don't need the **.dropdown** wrapper, as you normally would.



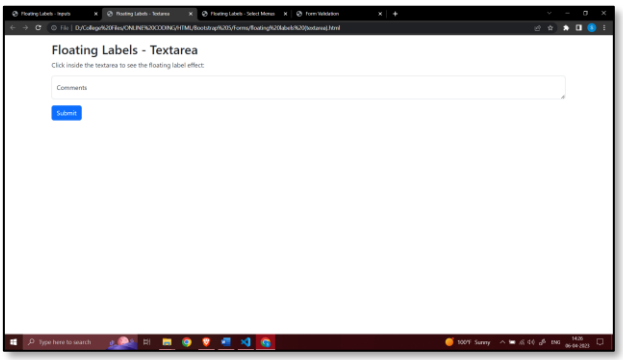
Floating Labels / Animated Labels

By default, when using labels, they normally appear on top of the input field. With floating labels, you can insert the label inside the input field, and make them float/animate when you click on the input field. The <label> elements must come after the <input> element, and the placeholder attribute is required for each <input> element (even though it is not shown):



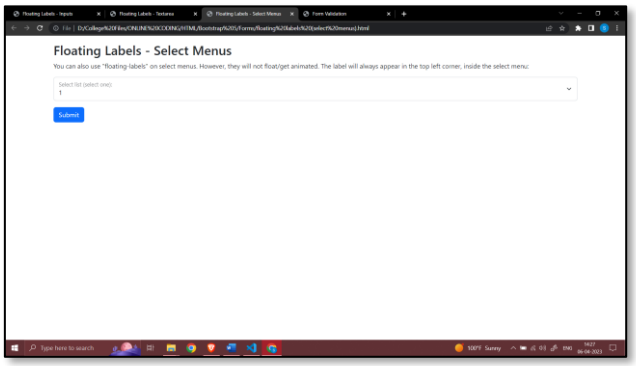
Textarea

It also works for textareas:



Select Menus

You can also use "floating-labels" on select menus. However, they will not float/get animated. The label will always appear in the top left corner, inside the select menu:



Form Validation

You can use different validation classes to provide valuable feedback to users. Add either **.was-validated** or **.needs-validation** to the <form> element, depending on whether you want to provide validation feedback before or after submitting the form. The input fields will have a green (valid) or red (invalid) border to indicate what's missing in the form. You can also add a **.valid-feedback** or **.invalid-feedback** message to tell the user explicitly what's missing, or needs to be done before submitting the form.