Agenda:

- What is a UI library ?
- What is Bootstrap ?
- Why Bootstrap is useful ?
- Getting started with bootstrap
- Bootstrap’s Grid System
- Components in Bootstrap

In this session we are going to learn about Bootstrap which is one of the famous User Interface Library (UI library), so let’s start the session by learnir what is a UI library:

What is a UI Library ?

A UI component library is a (usually) **robust set of ready-made UI components** such as buttons, inputs, dialogs, and so on. They serve as buildir blocks for layouts. Thanks to their modular nature, we can arrange components in many different ways to achieve unique effects.

One of the biggest perks of working with these libraries is that you get access to a very sizable **collection of ready-made UI elements**. These materia can greatly facilitate the work of your developers, allowing them to either directly apply them in a project, or modify them, achieving more of a custo feel.

A UI library, in the simplest terms, is a **collection of materials, or components, that you can readily use or modify to meet your needs**. You ca think of UI components as single elements available in that library, such as **buttons, input fields**, or **banners** made out of pure HTML and CSS.

So, now after getting the gist that what is a ui library, we can start learning Bootstrap so let’s start from the scratch:

What is Bootstrap ?

Bootstrap is an open-source CSS framework designed to come up with mobile-friendly, responsive front-end web development. It consists of enormou versatile and reusable pieces of code written in HTML, JavaScript, and CSS. As a framework, fundamentals are already placed for responsive we development, and developers simply need to position the code in a premeditated grid system. Therefore, while coming up with a new website application, Bootstrap is a boon because it waives off the burden of coding from scratch. You can efficiently blend its ready-made coding blocks, CS Less functionality, and cross-browser compatibility to save tedious hours of coding.

Why Bootstrap is useful ?

- **Easy Initiaion : T**he integration process of Bootstrap is easy on existing and newer websites. In your current CSS, you can incorporate diverse platforms, frameworks, elements of Bootstrap without any hassles.
- **Responsiveness :** Due to the extensive demand, having a mobile-responsive website has become a prerequisite, and this task is a cakewalk with Bootstrap by your side. It has a fluid grid layout, which adapts as per the screen resolution.
- **Highly Customizable :** You can find an array of templates in Bootstrap. However, if you find them unappealing, you can concoct your customization using the CSS file. Besides, if you cannot spare time to start everything from scratch, you can blend the customization with the existing code and enhance the functionality. All this can be undertaken under the customization page.
- **Cross-Browser Compatibility** : Every user is different, and so is their device. Along with the device, they use varied browsers too. Bootstrap is compatible with the latest versions of all modern browsers and platforms. One of the biggest advantages of using Bootstrap is compatibility across browsers. With Bootstrap by your side, heave a sigh of relief when it comes to displaying your landing page across multiple browsers.

Now after learning that what is bootstrap and why it is that widely used, let's straight dive into how we can use bootstrap in our project:

Getting started with Bootstrap:

We can add bootstrap into our project in the following ways described below:

1. **Using Bootstrap's CDN link :** In order to make a plain HTML file a Bootstrap template, just include the Bootstrap CSS and JS files using their CDN links. Also, you should include JavaScript files at the bottom of the page, right before the closing `</body>` tag to improve the performance of your web pages as it is shown below:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Basic Bootstrap Template</title>
    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/
        bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm
        3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">
</head>
<body>
    <h1>Hello, world!</h1>
    <!-- Bootstrap JS Bundle with Popper -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.
        bundle.min.js" integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YL
EuNSfAP+JcXn/tWtIaxVXM" crossorigin="anonymous"></script>
</body>
</html>
```

1. **By downloading the Bootstrap files** :

Alternatively, you can also download the Bootstrap's CSS and JS files from their official website and include in your project. There are two versior available for download, **compiled Bootstrap** and **Bootstrap source** files.

Compiled download contains compiled and minified version of CSS and JavaScript files for faster and easier web development. The compiled versic also includes optional JavaScript dependencies such as Popper bundled with Bootstrap JS (`bootstrap.bundle.*`). Whereas, the source downloa contains original source files for all CSS and JavaScript, along with a local copy of the docs.

Download and unzip the compiled Bootstrap. Now if you look inside the folders you'll find it contains the compiled CSS and JS files (`bootstrap.*`), a well as the compiled and minified CSS and JS (`bootstrap.min.*`). Use the `bootstrap.min.css` and `bootstrap.bundle.min.js` files.

Using the minified version of CSS and JS files will improve the performance of your website and saves the precious bandwidth because of lesser HTT request and download size.

Now before diving into bootstrap's grid system, let's first learn about responsive web design, breakpoints and containers in bootstrap:

Responsive web design:

Responsive web design is a process of designing and building websites to provide better accessibility and optimal viewing experience to the user l optimizing it for different devices.

With the growing trend of smart phones and tablets, it has become almost unavoidable to ignore the optimization of sites for mobile devices. Responsiv web design is a preferable alternative and an efficient way to target a wide range of devices with much less efforts.

Responsive layouts automatically adjust and adapts to any device screen size, whether it is a desktop, a laptop, a tablet, or a mobile phone.

Consider the picture shown below:

Breakpoints in bootstrap:

Breakpoints are customizable widths that determine how your responsive layout behaves across device or viewport sizes in Bootstrap.

- **Breakpoints are the building blocks of responsive design.** Use them to control when your layout can be adapted at a particular viewport or device size.
- **Use media queries to architect your CSS by breakpoint.** Media queries are a feature of CSS that allow you to conditionally apply styles based on a set of browser and operating system parameters. We most commonly use `min-width` in our media queries.
- **Mobile first, responsive design is the goal.** Bootstrap's CSS aims to apply the bare minimum of styles to make a layout work at the smallest breakpoint, and then layers on styles to adjust that design for larger devices. This optimizes your CSS, improves rendering time, and provides a great experience for your visitors.

Available breakpoints in bootstrap :

Breakpoint	Class infix	Dimensions
X-Small	none	<576px
Small	sm	≥576px
Medium	md	≥768px
Large	lg	≥992px
Extra Large	xl	≥1200px
Extra Extra Large	xxl	≥1400px

Containers in bootstrap:

Bootstrap requires a containing element to wrap elements and contain its grid system (more on the grid system next). Bootstrap's container classes we created specifically for this purpose.

Bootstrap includes three different container types:

Fixed Containers : A fixed container is a (responsive) fixed width container. As you resize your browser, its width remains intact, until it passes a certa breakpoint (as specified by you), at which time it will resize to the new width for that break point.

```
<div class="container">
...
</div>
```

Fluid Containers : A fluid container spans the full width of the viewport. It will expand and contract fluidly as you resize the browser. This is in contrast the fixed width container which will appear to "jump" to the new size as you pass a given break point.

```
<div class="container-fluid">
...
</div>
```

Responsive Containers : esponsive containers allow you to specify a class that is 100% wide until the specified breakpoint is reached. You specify tl breakpoint by appending it to the container's class (for example `container-lg` to make it 100% wide up until the large breakpoint). The breakpoin can be any of `sm` , `md` , `lg` , `xl` , and `xxl` .

```
<div class="container-lg">
  ...
  100% wide up until the large breakpoint
  ...
</div>
```

Bootstrap's Grid System:

Grid systems in bootstrap enable you to create advanced layouts using rows and columns. The Bootstrap grid system can have up to 12 columns, ar you can specify how these columns scale for different viewport sizes.

Working of Bootstrap Grid System :

Grid systems are used for creating page layouts through a series of rows and columns that house your content. Here's how the Bootstrap grid syste works –

- Rows must be placed within a **.container** class for proper alignment and padding.
- Use rows to create horizontal groups of columns.
- Content should be placed within the columns, and only columns may be the immediate children of rows.
- Predefined grid classes like **.row and .col-xs-4** are available for quickly making grid layouts. LESS mixins can also be used for more semantic layouts.
- Columns create gutters (gaps between column content) via padding. That padding is offset in rows for the first and the last column via negative margin on **.rows**.
- Grid columns are created by specifying the number of twelve available columns you wish to span. For example, three equal columns would use three **.col-xs-4**.

Consider the example shown below:

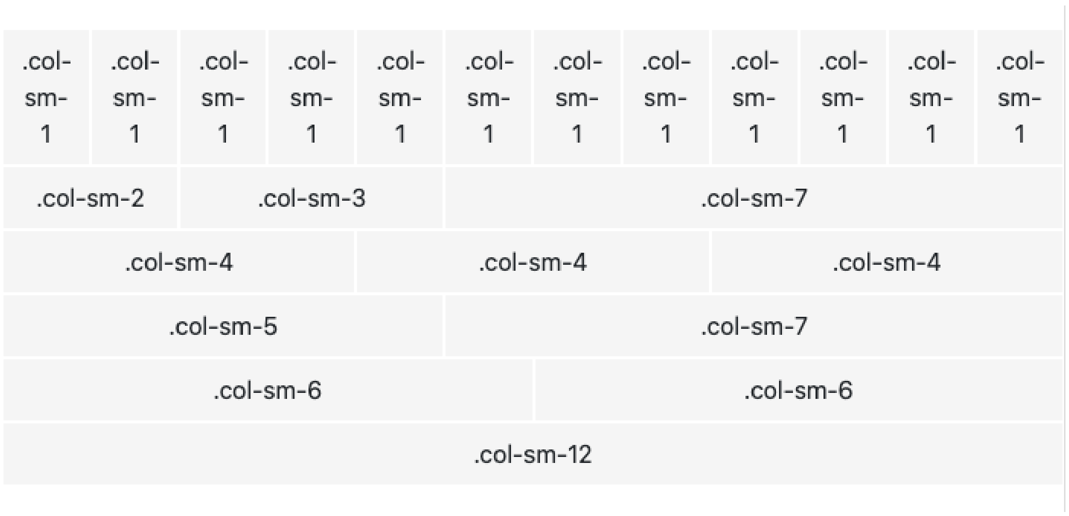
```
<!DOCTYPE html>
<title>My Example</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-

<style>
body {
padding-top: 1em;
}
</style> <div class="container-fluid">

<!-- Styles (so that we can see the grid) -->
<style>
.bs-example  div[class^="col"] {
    border: 1px solid white;
    background: #f5f5f5;
    text-align: center;
    padding-top: 8px;
    padding-bottom: 8px;
    }
</style>

<div class="bs-example">
    <!-- Bootstrap Grid -->
    <div class="row">
      <div class="col-sm-1">.col-sm-1</div>
      <div class="col-sm-1">.col-sm-1</div>
      <div class="col-sm-1">.col-sm-1</div>
      <div class="col-sm-1">.col-sm-1</div>
      <div class="col-sm-1">.col-sm-1</div>
      <div class="col-sm-1">.col-sm-1</div>
```

```
   <div class="col-sm-1">.col-sm-1</div>
   <div class="col-sm-1">.col-sm-1</div>
   <div class="col-sm-1">.col-sm-1</div>
   <div class="col-sm-1">.col-sm-1</div>
   <div class="col-sm-1">.col-sm-1</div>
   <div class="col-sm-1">.col-sm-1</div>
  </div>
  <div class="row">
   <div class="col-sm-2">.col-sm-2</div>
   <div class="col-sm-3">.col-sm-3</div>
   <div class="col-sm-7">.col-sm-7</div>
  </div>
  <div class="row">
   <div class="col-sm-4">.col-sm-4</div>
   <div class="col-sm-4">.col-sm-4</div>
   <div class="col-sm-4">.col-sm-4</div>
  </div>
  <div class="row">
   <div class="col-sm-5">.col-sm-5</div>
   <div class="col-sm-7">.col-sm-7</div>
  </div>
  <div class="row">
   <div class="col-sm-6">.col-sm-6</div>
   <div class="col-sm-6">.col-sm-6</div>
  </div>
  <div class="row">
   <div class="col-sm-12">.col-sm-12</div>
  </div>
</div>

</div>

<!-- Bundled -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/js/bootstrap.bundle.min.js" integrity="sha384-gtEjrD/SeCt
</script>
```

In the above example, we have created a grid which on smaller viewports, each grid cell will be stacked on top of each other. On larger viewports, the fu
grid comes into effect as intended.

The numbers at the end of each class name represent the number of columns that the column spans. So `.col-sm-1` spans one column and `.col`
`sm-8` spans eight. The `sm` means that the colspan applies to small devices and everything above.

Output on larger viewports :



```
.col-sm-1 .col-sm-1 .col-sm-1 .col-sm-1 .col-sm-1 .col-sm-1 .col-sm-1 .col-sm-1 .col-sm-1 .col-sm-1 .col-sm-1 .col-sm-1
.col-sm-2 .col-sm-3 .col-sm-7
.col-sm-4 .col-sm-4 .col-sm-4
.col-sm-5 .col-sm-7
.col-sm-6 .col-sm-6
.col-sm-12
```

Components in Bootstrap:

Let’s see different main components that bootstrap provides one by one, starting with the buttons component:

Button Component:

We can use Bootstrap’s custom button styles for actions in forms, dialogs, and more with support for multiple sizes, states, and more.

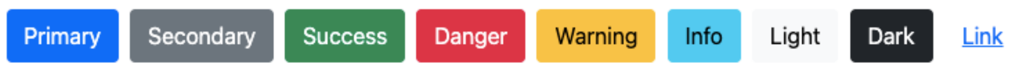
Bootstrap includes several predefined button styles, each serving its own semantic purpose, with a few extras thrown in for more control.

To style a button, use Bootstrap's `.btn` class, followed by the desired style. For example, `class="btn btn-primary"` results in a primary button.

Consider the examples shown below:

```
<button type="button" class="btn btn-primary">Primary</button>
<button type="button" class="btn btn-secondary">Secondary</button>
<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-info">Info</button>
<button type="button" class="btn btn-light">Light</button>
<button type="button" class="btn btn-dark">Dark</button>

<button type="button" class="btn btn-link">Link</button>
```

Output:



Button group:

We can group a series of buttons either on a single line or stack them in a vertical column.

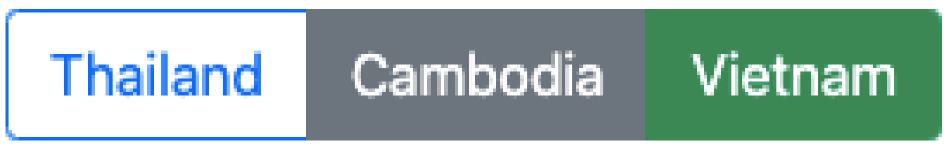
In order to make a `<div>` , a button group component we have add a class `.btn-group` to the div. Inside this `<div>` now we can add differe buttons and we can give any class to these single buttons and can style them as per our requirements. For example, we can add classes like `.btr outline-primary` to make the button outlined variant with the primary color to the border, we can add any color theme class we have discussed in tl button component, we can also add sizing classes to a button like `.btn-lg` , `.btn-sm` to make it bigger or smaller etc.

Also we can add sizing classes to the button group itself like `.btn-group-lg` , `.btn-group-sm` to make it bigger or smaller.

Consider the example shown below:

```
<div class="btn-group" role="group">
    <button type="button" class="btn btn-outline-primary">Thailand</button>
    <button type="button" class="btn btn-secondary">Cambodia</button>
    <button type="button" class="btn btn-success">Vietnam</button>
</div>
```

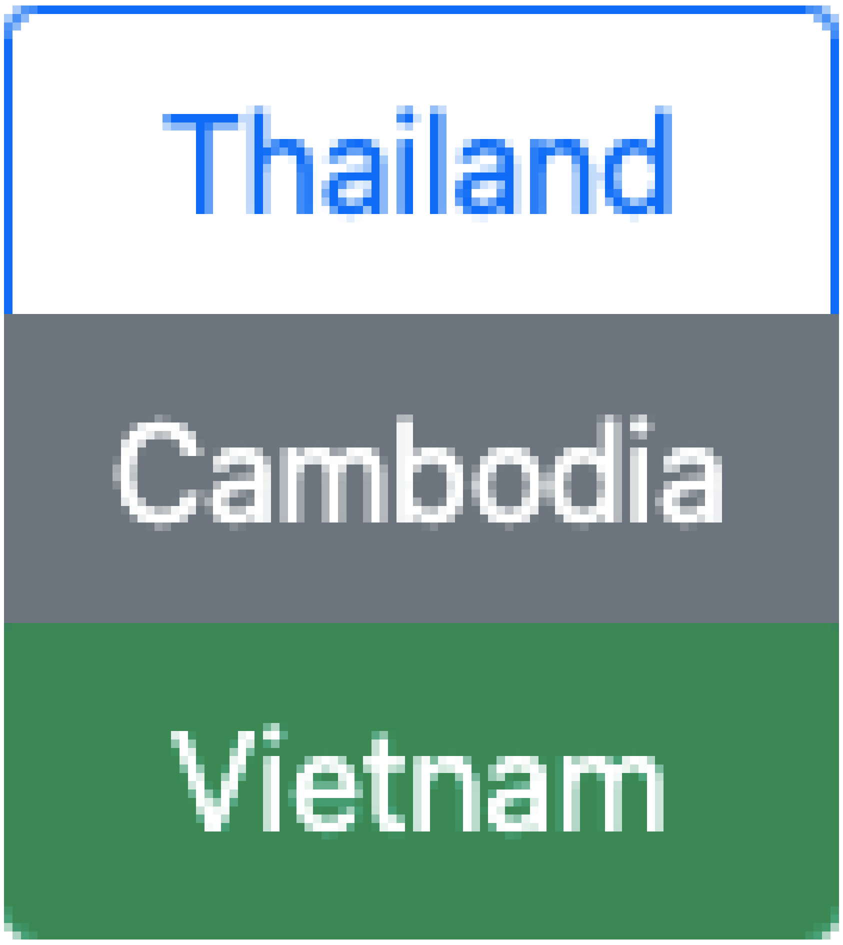
Output:



In order to stack them vertically we just need to add the class .btn-group-vertical instead of .btn-group. Consider the example shown below :

```
<div class="btn-group-vertical" role="group">
    <button type="button" class="btn btn-outline-primary">Thailand</button>
    <button type="button" class="btn btn-secondary">Cambodia</button>
    <button type="button" class="btn btn-success">Vietnam</button>
</div>
```

Output:



Cards:

Bootstrap’s cards provide a flexible and extensible content container with multiple variants and options.

A **card** is a flexible and extensible content container. It includes options for headers and footers, a wide variety of content, contextual background color and powerful display options.

Cards are built with as little markup and styles as possible, but still manage to deliver a ton of control and customization. Built with flexbox, they off easy alignment and mix well with other Bootstrap components. They have no `margin` by default, so use spacing utilities as needed.

Consider the example of a basic card as shown below:

```
<div class="card" style="width: 18rem;">
  <img src="..." class="card-img-top" alt="...">
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">Some quick example text to build on the card
                title and make up the bulk of the card's content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

Output:

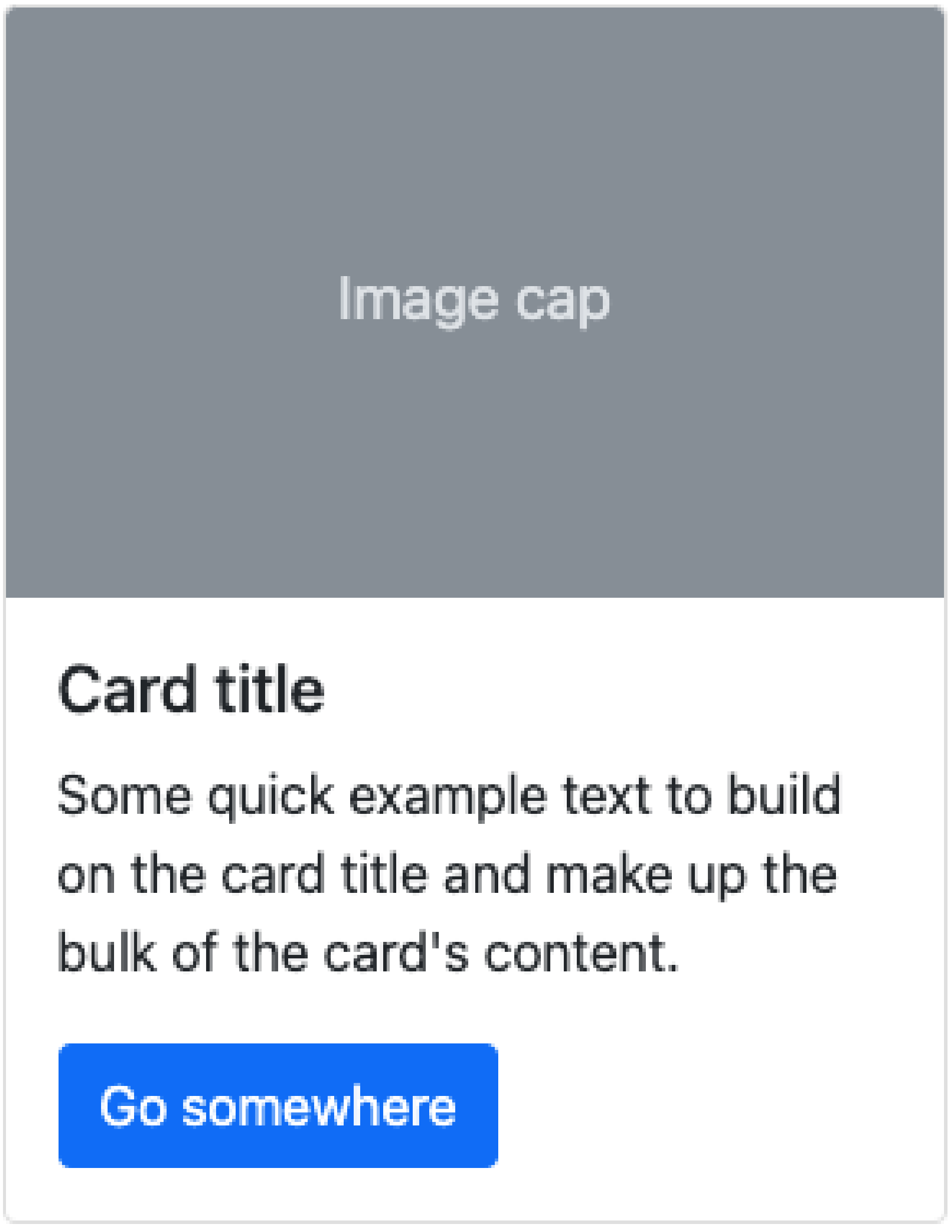


Image cap
Card title
Some quick example text to build on the card title and make up the bulk of the card's content.
Go somewhere

Dropdowns:

Dropdowns allow us to toggle contextual overlays for displaying lists of links and more with the Bootstrap dropdown plugin.They're made interactive wi the included Bootstrap dropdown JavaScript plugin. They're toggled by clicking, not by hovering; this is an international design decision.

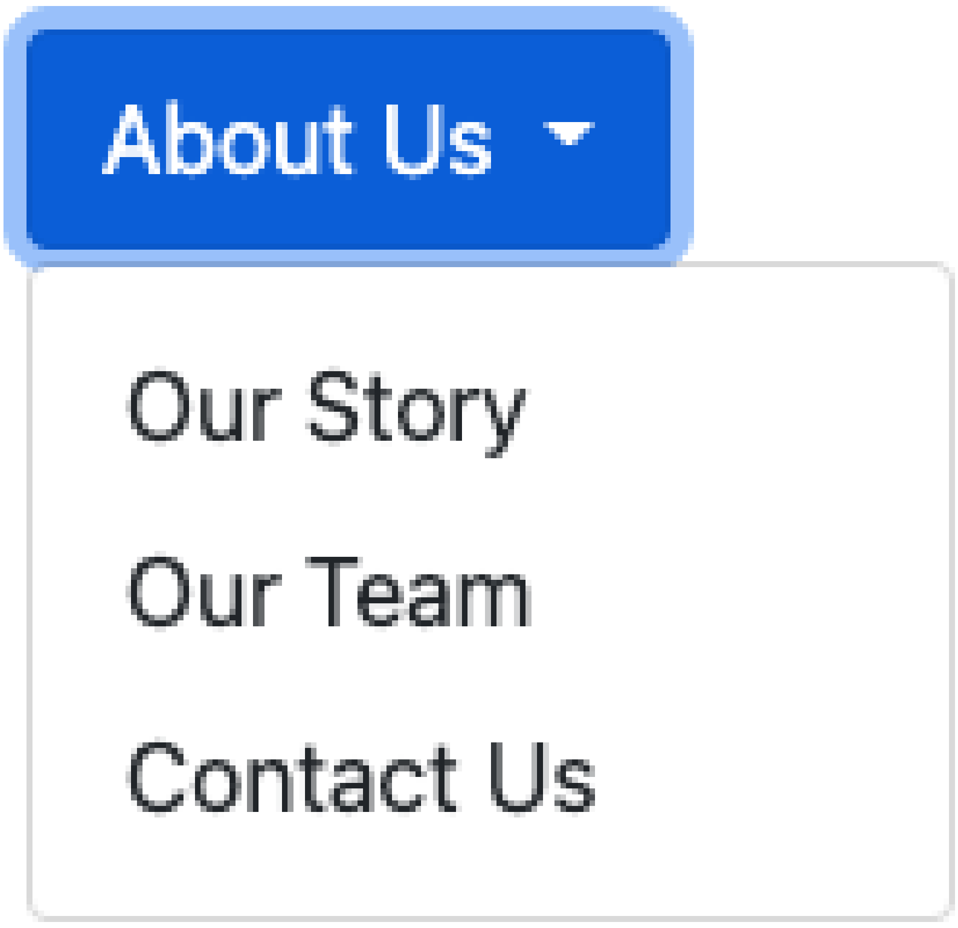
Wrap the dropdown's toggle (your button or link) and the dropdown menu within `.dropdown` , or another element that declares `positior relative;` . Dropdowns can be triggered from `<a>` or `<button>` elements to better fit your potential needs. The examples shown here us semantic `<ul>` elements where appropriate, but custom markup is supported.

Consider the example shown below:

```
<div class="dropdown">
<button class="btn btn-primary dropdown-toggle" type="button" id="about-us"
 data-bs-toggle="dropdown" aria-expanded="false">
About Us
</button>
<ul class="dropdown-menu" aria-labelledby="about-us">
<li><a class="dropdown-item" href="#">Our Story</a></li>
<li><a class="dropdown-item" href="#">Our Team</a></li>
<li><a class="dropdown-item" href="#">Contact Us</a></li>
```

```
</ul>
</div>
```

Output:



List Group:

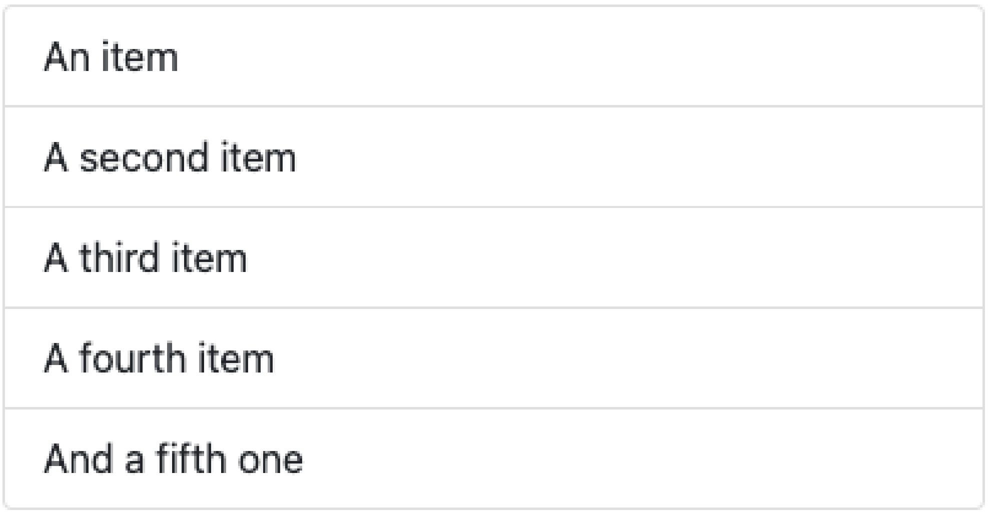
List groups are a flexible and powerful component for displaying a series of content. Modify and extend them to support just about any content within.

To create a default list group, apply the `.list-group` class to the `<ul>` tag, and the `.list-group-item` to each `<li>` tag.

Consider the example shown below:

```
<ul class="list-group">
  <li class="list-group-item">An item</li>
  <li class="list-group-item">A second item</li>
  <li class="list-group-item">A third item</li>
  <li class="list-group-item">A fourth item</li>
  <li class="list-group-item">And a fifth one</li>
</ul>
```

Output:


An item

A second item

A third item

A fourth item

And a fifth one

Modal:

We can use Bootstrap's JavaScript modal plugin to add dialogs to our site for lightboxes, user notifications, or completely custom content.

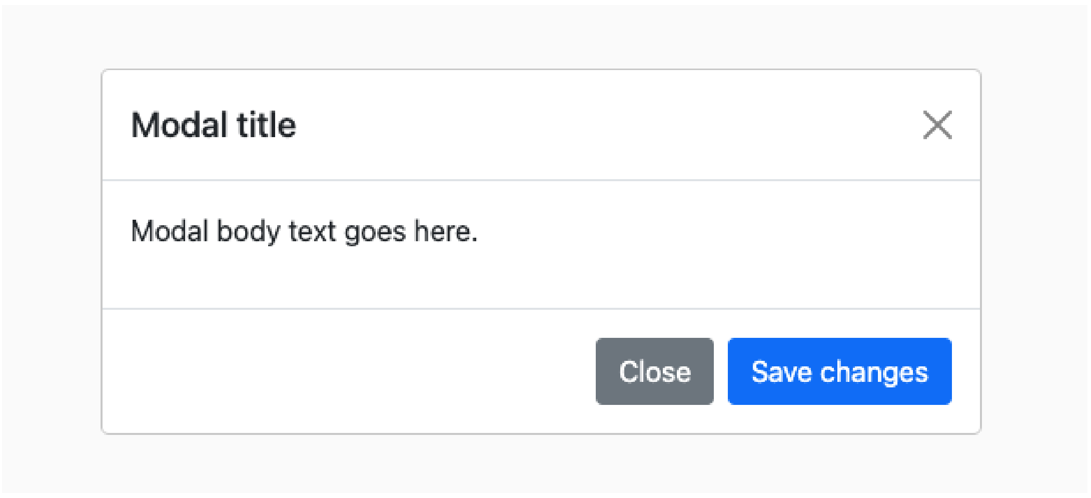
Working of a Modal:

- Modals are built with HTML, CSS, and JavaScript. They're positioned over everything else in the document and remove scroll from the `<body>` so that modal content scrolls instead.
- Clicking on the modal "backdrop" will automatically close the modal.
- Bootstrap only supports one modal window at a time. Nested modals aren't supported as we believe them to be poor user experiences.
- Modals use `position: fixed`, which can sometimes be a bit particular about its rendering. Whenever possible, place your modal HTML in a top-level position to avoid potential interference from other elements. You'll likely run into issues when nesting a `.modal` within another fixed element.

Consider the example shown below:

```
<div class="modal" tabindex="-1">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">Modal title</h5>
        <button type="button" class="btn-close"
                    data-bs-dismiss="modal" aria-label="Close"></button>
      </div>
      <div class="modal-body">
        <p>Modal body text goes here.</p>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary"
                 data-bs-dismiss="modal">Close</button>
        <button type="button" class="btn btn-primary">Save changes</button>
      </div>
    </div>
  </div>
</div>
```

Output:



Progress Bars:

We can provide graphical feedback on the progress of a process with Bootstrap's progress bars.

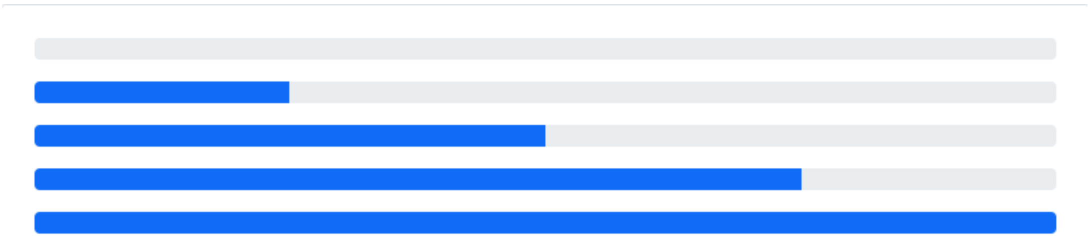
Working of a Progress Bar:

- We use the `.progress` as a wrapper to indicate the max value of the progress bar.
- We use the inner `.progress-bar` to indicate the progress so far.
- The `.progress-bar` requires an inline style, utility class, or custom CSS to set their width.
- The `.progress-bar` also requires some `role` and `aria` attributes to make it accessible.

Consider the example shown below:

```
<div class="progress">
  <div class="progress-bar" role="progressbar" aria-valuenow="0"
         aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar" role="progressbar" style="width: 25%"
         aria-valuenow="25" aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar" role="progressbar" style="width: 50%"
         aria-valuenow="50" aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar" role="progressbar" style="width: 75%"
         aria-valuenow="75" aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar" role="progressbar" style="width: 100%"
         aria-valuenow="100" aria-valuemin="0" aria-valuemax="100"></div>
</div>
```

Output:



Navbar :

Using bootstrap's Navbar component, we can add a fully-functional, responsive navbar to our website with just the minimal code.

Let's see how it works:

- Navbars require a wrapping `.navbar` with `.navbar-expand{-sm|-md|-lg|-xl|-xxl}` for responsive collapsing and color scheme classes like `.navbar-light` , `.navbar-dark` etc. and we can add the background color using `.bg-primary` , `.bg-secondary` etc.
- Navbars and their contents are fluid by default. Change the container to limit their horizontal width in different ways.
- Use our spacing and flex utility classes (discussed later in this session) for controlling spacing and alignment within navbars.
- Navbars are responsive by default, but you can easily modify them to change that. Responsive behavior depends on our Collapse JavaScript plugin.
- Ensure accessibility by using a `<nav>` element or, if using a more generic element such as a `<div>` , add a `role="navigation"` to every navbar to explicitly identify it as a landmark region for users of assistive technologies.

Navbars come with built-in support for a handful of sub-components. Choose from the following as needed:

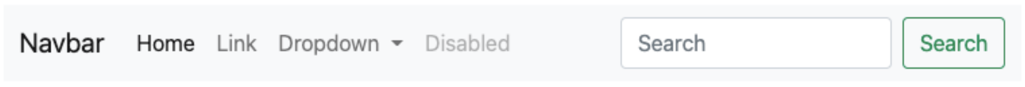
- `.navbar-brand` for your company, product, or project name.
- `.navbar-nav` for a full-height and lightweight navigation (including support for dropdowns).
- `.navbar-toggler` for use with our collapse plugin and other navigation toggling behaviors.
- Flex and spacing utilities for any form controls and actions.
- `.navbar-text` for adding vertically centered strings of text.
- `.collapse.navbar-collapse` for grouping and hiding navbar contents by a parent breakpoint.
- Add an optional `.navbar-scroll` to set a `max-height` and scroll expanded navbar content.

Consider the example shown below:

```html
<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <div class="container-fluid">
    <a class="navbar-brand" href="#">Navbar</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
         data-bs-target="#navbarSupportedContent"
            aria-controls="navbarSupportedContent" aria-expanded="false"
                aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarSupportedContent">
      <ul class="navbar-nav me-auto mb-2 mb-lg-0">
        <li class="nav-item">
          <a class="nav-link active" aria-current="page" href="#">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Link</a>
        </li>
        <li class="nav-item dropdown">
          <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown"
                    role="button" data-bs-toggle="dropdown" aria-expanded="false">
            Dropdown
          </a>
          <ul class="dropdown-menu" aria-labelledby="navbarDropdown">
            <li><a class="dropdown-item" href="#">Action</a></li>
            <li><a class="dropdown-item" href="#">Another action</a></li>
            <li><hr class="dropdown-divider"></li>
            <li><a class="dropdown-item" href="#">Something else here</a></li>
          </ul>
        </li>
        <li class="nav-item">
          <a class="nav-link disabled" href="#" tabindex="-1"
                    aria-disabled="true">Disabled</a>
        </li>
      </ul>
      <form class="d-flex">
        <input class="form-control me-2" type="search" placeholder="Search"
```

```
              aria-label="Search">
        <button class="btn btn-outline-success" type="submit">Search</button>
      </form>
    </div>
  </div>
</nav>
```

Output:



Carousel :

We can add a slideshow component for cycling through elements like images or slides of text using bootstrap's carousel component.

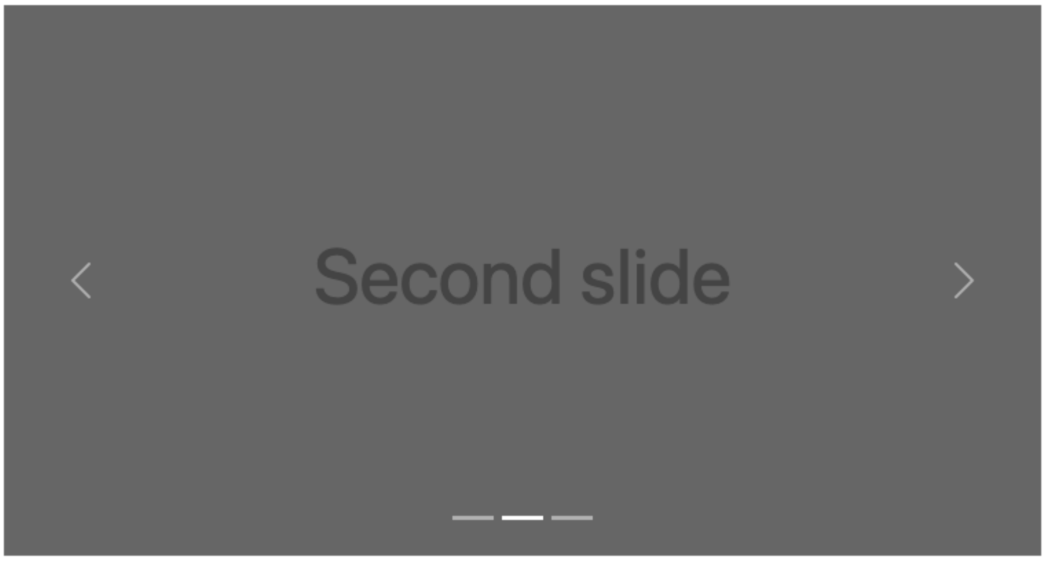
Carousels don't automatically normalize slide dimensions. As such, you may need to use additional utilities or custom styles to appropriately size conter
While carousels support previous/next controls and indicators, they're not explicitly required. Add and customize as you see fit.

The `.active` class needs to be added to one of the slides otherwise the carousel will not be visible. Also be sure to set a unique `id` c
the `.carousel` for optional controls, especially if you're using multiple carousels on a single page. Control and indicator elements must have a `data`
`bs-target` attribute (or `href` for links) that matches the `id` of the `.carousel` element.

Consider the example given below :

```
<div id="carouselExampleIndicators" class="carousel slide"
        data-bs-ride="carousel">
  <div class="carousel-indicators">
        <button type="button" data-bs-target="#carouselExampleIndicators"
            data-bs-slide-to="0" class="active" aria-current="true"
            aria-label="Slide 1"></button>
        <button type="button" data-bs-target="#carouselExampleIndicators"
            data-bs-slide-to="1" aria-label="Slide 2"></button>
        <button type="button" data-bs-target="#carouselExampleIndicators"
            data-bs-slide-to="2" aria-label="Slide 3"></button>
  </div>
  <div class="carousel-inner">
    <div class="carousel-item active">
      <img src="..." class="d-block w-100" alt="...">
    </div>
    <div class="carousel-item">
      <img src="..." class="d-block w-100" alt="...">
    </div>
    <div class="carousel-item">
      <img src="..." class="d-block w-100" alt="...">
    </div>
  </div>
  <button class="carousel-control-prev" type="button"
        data-bs-target="#carouselExampleIndicators" data-bs-slide="prev">
    <span class="carousel-control-prev-icon" aria-hidden="true"></span>
    <span class="visually-hidden">Previous</span>
  </button>
  <button class="carousel-control-next" type="button"
        data-bs-target="#carouselExampleIndicators" data-bs-slide="next">
    <span class="carousel-control-next-icon" aria-hidden="true"></span>
    <span class="visually-hidden">Next</span>
  </button>
</div>
```

Output :



So, these all are some important components that bootstrap provides to us, there are other components which you can ref through https://getbootstrap.com/docs/5.0/components .

Interview Questions

> What are the advantages of Bootstrap?

The following are some advantages of Bootstrap:

- Bootstrap is simple to use and anyone with a basic understanding of HTML and CSS can get started.
- Features that adapt to phones, tablets, and desktops: Bootstrap's responsive CSS adapts to phones, tablets, and desktops.
- A mobile-first strategy: Mobile-first styles are built into the Bootstrap framework.
- Bootstrap 4 is compatible with all modern browsers, including Chrome, Firefox, Internet Explorer 10+, Edge, Safari, and Opera

> What is a Bootstrap Container, and how does it work?

A bootstrap container is a handy class that generates a central region on the page where we can put our site content. The bootstrap .container has th advantage of being responsive and containing all of our other HTML code. Containers are used to pad the content within them, and there are two type of containers:

- The .container class creates a fixed-width container that is responsive.
- The .container-fluid class creates a full-width container that spans the entire viewport width.

> What do you know about the Bootstrap Grid System?

The Bootstrap Grid System is a mobile-first, responsive grid system that scales up to 12 columns as the device or viewport size grows. Predefine classes for quick layout options and powerful mix-ins for creating successful semantic layouts are included in the system.

There are five classes in the Bootstrap 4 grid system:

- .col- for extra small devices, whose screen width is less than 576px.
- .col-sm- small devices, whose screen width is equal to or greater than 576px.
- .col-md- medium devices, whose screen width is equal to or greater than 768px.
- .col-lg- large devices, whose screen width is equal to or greater than 992px.
- .col-xl- extra large devices, whose screen width is equal to or greater than 1200px.

The classes listed above can be combined to build layouts that are more dynamic and adaptable.

Thank You !

Thank You !