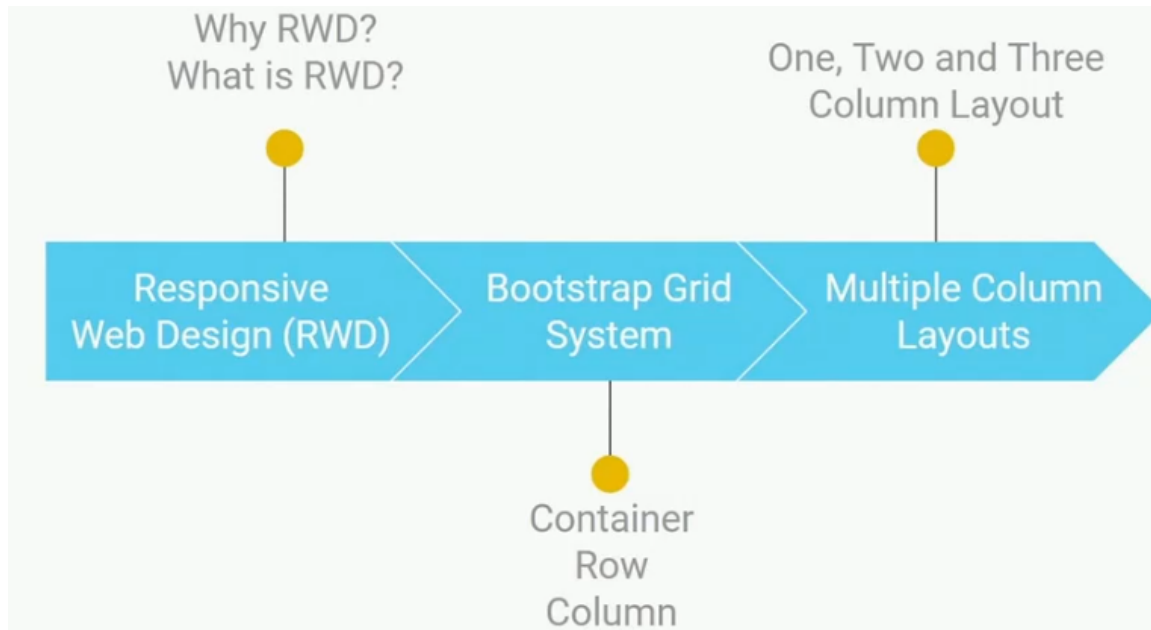


1. Intro to Responsive Web Design

In [1]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd1.png")
```

Out[1]:



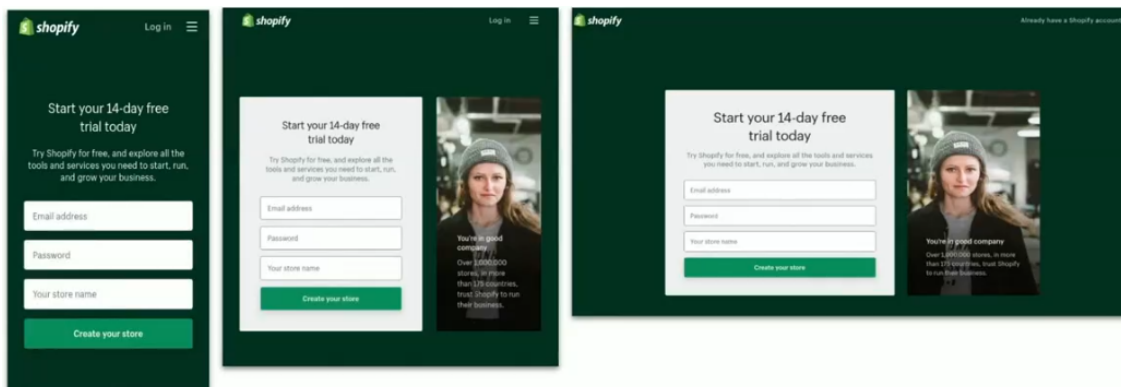
What is RWD?

- it is an approach to make web pages give best user experience in all the devices.

In [2]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd2.png")
```

Out[2]:



Why RWD ?

Ex: Amazon.in

How to build responsive websites ?

- using bootstrap grid system : it is a collection of Reusable Code snippets to create Responsive Layouts

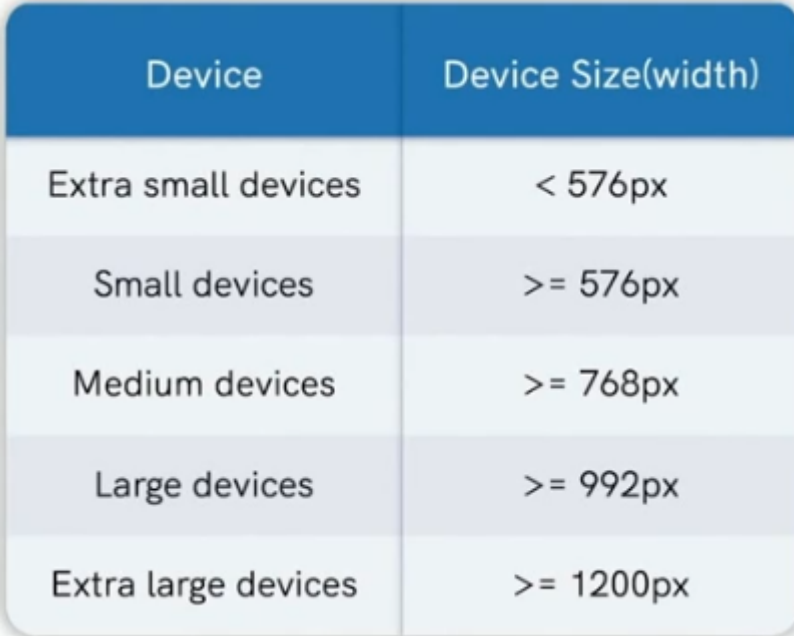
Five responsive tiers

- To make websites responsive categories devices into 5 responsive tiers these are known as Responsive Breakpoints

In [3]:

```
from IPython.display import Image  
Image("E:/code/frontend/img/rwd3.png")
```

Out[3]:



Device	Device Size(width)
Extra small devices	< 576px
Small devices	>= 576px
Medium devices	>= 768px
Large devices	>= 992px
Extra large devices	>= 1200px

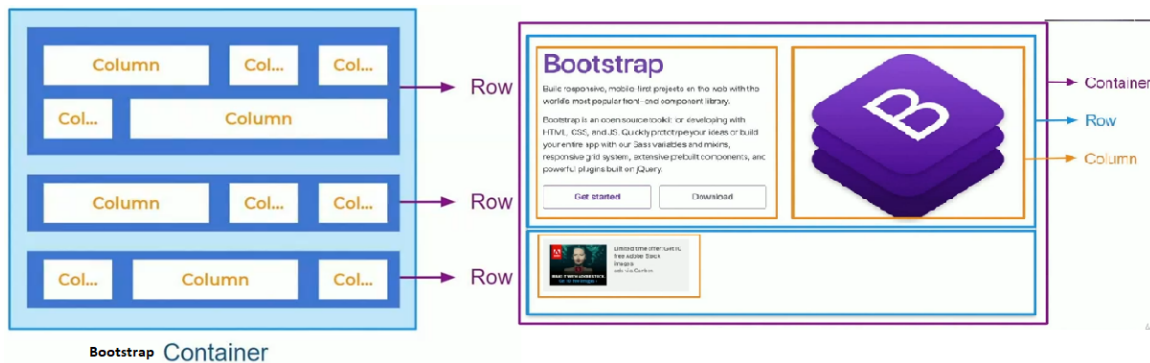
1. How Bootstrap Grid System works?

- It is made up of containers, rows, and columns.
- It uses a 12 column system for layouting. We can create up to 12 columns across the page.

In [4]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd4.png")
```

Out[4]:



1.1 Container

- The purpose of a container is to hold rows and columns.
- Here, the container is a div element with the Bootstrap class name container

1.2 Row

- The purpose of a row is to wrap all the columns.
- Here, the row is a div element with the Bootstrap class name row.

1.3 Column

- We should place the columns inside a row and the content inside a column.
- We can specify the number of columns our content should occupy in any device. The number of columns we specify should be a number in the range of 1 to 12.
- Here, the column is a div element with the Bootstrap class name col-12.

In [5]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd5.png")
```

Out[5]:

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" type="text/css" href="style3.css" />
</head>
<body>
<div class="container">
<div class="row">
<div class="col-12">
I'm your content inside the grid!
</div>
</div>
</div>
</body>
</html>
```

Note: If Bootstrap class name is col-12, it occupies the entire width available inside the row.

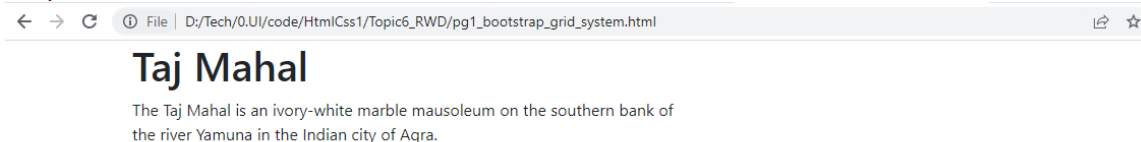
- The Bootstrap class names col-* indicates the number of columns you would like to use out of the possible 12 columns per row. For example, col-1, col-5, etc.
- Try out changing the number of columns in the below Code Playground and check the output.

In [6]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd6.png")
```

Out[6]:

```
<body>
<div class="container">
<div class="row">
<div class="col-6">
<h1 class="heading">Taj Mahal</h1>
<p>
The Taj Mahal is an ivory-white marble mausoleum on the southern
bank of the river Yamuna in the Indian city of Agra.
</p>
</div>
</div>
</body>
```



2. Creating Multiple Column Layouts

- The Layout in the below Code Playground is a Two Column Layout.
- Similarly try out multiple column layouts like One Column Layout, Three Column Layout, etc. in the below Code Playground.

In [7]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd7.png")
```

Out[7]:

```
<body>
<div class="container">
<div class="row">
<div class="col-6">
<h1 class="heading">Taj Mahal</h1>
<p>
The Taj Mahal is an ivory-white marble mausoleum on the southern
bank of the river Yamuna in the Indian city of Agra.
</p>
</div>
<div class="col-6">
<h1 class="heading">Mysore Palace</h1>
<p>
The Mysore Palace is a historical palace and the royal residence at
Mysore in the Indian State of Karnataka.
</p>
</div>
</div>
</div>
</body>
```

File | D:/Tech/0.UI/code/HtmlCss1/Topic6_RWD/pg2_multiple_col_layouts.html



Taj Mahal

The Taj Mahal is an ivory-white marble mausoleum on the southern bank of the river Yamuna in the Indian city of Agra.

Mysore Palace

The Mysore Palace is a historical palace and the royal residence in the Indian State of Karnataka.

2. Bootstrap Grid System | Part 1

1. Column Wrapping

- When we place more than 12 grid columns in a single row, the extra grid columns will wrap in a new line.
- Try out the different combinations of Bootstrap class names like col-4, col-4, col-6, col-6 and col-6, col-4, col-6, col-4, etc. in the below Code Playground.

In [8]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd8.png")
```

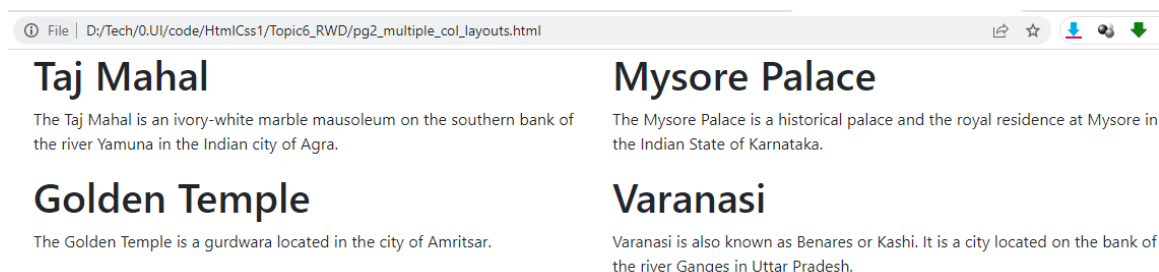
Out[8]:

```
<body>
  <div class="container">
    <div class="row">
      <div class="col-6">
        <h1 class="heading">Taj Mahal</h1>
        <p>
          The Taj Mahal is an ivory-white marble mausoleum on the southern
          bank of the river Yamuna in the Indian city of Agra.
        </p>
      </div>
      <div class="col-6">
        <h1 class="heading">Mysore Palace</h1>
        <p>
          The Mysore Palace is a historical palace and the royal residence at
          Mysore in the Indian State of Karnataka.
        </p>
      </div>
      <div class="col-6">
        <h1 class="heading">Golden Temple</h1>
        <p>The Golden Temple is a gurdwara located in the city of Amritsar.</p>
      </div>
      <div class="col-6">
        <h1 class="heading">Varanasi</h1>
        <p>Varanasi is also known as Benares or Kashi. It is a city located
          on the bank of the river Ganges in Uttar Pradesh.</p>
      </div>
    </div>
  </div>
</body>
```

In [9]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd9.png")
```

Out[9]:



2. The Layout at different Breakpoints

- Bootstrap provides different Bootstrap Grid Column class name prefixes for Five Responsive Tiers (Responsive Breakpoints).
- If we define the behaviour of the Bootstrap Grid Column in a particular device, similar behaviour is guaranteed in all devices with larger sizes.

In [10]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd12.png")
```

Out[10]:

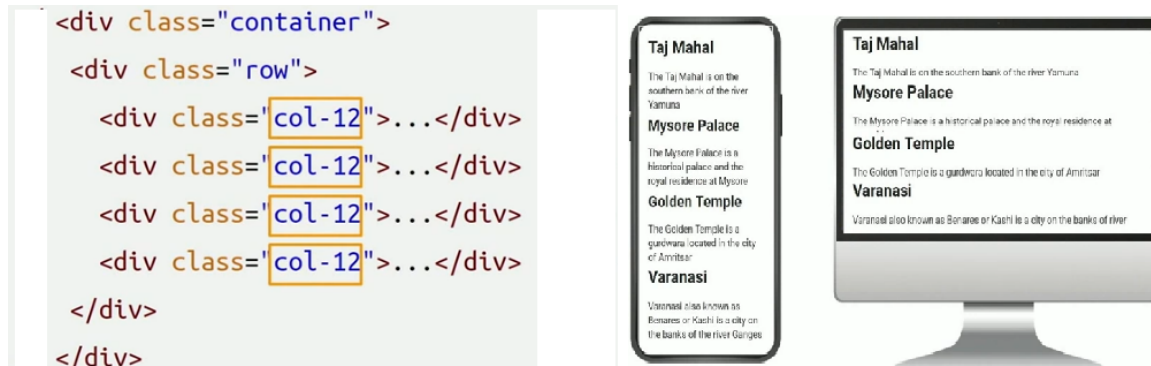
Device	Device Size (Width)	Class Name Prefix
Extra small devices	<576px	col-
Small devices	>=576px	col-sm-
Medium devices	>=768px	col-md-
Large devices	>=992px	col-lg-
Extra large devices	>=1200px	col-xl-

Large and extra large devices

In [11]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd11.png")
```

Out[11]:



2.1 Class names in combination

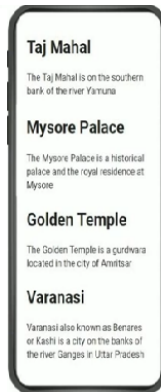
- We can use a combination of different Bootstrap class names for each Bootstrap Grid Column.
- Try out the combinations of different Bootstrap class names like col-lg-4, col-lg-3, col-lg-8, etc. in the below Code Playground.

In [12]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd10.png")
```

Out[12]:

```
<div class="container">
  <div class="row">
    <div class="col-12 col-lg-6">...</div>
    <div class="col-12 col-lg-6">...</div>
    <div class="col-12 col-lg-6">...</div>
    <div class="col-12 col-lg-6">...</div>
  </div>
</div>
```

**Note:**

- Bootstrap follows Mobile First Approach.
- First, design the Layout of a mobile version, and this will be adopted by devices with larger sizes.

3. Bootstrap Grid System | Part 2

In [13]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd13.png")
```

Out[13]:



1. CSS Box Properties

1.1 - margin

- We can get spacing between the two HTML elements with the CSS Box property margin.
- To get space only on one particular side, we use margin Variants.
 - margin-top
 - margin-right
 - margin-bottom
 - margin-left

2. Bootstrap Spacing Utilities

2.1 Margin :

- The asterisk (*) symbol can be any number in the range of 0 to 5. For example, m-5, mr-2, mb-3, etc.

2.1.1 Margin Values :

- The spacer is a variable and has a value of 16 pixels by default.
- For example,
 - $mb-3 = 1 * 16px = 16px$
 - $m-5 = 3 * 16px = 48px$

Note:

- Avoid using CSS margin-left and margin-right properties for Bootstrap Grid Columns. It disturbs the Bootstrap Grid System and gives unexpected results.

2.2 Padding

- The asterisk (*) symbol can be any number in the range of 0 to 5. For example, p-3, pr-1, pb-5, etc.

2.2.1 Padding Values

- The spacer is a variable and has a value of 16 pixels by default.
- For example,
 - $p-1 = 0.25 * 16px = 4px$
 - $pt-4 = 1.5 * 16px = 24px$

In [14]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd16.png")
```

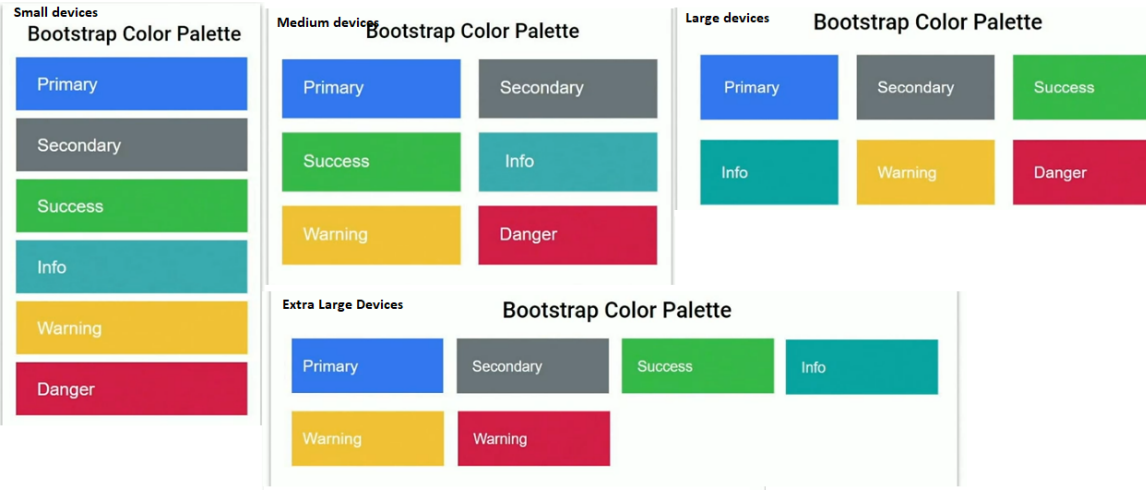
Out[14]:

CSS Margin property	Bootstrap class name	Size	Value	CSS Padding property	Bootstrap class name	Size	Value
margin	m-*	0	0	padding	p-*	0	0
margin-top	mt-*	1	0.25 * spacer	padding-top	pt-*	1	0.25 * spacer
margin-right	mr-*	2	0.5 * spacer	padding-right	pr-*	2	0.5 * spacer
margin-bottom	mb-*	3	1 * spacer	padding-bottom	pb-*	3	1 * spacer
margin-left	ml-*	4	1.5 * spacer	padding-left	pl-*	4	1.5 * spacer
		5	3 * spacer			5	3 * spacer

In [15]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd14.png")
```

Out[15]:



In [16]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd15.png")
```

Out[16]:

```
> <> pg4_color_pallate.html > html > body > div.container > div.row > div.col-
<div class="container">
  <div class="row">
    <div class="col-12 col-md-6 col-lg-4 col-xl-3 column-spacing">
      <div class="color-box bg-primary">
        <p class="color-name">Primary</p>
      </div>
    <div class="col-12 col-md-6 col-lg-4 col-xl-3 column-spacing">
      <div class="color-box bg-secondary">
        <p class="color-name">Secondary</p>
      </div>
    <div class="col-12 col-md-6 col-lg-4 col-xl-3 column-spacing">
      <div class="color-box bg-success">
        <p class="color-name">success</p>
      </div>
    <div class="col-12 col-md-6 col-lg-4 col-xl-3 column-spacing">
      <div class="color-box bg-info">
        <p class="color-name">info</p>
      </div>
    <div class="col-12 col-md-6 col-lg-4 col-xl-3 column-spacing">
      <div class="color-box bg-warning">
        <p class="color-name">warning</p>
      </div>
    <div class="col-12 col-md-6 col-lg-4 col-xl-3 column-spacing">
      <div class="color-box bg-danger">
        <p class="color-name">danger</p>
      </div>
    </div>
  </div>
</div>
```

```
{WD} > # pg4_style.css > ...
@import url('https://font-

.color-heading{
  text-align:center;
  font-size:24;
  font-weight:bold;
}

.color-box{
  padding:20px;
}

.color-name{
  color:white;
  font-size:15px;
}

.column-spacing{
  margin-bottom:20px;
}
```

4. Bootstrap Navbar

1. Bootstrap Components

1.1 Navbar

- A Navbar is a navigation header that is placed at the top of the page. With Bootstrap, a Navbar can extend or collapse, depending on the device size.

1.1.1 HTML Nav element

- The HTML nav element is a container element similar to the HTML div element. We use the HTML nav element to add a Navbar to our website.

1.1.2 Nav Items inside Navbar

1.1.3 Nav link

2. HTML Elements

- In general, HTML elements can be divided into two categories.
 - Block-level Elements
 - Inline Elements

2.1 Block-level Elements

- These elements always start in a new line and take up the full width available. So, an HTML Block-level element occupies the entire horizontal space of its parent element.
- For example, the HTML h1 element, p element, div element, etc.

2.2 Inline Elements

- These elements do not start in a new line and only take up as much width as necessary.
- For example, the HTML button element, img element, a element, etc.

3. CSS Box properties

3.1 Margin

- We can align HTML Block-level elements horizontally using CSS margin property.
- Apart from values that are specified in pixels, it also accepts auto keyword as a value.

Note:

- If we specify the CSS text-align property to the HTML Block-level element, it aligns the text or HTML Inline elements inside it.

3.1.1 Auto Value

- The child element will be horizontally centred inside the HTML container element.

3.1.2 Auto Value with Margin Variants

- Using auto as a value for the CSS margin-right property takes up all the available space, and the element gets aligned to the left.
- Using auto as a value for the CSS margin-left property takes up all the available space, and the element gets aligned to the right.

4. Bootstrap Utilities

4.1 Margin

- Apart from the numbers 0-5, the margin also has the below Bootstrap class names.
 - m-auto
 - ml-auto
 - mr-auto

Steps to create Navbar

- go to <https://getbootstrap.com/docs/4.6/getting-started/introduction/> (<https://getbootstrap.com/docs/4.6/getting-started/introduction/>) --> components --> Navbar --> select avoid list base approach
- remove the disabled tag down the code snippet
- add Logo (in place navbar anchor tag add image tag)
- align the navbar items to the right side
 - div, p and are block level elements so we cannot align the navbar to right using text-align property
 - we have to use margin-left:auto; margin:auto; margin-right:auto; or ml-auto, mr-auto, m-auto
- change the navbar background color bg-white

In [17]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd17.png")
```

Out[17]:

```

<body>
<nav class="navbar navbar-expand-lg navbar-light bg-white">
<a class="navbar-brand" href="#">

</a>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavAltMarkup" an:
</button>
<div class="collapse navbar-collapse" id="navbarNavAltMarkup">
<!--<div class="navbar-nav navbar-align-right">-->
<div class="navbar-nav ml-auto">
<a class="nav-link active" href="#">Home <span class="sr-only">(current)</span></a>
<a class="nav-link" href="#">About Me</a>
<a class="nav-link" href="#">Projects</a>
<a class="nav-link" href="#">Testimonials</a>
</div>
</div>
</nav>
</body>

```

```

css
RWD > # pg5_style.css > ...
@import url('https://for
.navbar-image{
width: 40px;
height: 40px;
padding: 5px;
}
.navbar-align-right{
margin-left: auto;
}

```

In [18]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd12.png")
```

Out[18]:

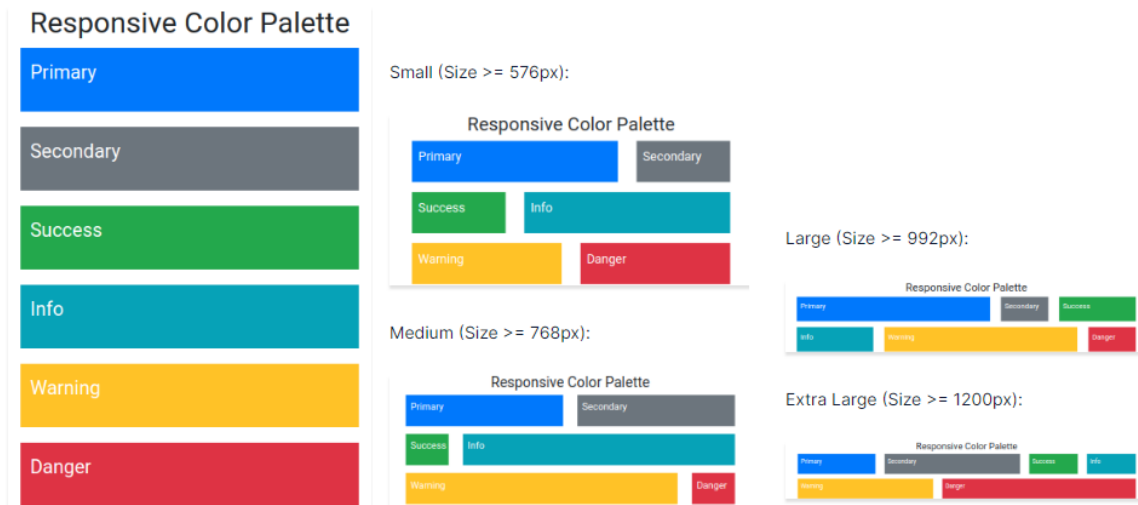
Device	Device Size (Width)	Class Name Prefix
Extra small devices	<576px	col-
Small devices	>=576px	col-sm-
Medium devices	>=768px	col-md-
Large devices	>=992px	col-lg-
Extra large devices	>=1200px	col-xl-

Task1

In [19]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd18.png")
```

Out[19]:



In [20]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd19.png")
```

Out[20]:

WD > <> Task1_color_palatte.html > html

```
<body>
<h1 class="color-heading">Bootstrap Color Palette</h1>
<div class="container">
  <div class="row">
    <div class="col-12 col-sm-8 col-md-6 col-lg-6 col-xl-2 mb-3">
    </div>
    <div class="col-12 col-sm-4 col-md-6 col-lg-2 col-xl-6 mb-3">
    </div>
    <div class="col-12 col-sm-4 col-md-3 col-lg-4 col-xl-2 mb-3">
    </div>
    <div class="col-12 col-sm-8 col-md-9 col-lg-4 col-xl-2 mb-3">
    </div>
    <div class="col-12 col-sm-6 col-md-9 col-lg-6 col-xl-4 mb-3">
    </div>
    <div class="col-12 col-sm-6 col-md-3 col-lg-2 col-xl-8 mb-3">
    </div>
  </div>
</div>
</body>
```

Task2

In [21]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd20.png")
```

Out[21]:



In [22]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd21.png")
```

Out[22]:

WD > <> Task2.html > html > body > div.container > div.row > div.col-12.col-sm-6

```

<body>
  <h1 class="color-heading">Bootstrap Color Palette</h1>
  <div class="container">
    <div class="row">
      <div class="col-12 col-sm-6 col-md-2 col-lg-3 col-xl-1 mb-3">..
    </div>
      <div class="col-12 col-sm-6 col-md-10 col-lg-2 col-xl-3 mb-3">
    </div>
      <div class="col-12 col-sm-6 col-md-10 col-lg-7 col-xl-7 mb-3">
    </div>
      <div class="col-12 col-sm-6 col-md-2 col-lg-7 col-xl-1 mb-3">..
    </div>
      <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-8 mb-3">..
    </div>
      <div class="col-12 col-sm-6 col-md-8 col-lg-2 col-xl-2 mb-3">..
    </div>
    </div>
  </div>
</body>

```

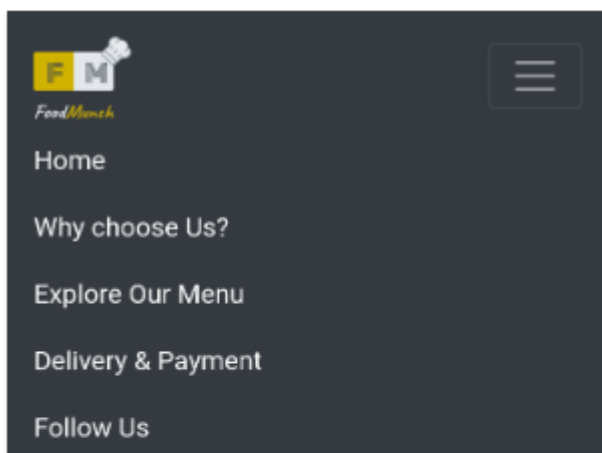
Task3

In [23]:

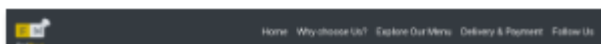
```
from IPython.display import Image  
Image("E:/code/frontend/img/rwd22.png")
```

Out[23]:

Extra Small (Size < 576px), Small (Size >= 576px)
and Medium (Size >= 768px):



Large (Size >= 992px) and Extra Large (Size >= 1200px):



In [24]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd27.png")
```

Out[24]:

```
<body>
<div class="bg-container">
<nav class="navbar navbar-expand-lg navbar-light bg-dark">
<a class="navbar-brand" href="#">

</a>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavAltMarkup"
aria-controls="navbarNavAltMarkup" aria-expanded="false" aria-label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>
<div class="collapse navbar-collapse" id="navbarNavAltMarkup">
<!--<div class="navbar-nav navbar-align-right">-->
<div class="navbar-nav ml-auto">
<a class="nav-link active myfonts" id="navitem" href="#">Home <span class="sr-only">(current)
<a class="nav-link" id="navitem1" href="#">Why choose Us?</a>
<a class="nav-link" id="navitem2" href="#">Explore Our Menu</a>
<a class="nav-link" id="navitem3" href="#">Delivery & Payments</a>
<a class="nav-link" id="navitem4" href="#">Follow Us</a>
</div>
</div>
</nav>
</div>
</body>

@import url('https://fonts.google
.bg-container{
background-image: black;
background-size: cover;
}
.navbar-image{
width: 90px;
height: 70px;
padding: 5px;
}
.navbar-align-right{
margin-left: auto;
}
#navitem{
color: white;
font-family: 'Roboto';
}
#navitem1{
color: white;
font-family: 'Roboto';
}
#navitem2{
color: white;
font-family: 'Roboto';
}
#navitem3{
color: white;
font-family: 'Roboto';
}
#navitem4{
color: white;
font-family: 'Roboto';
}
```

Topic 7. CSS Selectors and Inheritance

1. CSS Selectors

- The CSS Selectors are used to select the HTML Elements that we want to style.
- The different types of CSS Selectors are:
 - Simple Selectors
 - Class Selector
 - ID Selector
 - Type (tag name) Selector
 - Attribute Selector
 - Universal Selector
 - Pseudo-class
 - Compound Selectors
 - Complex Selectors and many more.

1.1 Class Selector

- The CSS Class Selector selects all the HTML elements that have a given CSS class selector as their class attribute value.
- It consists of a dot (.), followed by the class name of the HTML element.
- Here, the CSS class selector is .paragraph. So, it selects all the HTML elements that have an HTML attribute name class, and it's value paragraph.

Note:

- There can be more than one HTML element with the same class name in the HTML document.

1.2 ID Selector

- The CSS ID selector selects an HTML element based on its ID attribute value.
- It consists of a hash (#), followed by the ID of the HTML element.
- Here, the CSS ID selector is #populationParagraph. So, it selects the HTML element that has an HTML attribute name id and it's value populationParagraph

Note:

- There should be only one HTML element with a given ID in the entire HTML document. The HTML id attribute value doesn't need to have the prefix section as CCBP UI Kit is not used.

1.3 Type (tag name) Selector

- The CSS Type Selector selects all the HTML elements based on their tag names (h1, p, div, etc.)
- Here, the CSS Type selector is p. So, it selects all the HTML elements that have a tag name p.

In [25]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd23.png")
```

Out[25]:

```
html
<p class="paragraph">The population of India is around 138 crores.</p>
css      class selector
.paragraph {
  color: blue;
}

html
<p id="paragraph">The population of India is around 138 crores.</p>
css
#paragraph {
  color: blue;
}

html
<p>The population of India is around 138 crores.</p>
css      type selector
p {
  color: blue;
}
```

2. Most fundamental concepts of CSS

- In CSS, the styles that are applied to HTML elements depend on three fundamental concepts.
 - Inheritance
 - Specificity
 - Cascade

2.1 CSS Inheritance

- The mechanism through which the value of certain CSS properties is passed on from parent elements to child elements is called Inheritance.

2.1.1 Parent Element

- If the HTML elements are placed inside the other HTML element, then the outer HTML element is called the parent element of the HTML elements inside it.

In [26]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd25.png")
```

Out[26]:

```
<div>
  <h1>The seven wonders of the world</h1>
  <p>
    The <a href="https://en.wikipedia.org/wiki/Taj_Mahal">Taj Mahal</a>
    is one of the seven wonders of the world.
  </p>
</div>
```

From the above Code Snippet, we can say:

- The HTML div element is the parent element of the HTML h1 and p elements.
- The HTML p element is the parent element of the HTML a element.

2.1.2 Child Element

- An HTML element that is directly inside the parent element is called the child element of that parent element.
- From the above Code Snippet, we can say:
 - The HTML h1 and p elements are the child elements of the HTML div element.
 - The HTML a element is the child element of the HTML p element.

CSS properties can be categorized into two types:

- Inherited properties
- Non-inherited properties

2.1.3 Inherited Properties

- If the CSS properties applied to the parent element are inherited by the child elements, then they are called Inherited properties.
- Some of the CSS Inherited Properties are:-
 - Text related Properties : font-family, font-style, font-weight, text-align
 - List related Properties : list-style-type
 - color property and many more.

2.1.4 Non-inherited Properties

- If the CSS properties applied to the parent element are not inherited by the child elements, then they are called Non-inherited properties.
- Some of the CSS Non-inherited properties are:
 - CSS Box Properties : width, height, margin, padding, border-style, border-width, border-color, border-radius
 - CSS Background Properties : background-image, background-color, background-size
 - text-decoration and many more.

In [27]:

```
from IPython.display import Image
Image("E:/code/frontend/img/rwd24.png")
```

Out[27]:

```
<div class="country-container">
  <h2>About the information</h2>
  <p>The population of India is around 138 crores.</p>
  <p>The population of India is around 138 crores.</p>
</div>

.country-container{
  color: blue;
  text-align: center;
}
```

About the information

The population of India is around 138 crores.

The population of India is around 138 crores.

CSS Specificity & Cascade

1. Specificity

- Specificity is how browsers decide which CSS property values are the most relevant to an HTML element and, therefore, will be applied.
- The following list of CSS Selectors is in the lowest to highest order by specificity.
 - Type (tag name) Selector
 - Class Selector
 - ID Selector

1.1 Type Selector & Class Selector

- A Class Selector is more specific compared to Type (tag name) Selector as it selects only the HTML elements that have a specific class attribute value in the HTML document.

Note:

- It doesn't overwrite the entire CSS Ruleset but only overwrites the CSS properties that are the same.

1.2 Class Selector & ID Selector

- An ID Selector is more specific when compared to a Class Selector as we provide a unique ID within the HTML document and it selects only a single HTML Element.

ID Selector --> more specific than --> Class Selector -->more specific than --> Type Selector

2. Inline Styles

- The Inline styles are applied directly to an HTML element. They use the HTML style attribute, with CSS property values defined within it.
- Syntax: tag style = "property1: value1; property2: value2; ...">Content/tag<
- A HTML style attribute value can consist of one or more CSS property values.

Note:

- Inline Styles have the highest specificity. They overwrite any other styles specified using CSS Selectors.
- Using Inline Styles is not recommended because
 - Inline Styles are not reusable.
 - Writing HTML and CSS separately increases code readability.

3. CSS Cascade

- The source order of CSS Rulesets matters. When two CSS Rulesets have equal specificity, the one that comes last in the CSS is applied.

Note:

- The styles that apply to the HTML Elements are not determined by the order the classes defined in the HTML class attribute, but instead the order in which they appear in the CSS.

3.1 The !important exception

- It is a special piece of CSS used to make a particular CSS property and value the most specific thing, irrespective of source order and specificity.
- The only way to override a !important property value is to include another !important property value. The added property value should either come later in the order or should be of higher specificity.

In [28]:

```
from IPython.display import Image  
Image("E:/code/frontend/img/rwd26.png")
```

Out[28]:

```
<<body>  
<<<h1 class="style-2 style-1">About India(blue color)</h1>  
<<</body>  
  
.style-1 {  
  color: green !important;  
}  
h1 {  
  color: orange !important;  
}  
.style-2 {  
  color: blue !important;  
}
```

← → ↻ ⓘ File | D:/Tech/0.UI/code/HtmlCss1/Topic6_7_F

About India(blue color)