National University of Computer & Emerging Sciences,
Karachi Computer Science Department

Course Code: SE-3003	Course : Web Engineering Lab

Spring 2023, Lab Manual – 05

LLO 01: Design and Implement a simple web application.

Contents:

- Intro to Web Engineering
- Technologies
- Tools
- Bootstrap 5 Form and its types
- Flex and its types with examples
- Slider using carousel classes

Introduction to Web Engineering

Web Engineering is the application of systematic and quantifiable approaches (concepts methods, techniques tools) to cost - effective requirements analysis, design, implementation, testing, operation, and maintenance of **high quality Web applications.**

Technologies to be studied

- HTML
- CSS
- JavaScript
- Bootstrap
- JQuery
- PHP
- MySQL [Database]
- Laravel [PHP FRAMEWORK]

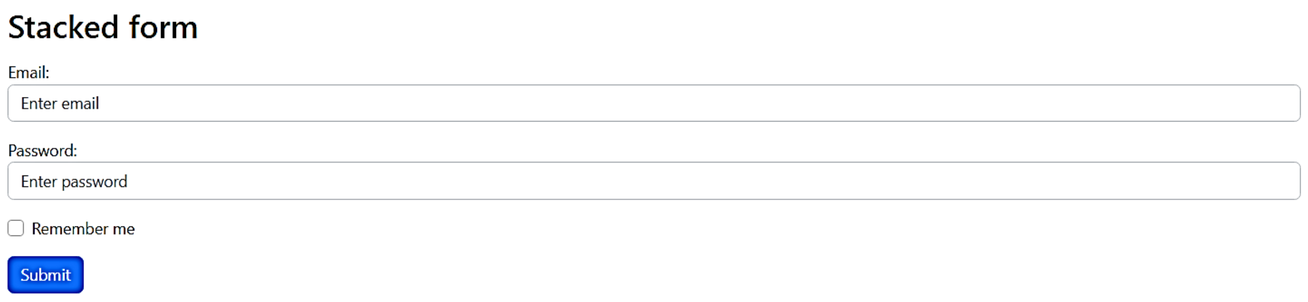
Tools – IDEs

- Visual Studio Code
- Adobe Dreamweaver
- Visual Studio

5.1 Bootstrap 5 Forms

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.

```
<form action="/action_page.php">
  <div class="mb-3 mt-3">
    <label for="email" class="form-label">Email:</label>
    <input type="email" class="form-control" id="email" placeholder="Enter
email" name="email">
  </div>
  <div class="mb-3">
    <label for="pwd" class="form-label">Password:</label>
    <input type="password" class="form-
control" id="pwd" placeholder="Enter password" name="pswd">
  </div>
  <div class="form-check mb-3">
    <label class="form-check-label">
      <input class="form-check-
input" type="checkbox" name="remember"> Remember me
    </label>
  </div>
  <button type="submit" class="btn btn-primary">Submit</button>
</form>
```


Stacked form

Email:

Enter email

Password:

Enter password

Remember me

Text-area

```
<label for="comment">Comments:</label>
<textarea class="form-
control" rows="5" id="comment" name="text"></textarea>
```

Form Row/Grid (Inline Forms)

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

```
<form>
  <div class="row">
    <div class="col">
      <input type="text" class="form-control" placeholder="Enter
email" name="email">
    </div>
    <div class="col">
      <input type="password" class="form-control" placeholder="Enter
password" name="pswd">
    </div>
  </div>
</form>
```

Form Control Size

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

```
<input type="text" class="form-control form-control-lg" placeholder="Large
input">
<input type="text" class="form-control" placeholder="Normal input">
<input type="text" class="form-control form-control-sm" placeholder="Small
input">
```

Disabled and Readonly

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

```
<input type="text" class="form-control" placeholder="Normal input">
<input type="text" class="form-control" placeholder="Disabled
input" disabled>
<input type="text" class="form-control" placeholder="Readonly
input" readonly>
```

Plain text Inputs

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

```
<input type="text" class="form-control-plaintext" placeholder="Plaintext
input">
```

```
<input type="text" class="form-control" placeholder="Normal input">
```

Color Picker

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

```
<input type="color" class="form-control form-control-
color" value="#CCCCCC">
```

File input

```
<div class="mb-3">
  <label for="formFile" class="form-label">Default file input
example</label>
  <input class="form-control" type="file" id="formFile">
</div>
<div class="mb-3">
  <label for="formFileMultiple" class="form-label">Multiple files input
example</label>
  <input class="form-control" type="file" id="formFileMultiple" multiple>
</div>
<div class="mb-3">
  <label for="formFileDisabled" class="form-label">Disabled file input
example</label>
  <input class="form-control" type="file" id="formFileDisabled" disabled>
</div>
<div class="mb-3">
  <label for="formFileSm" class="form-label">Small file input
example</label>
  <input class="form-control form-control-sm" id="formFileSm" type="file">
</div>
<div>
  <label for="formFileLg" class="form-label">Large file input
example</label>
  <input class="form-control form-control-lg" id="formFileLg" type="file">
</div>
```

Datalists

Datalists allow you to create a group of <option>s that can be accessed (and autocompleted) from within an <input>. These are similar to <select> elements, but come with more menu styling limitations and differences. While most browsers and operating systems include some support for <datalist> elements, their styling is inconsistent at best.

```
<label for="exampleDataList" class="form-label">Datalist example</label>
<input class="form-control" list="datalistOptions" id="exampleDataList"
placeholder="Type to search...">
<datalist id="datalistOptions">
  <option value="San Francisco">
  <option value="New York">
  <option value="Seattle">
  <option value="Los Angeles">
  <option value="Chicago">
</datalist>
```

Approach

Browser default checkboxes and radios are replaced with the help of .form-check, a series of classes for both input types that improves the layout and behavior of their HTML elements, that provide greater customization and cross browser consistency. Checkboxes are for selecting one or several options in a list, while radios are for selecting one option from many.

Structurally, our <input>s and <label>s are sibling elements as opposed to an <input> within a <label>. This is slightly more verbose as you must specify id and for attributes to relate the <input> and <label>. We use the sibling selector (~) for all our <input> states, like :checked or :disabled. When combined with the .form-check-label class, we can easily style the text for each item based on the <input>'s state.

Our checks use custom Bootstrap icons to indicate checked or indeterminate states.

Checks

```
<div class="form-check">
  <input class="form-check-input" type="checkbox" value="" id="flexCheckDefault">
  <label class="form-check-label" for="flexCheckDefault">
    Default checkbox
  </label>
</div>
```

```
<div class="form-check">
  <input class="form-check-input" type="checkbox" value="" id="flexCheckChecked"
checked>
  <label class="form-check-label" for="flexCheckChecked">
    Checked checkbox
  </label>
</div>
```

Indeterminate

Checkboxes can utilize the :indeterminate pseudo class when manually set via JavaScript (there is no available HTML attribute for specifying it).

```
<div class="form-check">
  <input class="form-check-input" type="checkbox" value=""
id="flexCheckIndeterminate">
  <label class="form-check-label" for="flexCheckIndeterminate">
    Indeterminate checkbox
  </label>
</div>
```

Radio:

```
<div class="form-check">
  <input class="form-check-input" type="radio" name="flexRadioDefault"
id="flexRadioDefault1">
  <label class="form-check-label" for="flexRadioDefault1">
    Default radio
  </label>
</div>
<div class="form-check">
  <input class="form-check-input" type="radio" name="flexRadioDefault"
id="flexRadioDefault2" checked>
  <label class="form-check-label" for="flexRadioDefault2">
    Default checked radio
  </label>
</div>
```

Sitches

A switch has the markup of a custom checkbox but uses the .form-switch class to render a toggle switch. Switches also support the disabled attribute.

```
<div class="form-check form-switch">
  <input class="form-check-input" type="checkbox" id="flexSwitchCheckDefault">
  <label class="form-check-label" for="flexSwitchCheckDefault">Default switch
checkbox input</label>
</div>
<div class="form-check form-switch">
  <input class="form-check-input" type="checkbox" id="flexSwitchCheckChecked"
checked>
  <label class="form-check-label" for="flexSwitchCheckChecked">Checked switch
checkbox input</label>
</div>
<div class="form-check form-switch">
  <input class="form-check-input" type="checkbox" id="flexSwitchCheckDisabled"
disabled>
  <label class="form-check-label" for="flexSwitchCheckDisabled">Disabled switch
checkbox input</label>
</div>
<div class="form-check form-switch">
  <input class="form-check-input" type="checkbox"
id="flexSwitchCheckCheckedDisabled" checked disabled>
  <label class="form-check-label" for="flexSwitchCheckCheckedDisabled">Disabled
checked switch checkbox input</label>
</div>
```

Default (stacked)

By default, any number of checkboxes and radios that are immediate sibling will be vertically stacked and appropriately spaced with .form-check.

```
<div class="form-check">
  <input class="form-check-input" type="checkbox" value="" id="defaultCheck1">
  <label class="form-check-label" for="defaultCheck1">
    Default checkbox
  </label>
</div>
<div class="form-check">
  <input class="form-check-input" type="checkbox" value="" id="defaultCheck2"
disabled>
  <label class="form-check-label" for="defaultCheck2">
    Disabled checkbox
  </label>
</div>
```

Inline

Group checkboxes or radios on the same horizontal row by adding .form-check-inline to any .form-check.

```
<div class="form-check form-check-inline">
  <input class="form-check-input" type="checkbox" id="inlineCheckbox1"
value="option1">
  <label class="form-check-label" for="inlineCheckbox1">1</label>
</div>
<div class="form-check form-check-inline">
  <input class="form-check-input" type="checkbox" id="inlineCheckbox2"
value="option2">
  <label class="form-check-label" for="inlineCheckbox2">2</label>
</div>
<div class="form-check form-check-inline">
  <input class="form-check-input" type="checkbox" id="inlineCheckbox3"
value="option3" disabled>
  <label class="form-check-label" for="inlineCheckbox3">3 (disabled)</label>
</div>
```

Toggle buttons

Create button-like checkboxes and radio buttons by using .btn styles rather than .form-check-label on the <label> elements. These toggle buttons can further be grouped in a button group if needed.

```
<input type="checkbox" class="btn-check" id="btn-check" autocomplete="off">
<label class="btn btn-primary" for="btn-check">Single toggle</label>


<input type="checkbox" class="btn-check" id="btn-check-2" checked
autocomplete="off">
<label class="btn btn-primary" for="btn-check-2">Checked</label>


<input type="checkbox" class="btn-check" id="btn-check-3" autocomplete="off"
disabled>
<label class="btn btn-primary" for="btn-check-3">Disabled</label>
```

Outlined styles

Different variants of .btn, such at the various outlined styles, are supported.

```
<input type="checkbox" class="btn-check" id="btn-check-outlined" autocomplete="off">
<label class="btn btn-outline-primary" for="btn-check-outlined">Single
toggle</label><br>

<input type="checkbox" class="btn-check" id="btn-check-2-outlined" checked
autocomplete="off">
<label class="btn btn-outline-secondary" for="btn-check-2-
outlined">Checked</label><br>

<input type="radio" class="btn-check" name="options-outlined" id="success-outlined"
autocomplete="off" checked>
<label class="btn btn-outline-success" for="success-outlined">Checked success
radio</label>

<input type="radio" class="btn-check" name="options-outlined" id="danger-outlined"
autocomplete="off">
<label class="btn btn-outline-danger" for="danger-outlined">Danger radio</label>
```

5.2 Flex:

Give your forms some structure—from inline to horizontal to custom grid implementations—with our form layout options.

Enable flex behaviors

Apply display utilities to create a flexbox container and transform direct children elements into flex items. Flex containers and items are able to be modified further with additional flex properties.

<div class="d-flex p-2 bd-highlight">I'm a flexbox container!</div>

<div class="d-inline-flex p-2 bd-highlight">I'm an inline flexbox container!</div>

Responsive variations also exist for .d-flex and .d-inline-flex.

- .d-flex
- .d-inline-flex
- .d-sm-flex
- .d-sm-inline-flex
- .d-md-flex
- .d-md-inline-flex
- .d-lg-flex

- .d-lg-inline-flex
- .d-xl-flex
- .d-xl-inline-flex
- .d-xxl-flex
- .d-xxl-inline-flex

Direction

Set the direction of flex items in a flex container with direction utilities. In most cases you can omit the horizontal class here as the browser default is row. However, you may encounter situations where you needed to explicitly set this value (like responsive layouts). Use .flex-row to set a horizontal direction (the browser default), or .flex-row-reverse to start the horizontal direction from the opposite side.

```
<div class="d-flex flex-row bd-highlight mb-3">
  <div class="p-2 bd-highlight">Flex item 1</div>
  <div class="p-2 bd-highlight">Flex item 2</div>
  <div class="p-2 bd-highlight">Flex item 3</div>
</div>
<div class="d-flex flex-row-reverse bd-highlight">
  <div class="p-2 bd-highlight">Flex item 1</div>
  <div class="p-2 bd-highlight">Flex item 2</div>
  <div class="p-2 bd-highlight">Flex item 3</div>
</div>
```

```
<div class="d-flex flex-column bd-highlight mb-3">
  <div class="p-2 bd-highlight">Flex item 1</div>
  <div class="p-2 bd-highlight">Flex item 2</div>
  <div class="p-2 bd-highlight">Flex item 3</div>
</div>
<div class="d-flex flex-column-reverse bd-highlight">
  <div class="p-2 bd-highlight">Flex item 1</div>
  <div class="p-2 bd-highlight">Flex item 2</div>
  <div class="p-2 bd-highlight">Flex item 3</div>
</div>
```

Justify content
Use justify-content utilities on flexbox containers to change the alignment of flex items on the main axis (the x-axis to start, y-axis if flex-direction: column). Choose from start (browser default), end, center, between, around, or evenly.

```
<div class="d-flex justify-content-start">...</div>
<div class="d-flex justify-content-end">...</div>
<div class="d-flex justify-content-center">...</div>
<div class="d-flex justify-content-between">...</div>
<div class="d-flex justify-content-around">...</div>
<div class="d-flex justify-content-evenly">...</div>
```

Align items

Use align-items utilities on flexbox containers to change the alignment of flex items on the cross axis (the y-axis to start, x-axis if flex-direction: column). Choose from start, end, center, baseline, or stretch (browser default).

```
<div class="d-flex align-items-start">...</div>
<div class="d-flex align-items-end">...</div>
<div class="d-flex align-items-center">...</div>
<div class="d-flex align-items-baseline">...</div>
<div class="d-flex align-items-stretch">...</div>
```

Align self

Use align-self utilities on flexbox items to individually change their alignment on the cross axis (the y-axis to start, x-axis if flex-direction: column). Choose from the same options as align-items: start, end, center, baseline, or stretch (browser default).

```
<div class="align-self-start">Aligned flex item</div>
<div class="align-self-end">Aligned flex item</div>
<div class="align-self-center">Aligned flex item</div>
<div class="align-self-baseline">Aligned flex item</div>
<div class="align-self-stretch">Aligned flex item</div>
```

Fill

Use the .flex-fill class on a series of sibling elements to force them into widths equal to their content (or equal widths if their content does not surpass their border-boxes) while taking up all available horizontal space.

```
<div class="d-flex bd-highlight">
  <div class="p-2 flex-fill bd-highlight">Flex item with a lot of content</div>
  <div class="p-2 flex-fill bd-highlight">Flex item</div>
  <div class="p-2 flex-fill bd-highlight">Flex item</div>
</div>
```

5.3: Carousel

A slideshow component for cycling through elements—images or slides of text—like a carousel.

How it works

The carousel is a slideshow for cycling through a series of content, built with CSS 3D transforms and a bit of JavaScript. It works with a series of images, text, or custom markup. It also includes support for previous/next controls and indicators. In browsers where the Page Visibility API is supported, the carousel will avoid sliding when the webpage is not visible to the user (such as when the browser tab is inactive, the browser window is minimized, etc.).

```
<div id="carouselExampleSlidesOnly" class="carousel slide" data-bs-
ride="carousel">
  <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>
</div>
```

With controls

Adding in the previous and next controls. We recommend using <button> elements, but you can also use <a> elements with role="button"

```
<div id="carouselExampleControls" class="carousel slide" data-bs-
ride="carousel">
  <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="#carouselExampleControls" 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="#carouselExampleControls" data-bs-slide="next">
    <span class="carousel-control-next-icon" aria-hidden="true"></span>
    <span class="visually-hidden">Next</span>
  </button>
</div>
```

With indicators

You can also add the indicators to the carousel, alongside the controls, too.

```
<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>
```

With captions

Add captions to your slides easily with the .carousel-caption element within any .carousel-item. They can be easily hidden on smaller viewports, as shown below, with optional display utilities. We hide them initially with .d-none and bring them back on medium-sized devices with .d-md-block.

```
<div id="carouselExampleCaptions" class="carousel slide" data-bs-
ride="carousel">
  <div class="carousel-indicators">
    <button type="button" data-bs-target="#carouselExampleCaptions" data-bs-
slide-to="0" class="active" aria-current="true" aria-label="Slide 1"></button>
    <button type="button" data-bs-target="#carouselExampleCaptions" data-bs-
slide-to="1" aria-label="Slide 2"></button>
```

```
    <button type="button" data-bs-target="#carouselExampleCaptions" 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 class="carousel-caption d-none d-md-block">
        <h5>First slide label</h5>
        <p>Some representative placeholder content for the first slide.</p>
      </div>
    </div>
    <div class="carousel-item">
      <img src="..." class="d-block w-100" alt="...">
      <div class="carousel-caption d-none d-md-block">
        <h5>Second slide label</h5>
        <p>Some representative placeholder content for the second slide.</p>
      </div>
    </div>
    <div class="carousel-item">
      <img src="..." class="d-block w-100" alt="...">
      <div class="carousel-caption d-none d-md-block">
        <h5>Third slide label</h5>
        <p>Some representative placeholder content for the third slide.</p>
      </div>
    </div>
  </div>
  <button class="carousel-control-prev" type="button" data-bs-
target="#carouselExampleCaptions" 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="#carouselExampleCaptions" data-bs-slide="next">
    <span class="carousel-control-next-icon" aria-hidden="true"></span>
    <span class="visually-hidden">Next</span>
  </button>
</div>
```

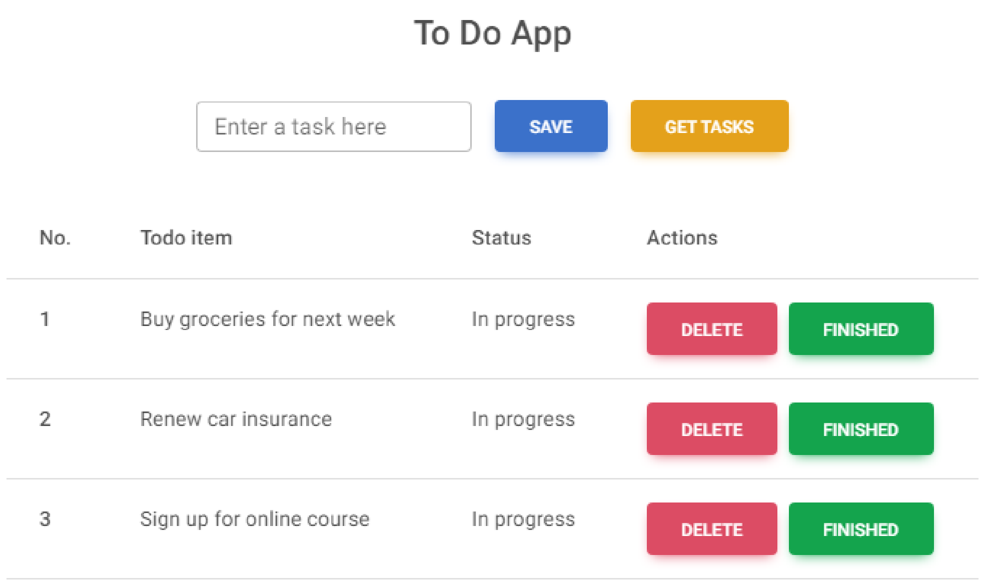
Lab Task

Task-1

Create a web page using Bootstrap-5 Form, all types of INPUT TYPES [which discussed in the lab manual]. **Make it interactive using Bootstrap -5 other classes like margin, padding.**

Task-2

Create a web page as like below snap using Bootstrap 5.



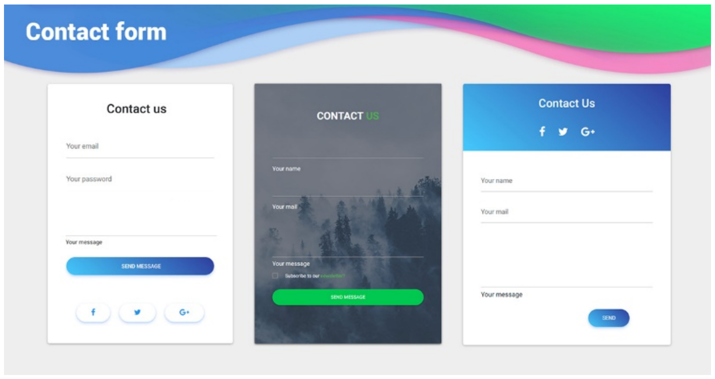
To Do App

Enter a task here | SAVE | GET TASKS

No.	Todo item	Status	Actions
1	Buy groceries for next week	In progress	DELETE FINISHED
2	Renew car insurance	In progress	DELETE FINISHED
3	Sign up for online course	In progress	DELETE FINISHED

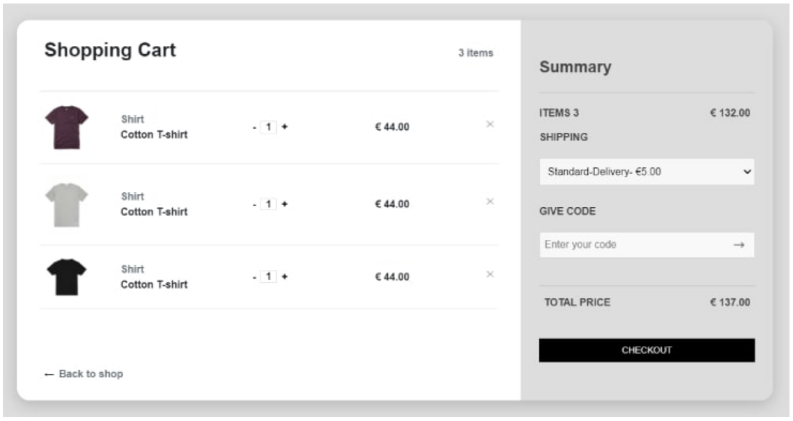
Task-3

Create a web page as like below snap using Bootstrap 5.



Task-4

Create a web page as like below snap using Bootstrap 5.



Task-5

Create a web page that showing the layout of any concept like **TABLE, NAVBAR or FOOTER** using these 3 main classes; container [and it types], Grid [and it types] and Flex [and its types], then distinguish it with respect to the working.

Task-6

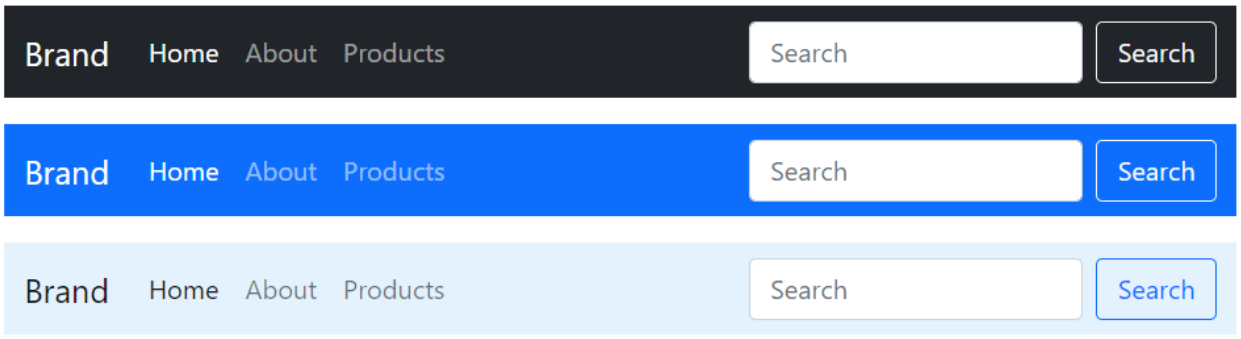
Create a web page having slider on using Carousel, the slider with content, indicators and control that will move automatically.

Task-7

In the Bootstrap you have already learn different classes where **MODEL, TOOLTIP and POPOVER** are the important classes with respect to the layout, so create a **Form of any type [login, signup, contact us, Message/Query]** and uses these types of classes on it to show the professional layout.

Task-8

Create a **NavBar** using **Flex** in Bootstrap-5 to as like below snap, where [**nav navbar]** classes will be used as usual for basic layout.


Brand
Home
About
Products
Search
Search
Brand
Home
About
Products
Search
Search
Brand
Home
About
Products
Search
Search