Software Engineering

A Faculty of Informatics and Computer Science Course: CSEN 303

Front-end Design & Technologies

10

Dr. Iman Awaad

iman.awaad@giu-uni.de

GERMAN INTERNATIONAL UNIVERSITY
الجامعة الألمانية الدولية

Review

9 Software Architectures

- How do we organise our system into components and how do they fit together?
- What are characteristics of a well-designed system?

- Architecture
- Characteristics of good architectures
- Patterns
- Client-server
- Model-View-Controller
- Layered

Acknowledgments

The slides are (**mostly**) by **Prof. Dr. John Zaki**. His contribution is gratefully acknowledged.

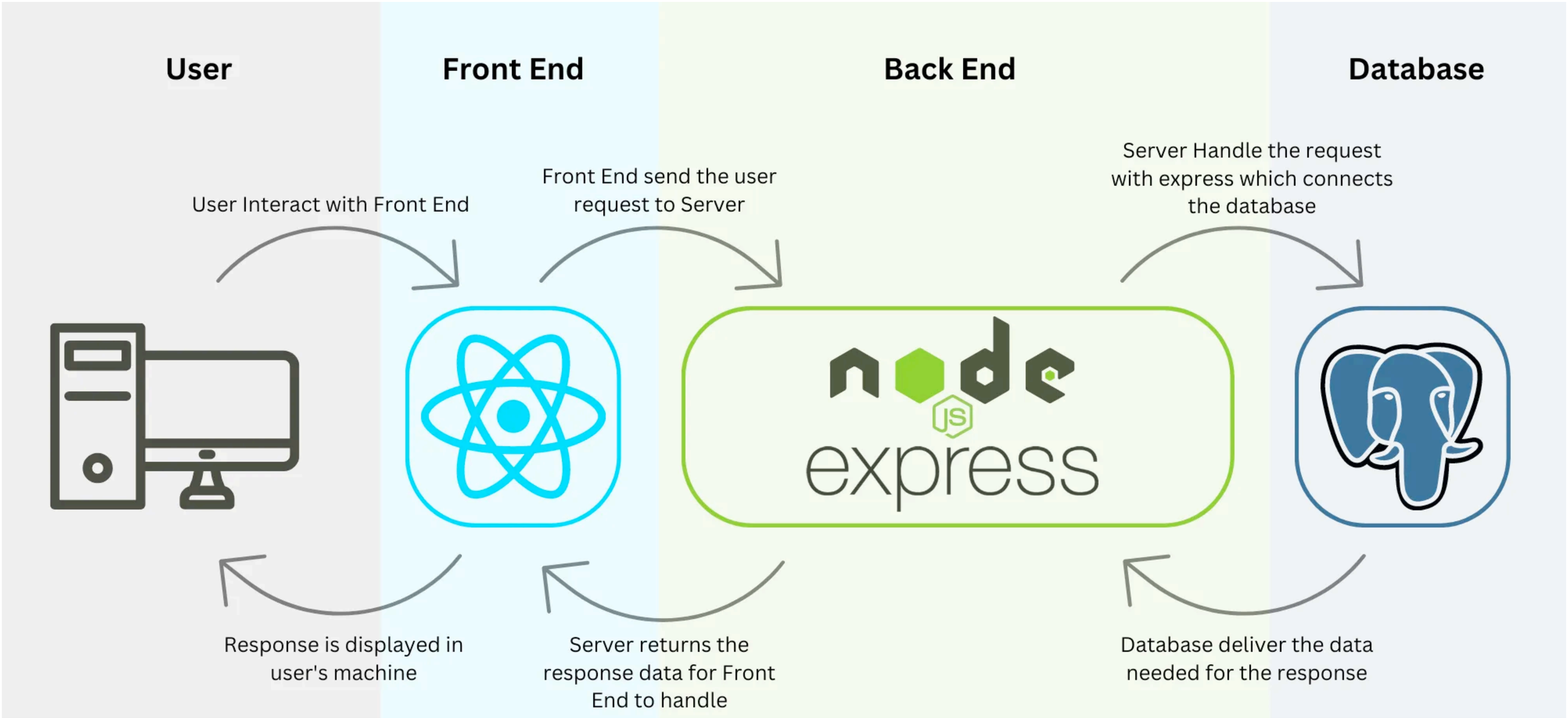
Any additional sources are referenced.

Front-end Design & Technologies

- Fullstack
  - Frontend
    - HTML
    - CSS
    - Bootstrap
    - JavaScript Libraries
      - jQuery
      - AJAX
  - Backend

How do we build and deliver robust and dynamic web applications?

The PERN stack



https://medium.com/@snehadissanayaka56/exploring-the-pern-stack-build-robust-and-dynamic-web-applications-1bf04d6e5c82

Front-End vs. Back-End

Front-End

The user interface that users see and interact with. It is responsible for the visual layout, user interaction, and overall feel of the website. This includes technologies like HTML, CSS, and JavaScript and many other frameworks.

Back-End

The server-side of a website, which handles data storage, logic, and server-side functionality. It is responsible for processing information and managing the website's data. Technologies used include databases, server-side languages, and APIs.

Front-end Design & Technologies

HTML: for **structuring the content of web pages** using a system of **markup tags** to define elements like headings, paragraphs, images, and links

CSS: a **style sheet language** used to **describe the presentation and visual formatting** of a document written in a markup language, most commonly HTML

Bootstrap: a free and open-source front-end **framework** **used for building responsive, mobile-first websites and web applications** (has collections of pre-designed HTML, CSS, and JavaScript components and templates) — also: **responsive design**, **cross-browser compatibility**, **accessible** design and much more!

jQuery: a fast, small, and feature-rich **JavaScript library** designed to **simplify client-side scripting of HTML**…streamlining common tasks that would typically require more verbose native JavaScript code…

AJAX: a set of **techniques** used to **create more dynamic and responsive web applications** by **enabling asynchronous communication** with a server **without requiring a full page reload**

HTML

…for **structuring the content of web pages** using a system of **markup tags** to define elements like headings, paragraphs, images, and links

Structure and Semantics: uses elements (like `<p>` for paragraphs, `<h1>` for headings, `<img>` for images, `<a>` for links) to provide a **logical structure** to the content. These elements also provide **semantic meaning**, **helping browsers, search engines, and assistive technologies understand the purpose of different parts of the page**!!!

Content Organization: arranging text, images, videos, forms, tables, and other elements in a meaningful hierarchy…. Without HTML, a web page would be an unformatted blob of information.

Creating Links: HTML's "Hypertext" aspect refers to its ability to **create links** (`<a>` tags) that connect different web pages, both within a single website and across the internet. **This linking mechanism is a core feature of the World Wide Web**!!!!

Foundation for Styling and Interactivity: paired with CSS (for styling and presentation) and JavaScript (for interactivity and dynamic behaviour), **creates a complete and engaging user experience**. HTML is the base upon which these other technologies build.

Introduction to HTML5

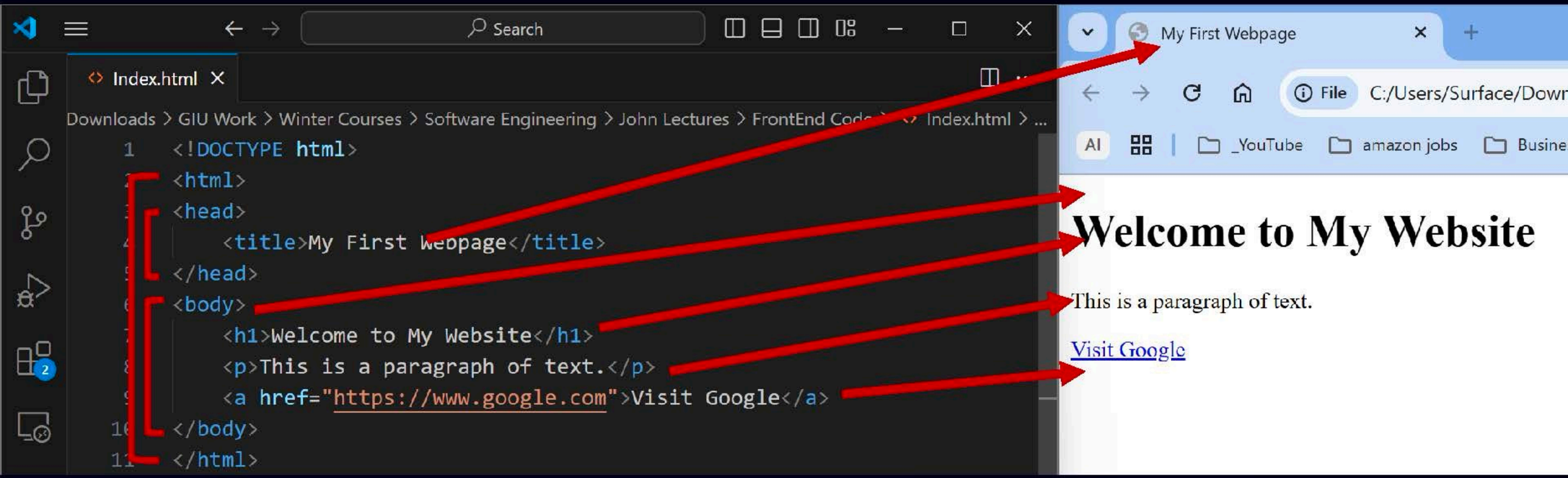
1 What is HTML5?

HTML5 is the latest version of the **HyperText Markup Language**.

It defines the structure and content of a website allowing users to interact with content on a web page.

2 Basic HTML Structure

Every HTML document follows a standardized structure. The basic HTML structure includes tags like: <html> , <head> , <body> <title> , <h1> ..... <h6>, <p> <a>, <img>, ,

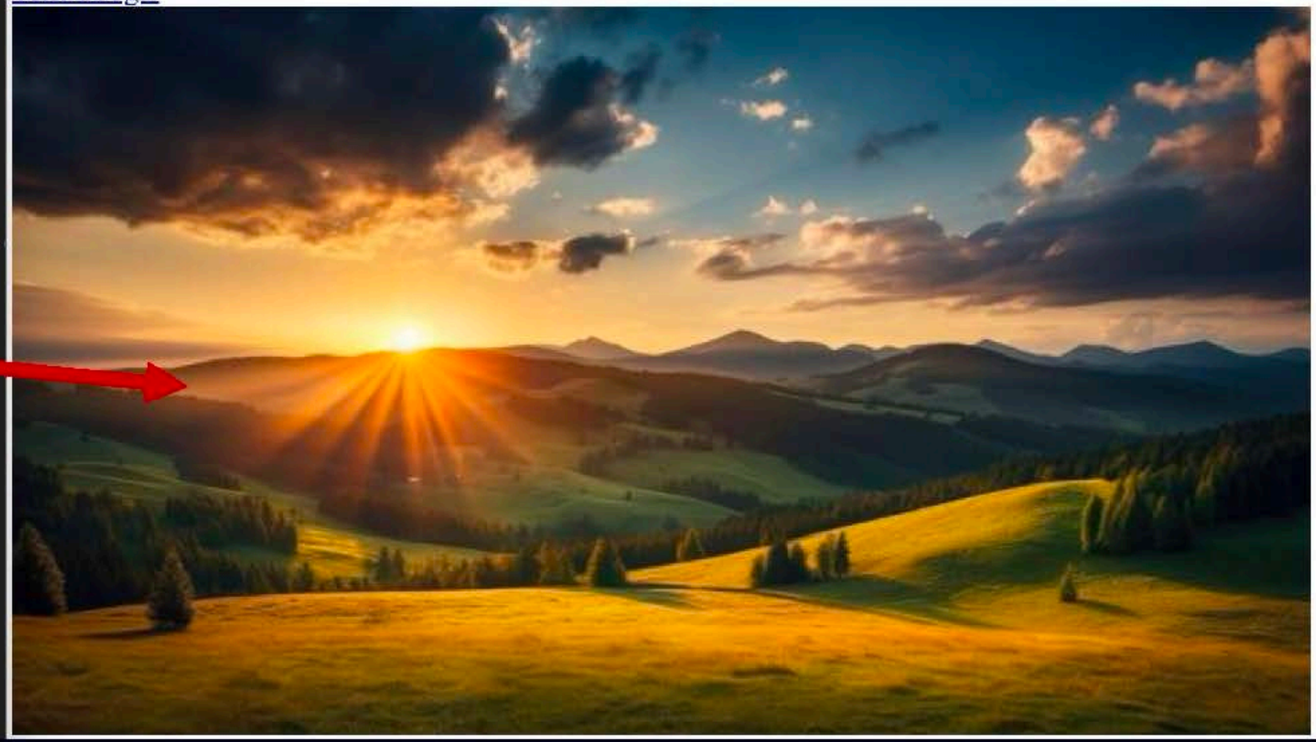

Search
Index.html
Downloads > GIU Work > Winter Courses > Software Engineering > John Lectures > FrontEnd Code > Index.html > ...
1 <!DOCTYPE html>
<html>
<head>
<title>My First Webpage</title>
</head>
<body>
<h1>Welcome to My Website</h1>
This is a paragraph of text.
Visit Google
</body>
11 </html>
My First Webpage
File
C:/Users/Surface/Down
AI
_YouTube
amazon jobs
Busine
Welcome to My Website
This is a paragraph of text.
Visit Google

```
3
4  <!DOCTYPE html>
5  <html>
6  <head>
7      <title>My First Webpage</title>
8  </head>
9  <body>
10
11     <h1>Welcome to My Website</h1>
12     <p>This is a paragraph of text.</p>
13     <a href="https://www.google.com">Visit Google</a>
14     <img src="nature.jpeg" alt="beautiful sunset on a mountain" width="
15 </body>
16 </html>
17
```

Welcome to My Website

This is a paragraph of text.

Visit Google



HTML5 Document Structure

"HTML Living Standard"... no longer versioned numerically!

Nesting Elements

HTML tags can be nested within each other, allowing you to create a hierarchical structure. This helps organize your content and define relationships between different elements.

```
<!DOCTYPE html>
<html>
<head>
    <title>Nested Elements Example</title>
</head>
<body>
    <header>
        <h1>Welcome to My Website</h1>
        <nav>
            <ul>
                <li><a href="Index.html">Home</a></li>
                <li><a href="about.html">About</a></li>
                <li><a href="contact.html">Contact</a></li>
            </ul>
        </nav>
    </header>
    <section>
        <article>
            <h2>Latest News</h2>
            <p>This is a paragraph inside an article section.</p>
        </article>
    </section>
    <footer>
        <p>&copy; 2024 My Website</p>
    </footer>
</body>
</html>
```

See: https://html.spec.whatwg.org/

Creating Forms

1 Form Tag

The <form> tag is used to create a form. It has attributes like action and method to specify where the form data should be sent and how it should be sent.

2 HTML5 Input Types

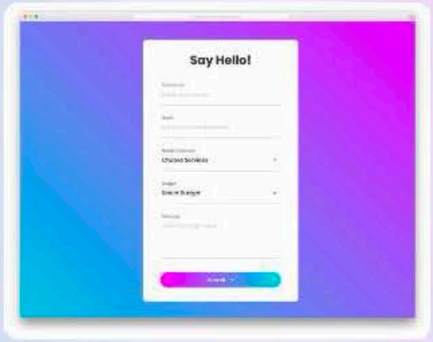
HTML5 provides various input types for different purposes, including: - text: for single-line text input - email: for email addresses - password: for password input - number: for numeric input - checkbox: for selecting multiple options - radio: for selecting one option from a group - file: for uploading files

3 Form Structure

A form typically consists of input fields, labels, and a submit button. Input fields allow users to enter data, labels provide descriptions for the fields, and the submit button sends the form data to the server.

4 Form Validation

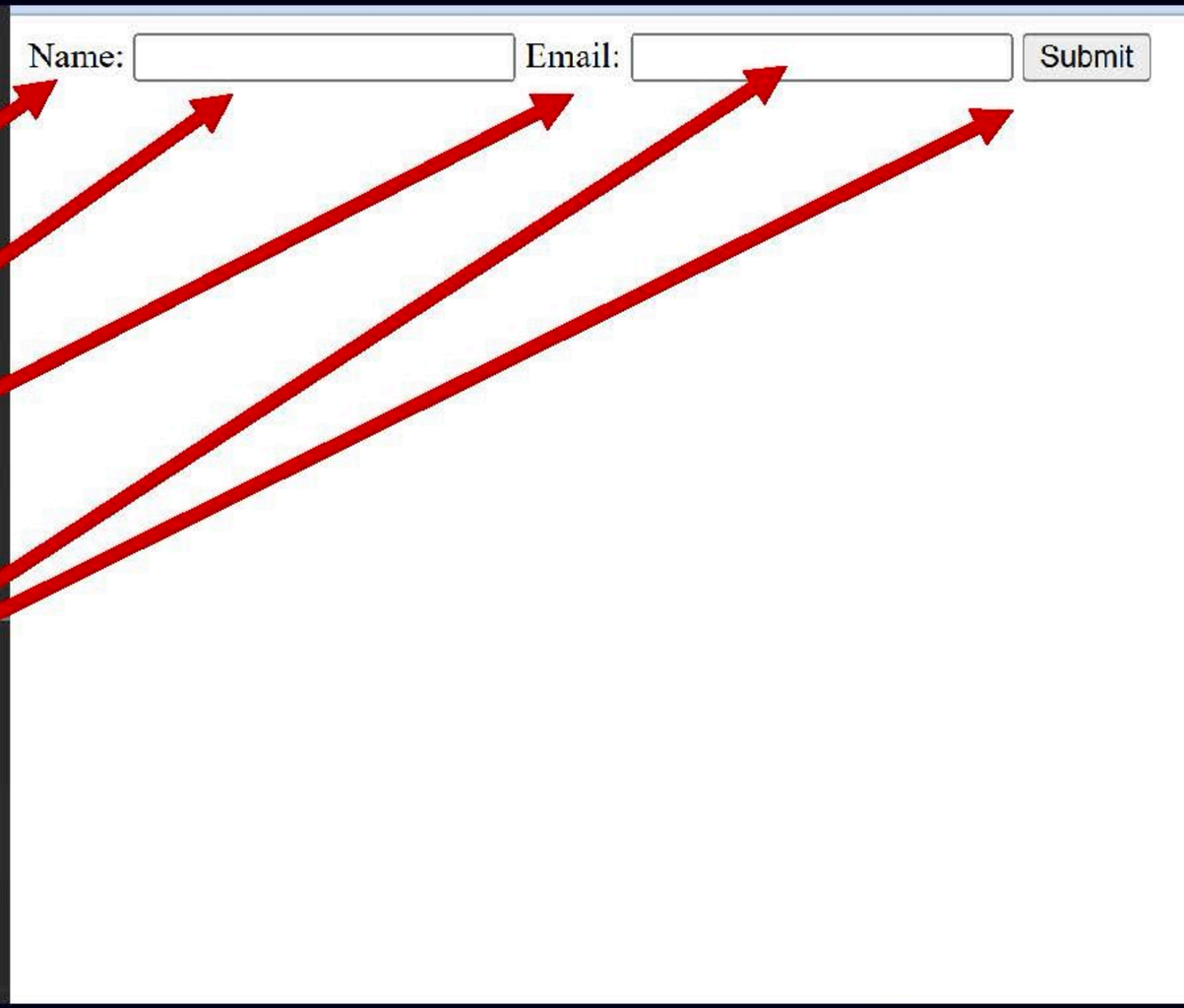
HTML5 provides built-in form validation features to ensure that users enter valid data. You can use attributes like required, min, max, and pattern to enforce specific input requirements.



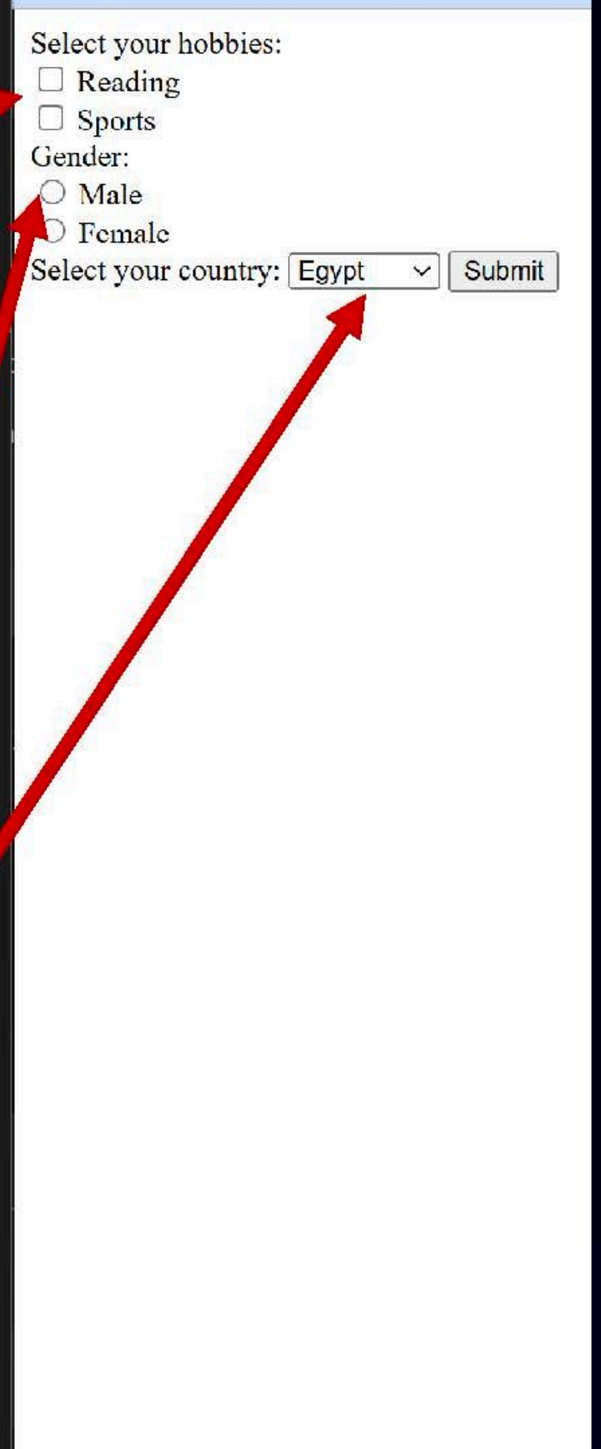
Creating Forms

Tag	Description
<form>	Defines a form for user input
INPUT	Defines a text input field for users to enter text
TEXTAREA	Defines a multi-line text input area for users to enter large amounts of text
SELECT	Defines a drop-down menu list for users to choose from a list of options
BUTTON	Defines a button that submits the form data

```
2
3   <!DOCTYPE html>
4   <html>
5   <head>
6       <title>My First Webpage</title>
7
8   </head>
9   <body>
10      <form action="/submit" method="post">
11          <label for="name">Name:</label>
12          <input type="text" id=" 123 " name="name" required>
13
14          <label for="email">Email:</label>
15          <input type="email" id=" 456 " name="email" required>
16
17          <input type="submit" value="Submit">
18      </form>
19
20  </body>
21  </html>
```



```html
<!DOCTYPE html>
<html>
<head>
    <title>My First Webpage</title>
</head>
<body>
    <form>
        <label>Select your hobbies:</label><br>
        <input type="checkbox" id="hobby1" name="hobby" value="reading">
        <label for="hobby1">Reading</label><br>

        <input type="checkbox" id="hobby2" name="hobby" value="sports">
        <label for="hobby2">Sports</label><br>

        <label>Gender:</label><br>
        <input type="radio" id="male" name="gender" value="male">
        <label for="male">Male</label><br>

        <input type="radio" id="female" name="gender" value="female">
        <label for="female">Female</label><br>


            <label for="country">Select your country:</label>
            <select id="country" name="country">
                <option value="us">Egypt</option>
                <option value="uk">Germany</option>
                <option value="canada">Canada</option>
            </select>


        <input type="submit" value="Submit">
    </form>
</body>
</html>
```



CSS

CSS: a **style sheet language** used to **describe the presentation and visual formatting** of a document writ
in a markup language, most commonly HTML

Styling HTML elements: *e.g.* properties like colors, fonts, text sizes, backgrounds, borders, and shadows

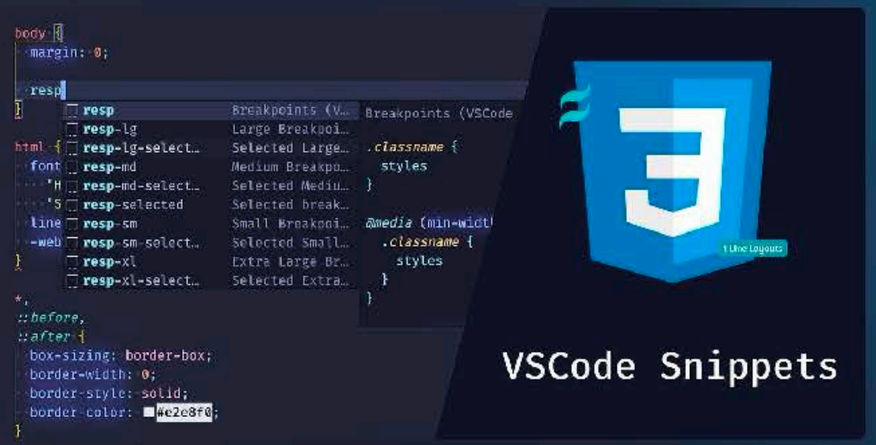
Layout and positioning: enables the arrangement of elements on a page, creating layouts like **grids**, **flexboxes**, and **multi-column** designs. It also handles the positioning of individual elements.

Responsive design: allows websites to **adapt their layout and appearance to different screen sizes and devices…** consistent and optimal user experience across devices.

Adding visual effects and animations: can be used to create transitions, animations, and other dynamic visual effects to enhance user engagement

Separation of concerns: promotes the **separation of content** (HTML) **from presentation** (CSS), making code more organized, maintainable, and reusable

Introduction to CSS3



1

CSS3 is a style sheet language that controls the appearance of web pages, including font styles, colors, layout, and responsiveness. It is used to enhance the visual presentation and user experience of websites.

2

CSS selectors are used to target specific elements on a web page. There are various types of selectors, such as element selectors, class selectors, and ID selectors.

3

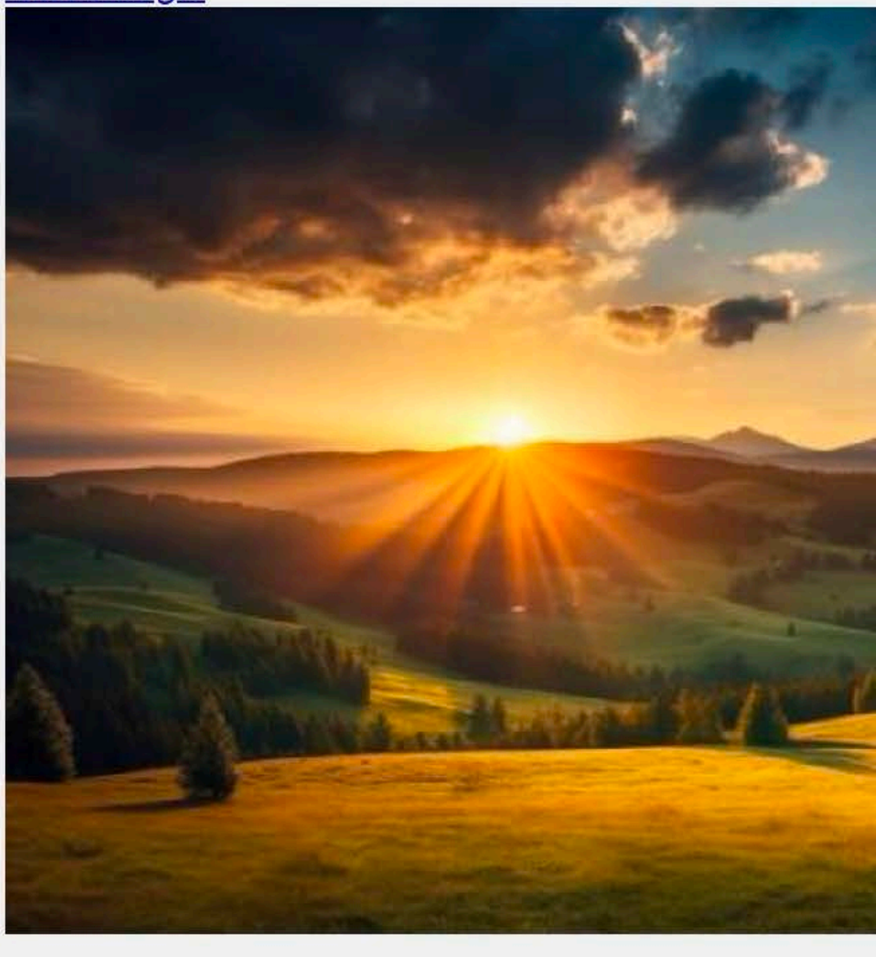
CSS syntax is straightforward. Properties are defined using the format "property: value" and enclosed within curly braces.

Selector {property: value; }

Ex: h1{ color:red; } will change the font color to red in any h1 tag in the page.

CSS in the Head

```html
<!DOCTYPE html>
<html>
<head>
    <title>My First Webpage</title>
<style>
    body {
        background-color: #f0f0f0;
        font-family: Arial;
        color: #3489db
    }
    h1 {
        color: #3498db;
    }
</style>
</head>
<body>

    <h1>Welcome to My Website</h1>
    <p>This is a paragraph of text.</p>
    <a href="https://www.google.com">Visit Google</a>
    <img src="nature.jpeg" alt="beautiful sunset on a mountain" width="" height="
</body>
</html>
```



CSS in an external .css file

Index.html ● | style.css

Work > Winter Courses > Software Engineering > John Lectures > FrontEnd Code >

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>My First Webpage</title>
5      <link rel="stylesheet" href="style.css">
6  </head>
7  <body>
8
9      <h1>Welcome to My Website</h1>
10     <p>This is a paragraph of text.</p>
11     <a href="https://www.google.com">Visit Goog
12     <img src="nature.jpeg" alt="beautiful sunse
13 </body>
14 </html>
```

Index.html | style.css ×

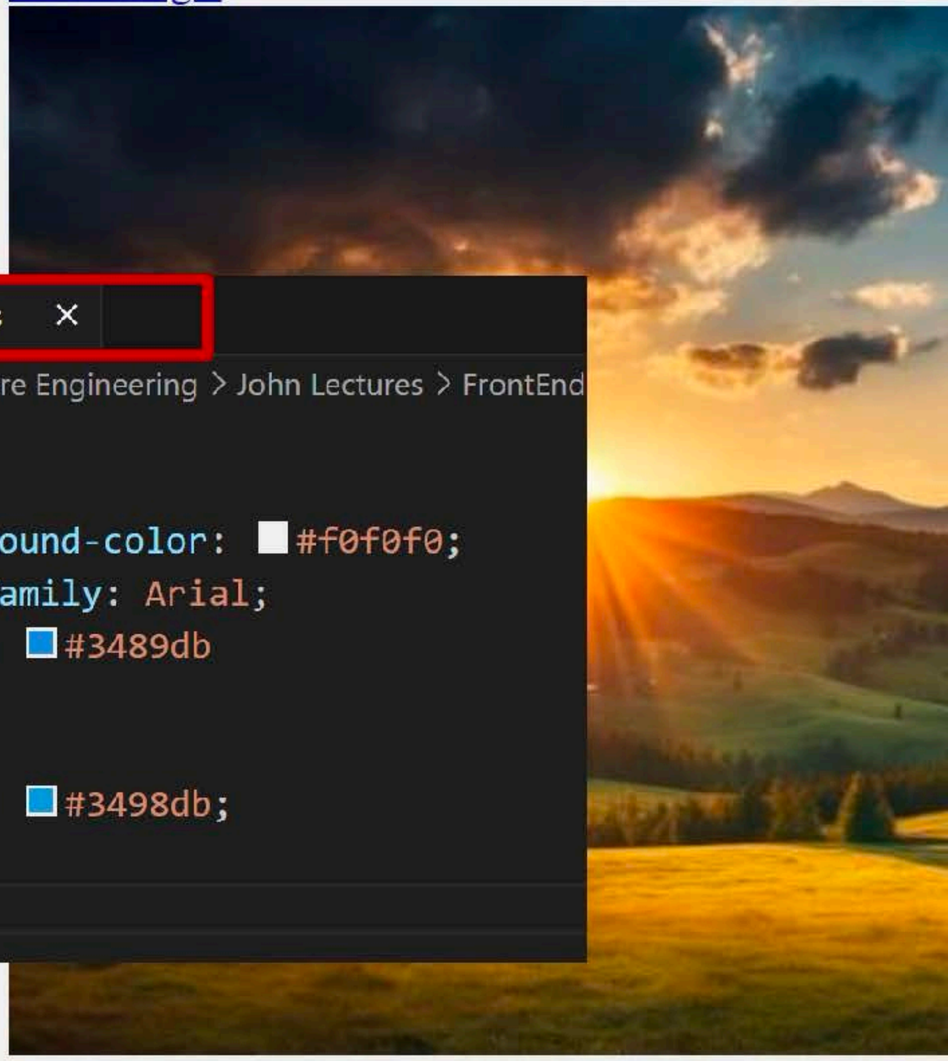
J Work > Winter Courses > Software Engineering > John Lectures > FrontEnd

```
1
2  body {
3      background-color: #f0f0f0;
4      font-family: Arial;
5      color: #3489db
6  }
7  h1 {
8      color: #3498db;
9  }
10
```

Welcome to My Website

This is a paragraph of text.

Visit Google



Class Selector

```html
<!DOCTYPE html>
<html>
<head>
    <title>My Portfolio</title>
    <style>
        body { font-family: Arial, sans-serif;
            background-color: #f0f0f0; }
        .container { margin: auto; width: 50%;
            padding: 50px; background: #fff; }
    </style>
</head>
<body>
    <div class="container">
        <h1>Hello, I'm a Web Developer!</h1>
        <p>Welcome to my portfolio page.</p>
    </div>
</body>
</html>
```

ID Selector

The #id selector in CSS is used to apply styles to a specific HTML element that has a unique id attribute. Unlike class selectors (.class), which can be used on multiple elements, an id should be unique within the page. This means that only one element can have a particular id value.

```
3
4   <!DOCTYPE html>
5   <html lang="en">
6   <head>
7       <title>ID Selector Example</title>
8       <link rel="stylesheet" href="style5.css">
9   </head>
10  <body>
11      <h1 id="main-heading">Welcome to My Website</h1>
12      <p id="intro">This is a paragraph with a unique ID.</p>
13      <p>This is another paragraph without an ID.</p>
14  </body>
15  </html>
```


```
/* ID selector styles */
#main-heading {
    color: #2c3e50;
    text-align: center;
    font-size: 36px;
}

#intro {
    color: #16a085;
    font-size: 18px;
    font-weight: bold;
}
```

Bootstrap

…a free and open-source front-end **framework** **used for building responsive, mobile-first websites and web applications** (has collections of pre-designed HTML, CSS, and JavaScript components and templates) — also: **responsive design**, **cross-browser compatibility**, **accessible** design and more!

Rapid Development: ready-to-use components *e.g.* navigation bars, buttons, forms, cards, and carousels. **Reduces the need to write custom CSS and JavaScript from scratch**

Responsive Design: Built with a **"mobile-first" approach**, the responsive grid system and utility classes ensure that websites **adapt** seamlessly **to various screen sizes**, from mobile phones to desktops

Cross-Browser Compatibility: ensures that the design and functionality work as expected across different web browsers

Consistent Styling: provides a unified set of **styles and design guidelines**, helping to maintain a consistent visual appearance throughout a website or application

Accessibility: includes features and guidelines to help developers create more accessible web interfaces!!!!!!!!!!!!

Extensibility and **Customisation**: the base set of styles can be easily customised using **Sass variables** and **custom CSS** to match specific design requirements

Introduction to Bootstrap

Framework

Bootstrap is a popular front-end framework that provides a collection of pre-designed components and utilities, simplifying the development process. It offers a consistent design system and helps build responsive websites.
It has **Pre-built Components:**

Grid System

Bootstrap's grid system is a flexible and efficient layout system that allows developers to easily create responsive layouts that adjust to different screen sizes. It uses a 12-column grid system, making it easy to arrange content in a well-organized manner.
Components of grid system are
Containers, rows, columns, breakpoints

Hands-On: Using Bootstrap Components

1

Bootstrap Components

Bootstrap provides ready-to-use components like buttons, navbars, forms, and more. These components can be easily customized to match the design requirements of a website. Download from bootstrap official website.

2

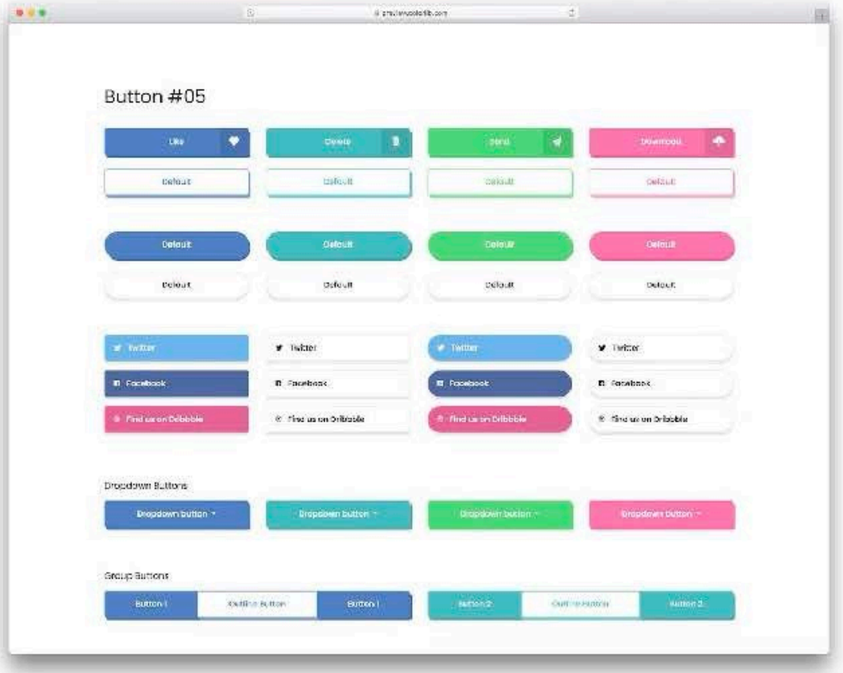
Integration

Integrating Bootstrap into your website is straightforward. Include the Bootstrap CSS and JavaScript files in your HTML and start using the components and styles it offers. Local installation or CDN delivery

```
<link rel="stylesheet" href="path/to/bootstrap.min.css">
<script src="path/to/bootstrap.bundle.min.js"></script>
```

CDN delivery ... Check bootstrap website

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3
```



FYI: Content Delivery Network to link your website to Bootstrap's framework files (CSS and JavaScript) instead of hosting them on your own server.

jQuery

…a fast, small, and feature-rich **JavaScript library** designed to **simplify client-side scripting of HTML**…streamlining common tasks that would typically require more verbose native JavaScript code…

DOM Manipulation: Easily **select**, **traverse**, and **modify elements** within the Document Object Model (DOM), such as changing content, attributes, or CSS styles.

Event Handling: simplify the process of attaching **event listeners** to HTML elements and handling user interactions like clicks, hovers, form submissions, and keyboard inputs.

Animations and Effects: Create various **visual effects and animations**, *e.g.* fading, sliding, and custom animations, to enhance user experience

AJAX: a set of **techniques** used to **create more dynamic and responsive web applications** by **enabling asynchronous communication** with a server **without requiring a full page reload**

Cross-Browser Compatibility: a consistent API that abstracts away browser inconsistencies, ensuring that code works reliably across different web browsers.

FYI…

jQuery was once the main player in front-end development, its role has evolved…

Modern JavaScript features , powerful front-end frameworks like React, Angular, and Vue.js….

It remains relevant for **maintaining legacy projects** and **can still be useful for smaller projects** or **specific tasks** where its **simplified syntax provides a quick and efficient solution**.

Introduction to JavaScript Libraries

1 What are JavaScript libraries?

JavaScript libraries are collections of pre-written JavaScript code that provide reusable functionality. They simplify common tasks and enhance the interactivity of web pages.

2 Overview of jQuery and AJAX

jQuery is a popular JavaScript library that simplifies DOM manipulation, event handling, and AJAX requests. AJAX (Asynchronous JavaScript and XML) allows web pages to communicate with servers without reloading the entire page, creating a more dynamic user experience.

3 Benefits of using libraries

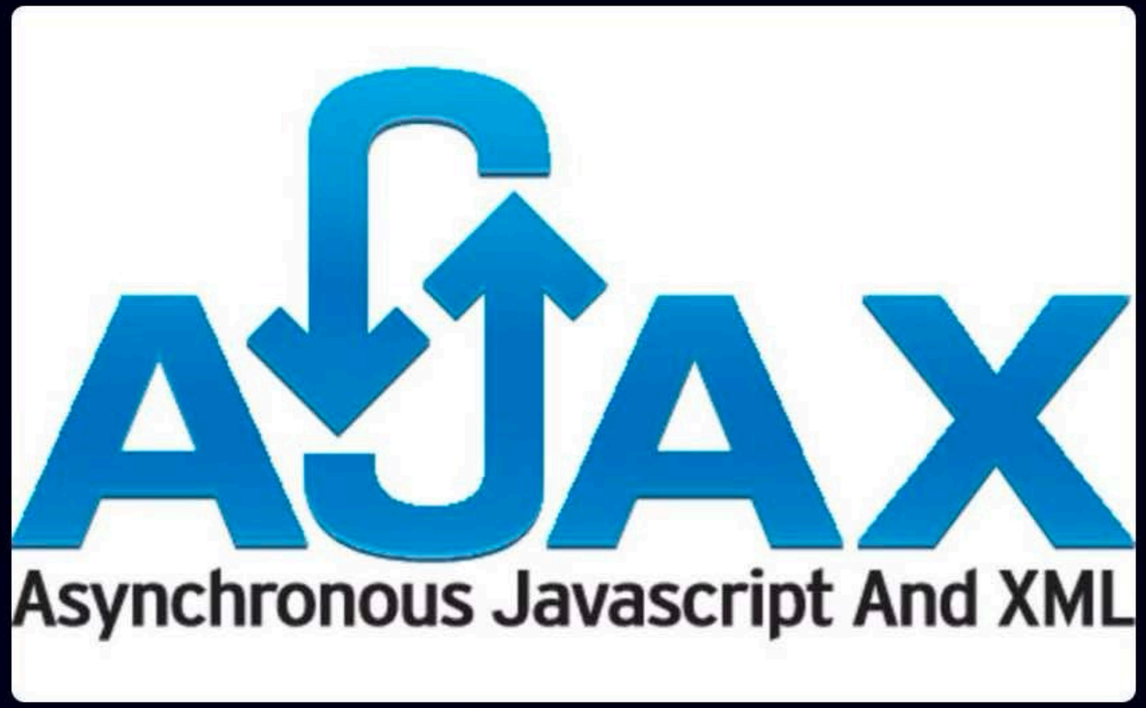
Using JavaScript libraries offers several benefits: -

- Reduced Development Time
- Libraries provide pre-written code
- Improved Code Quality:
- Libraries are well-tested and maintained, ensuring code quality and reliability.
- Enhanced Functionality:
- Libraries provide a wide range of features and functionalities.

Introduction to JavaScript Libraries

jQuery

jQuery is a popular JavaScript library that simplifies DOM manipulation, event handling, and AJAX interactions. It provides a concise and efficient way to write JavaScript code and interact with web pages.



AJAX

AJAX (Asynchronous JavaScript and XML) allows web pages to update without a full page reload. It enables dynamic interactions, making web pages more interactive and responsive.

Download jQuery

Latest version

To locally download these files, right-click the link and select "Save as..." from the menu.

Download the compressed, production version:

Download jQuery 3.7.1

- Download the uncompressed development version of jQuery 3.7.1
- Download the map file for jQuery 3.7.1
- jQuery 3.7.1 blog post with release notes

The slim build is a smaller version, that excludes the ajax and effects modules:

- Download jQuery 3.7.1 slim build
- Download the uncompressed development version of the jQuery 3.7.1 slim build
- Download the map for the jQuery 3.7.1 slim build

The uncompressed version is best used during development or debugging; the compressed file saves bandwidth and improves performance in production. You can download the source map file to help with debugging the compressed production version. The source map is *not* required for end-users to run jQuery; it is a tool to help improve a developer's debugging experience. As of jQuery 1.11/2.1, we no longer link source maps to compressed releases by default.

Browse the jQuery CDN at releases.jquery.com for a full list of assets, including older and historical versions.

jQuery CDN

To use the jQuery CDN, reference the file in the script tag directly from the jQuery CDN domain. You can get the complete script tag, including Subresource Integrity attribute, by visiting https://releases.jquery.com and clicking on the version of the file that you want to use. Copy and paste that tag into your HTML file.

The jQuery CDN supports Subresource Integrity (SRI) (specification) which allows the browser to verify that the files being delivered have not been modified. Adding the new integrity attribute will ensure your application gains this security improvement in supporting browsers.

Starting with jQuery 1.9, sourcemap files are available on the jQuery CDN. However, as of version 1.10.0/2.1.0 the compressed jQuery no longer includes the sourcemap comment in CDN copies because it requires the uncompressed file and sourcemap file to be placed at the same location as the compressed file. If you are maintaining local copies and can control the locations all three files, you can add the sourcemap comment to the compressed file for easier debugging.

To see all available files and versions, including older and historical versions, visit https://releases.jquery.com

Other CDNs

The following CDNs also host compressed and uncompressed versions of jQuery releases. Starting with jQuery 1.9 they may also host sourcemap files; check the site's documentation.

Note that **there may be delays between a jQuery release and its availability there**. Please be patient, they receive the files at the same time the blog post is made public. Beta and release candidates are not hosted by these CDNs.

- Google CDN
- Microsoft CDN
- CDNJS CDN
- jsDelivr CDN

```
<head>
    <title>jQuery Example</title>
    <!-- jQuery CDN -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
</head>
```

Basic jQuery Syntax

1 Selecting Elements

jQuery makes it easy to select HTML elements **using selectors, similar to CSS selectors**, allowing you to manipulate and interact with specific elements on a web page. The basic Syntax

2 Basic Syntax

$(selector).action()

$: is to access jQuery Library
(selector): used to find the HTML element
action(). Applies the action to the selected element

3 Examples

$(“p”).hide()
$(“button”).click()
$(this).hover()

Class selector → $(“.test).hide()
ID selector → $(“#test”).hide()

Basic jQuery Syntax

1 Document Ready

Use your jQuery inside a document.ready() function to ensure page finished loading before jQuery code starts to run.

This is to prevent any jQuery code from running before the document is finished loading (is ready).

```
$(document).ready(function(){
  // jQuery methods go here...
});
```

2 Loading JS in a separate file

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
<script src="my_jquery_functions.js"></script>
</head>
```

Basic jQuery Syntax

1 Event Handling

jQuery provides methods like .click(), .hover(), ...etc allowing you to add interactivity to your web page. You can define actions to be performed when a user clicks on an element or hovers over it.

2 Event Handling

An event is when something happens (event fires). Every action from the website user is an event such as moving the mouse over an element **.hover()**

3 Examples

Mouse Events:	.click(), .dblclick()
Keyboard Events:	.keypress(), .keyup()
Form Events:	.submit(), .change()
Window Events:	.resize(), .load()

EXAMPLE

Write your script inside a document ready function

Assign a click event to all paragraphs on a page

Define what happens in the function when the click action happens

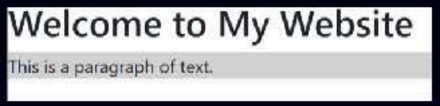
Show an alert ……. or maybe change the CSS of the paragraph – try it!

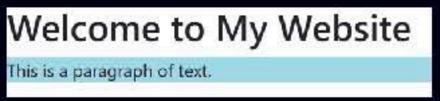
```
<script >
    document.ready(function(){

        alert("You clicked on a paragraph")

    });
</script>
```

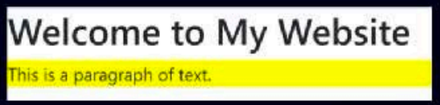
Event Handler

The on( ) method attaches one or more event handler to the selected element.

```
<script >
    $(document).ready(function(){
        $("p").on({
            mouseenter: function(){
                $(this).css("background-color", "lightgray");
            },
            mouseleave: function(){
                $(this).css("background-color", "lightblue");
            },
            click: function(){
                $(this).css("background-color", "yellow");
            }
            })
    });
</script>
```







jQuery Callback Function

JavaScript statements are executed line by line.

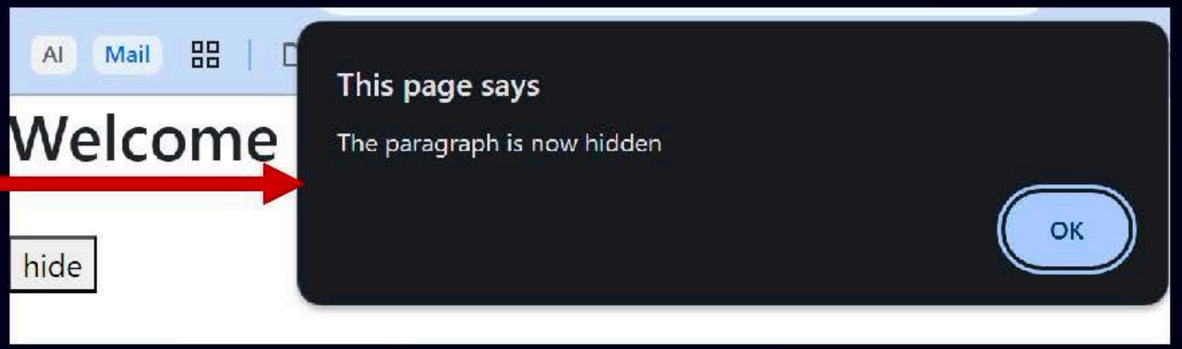
However, with effects, the next line of code can be run even though the effect is not finished. This can create errors.

To prevent this, you can create a callback function.

A callback function is executed after the current effect is finished.

$(selector).hide(speed,callback);

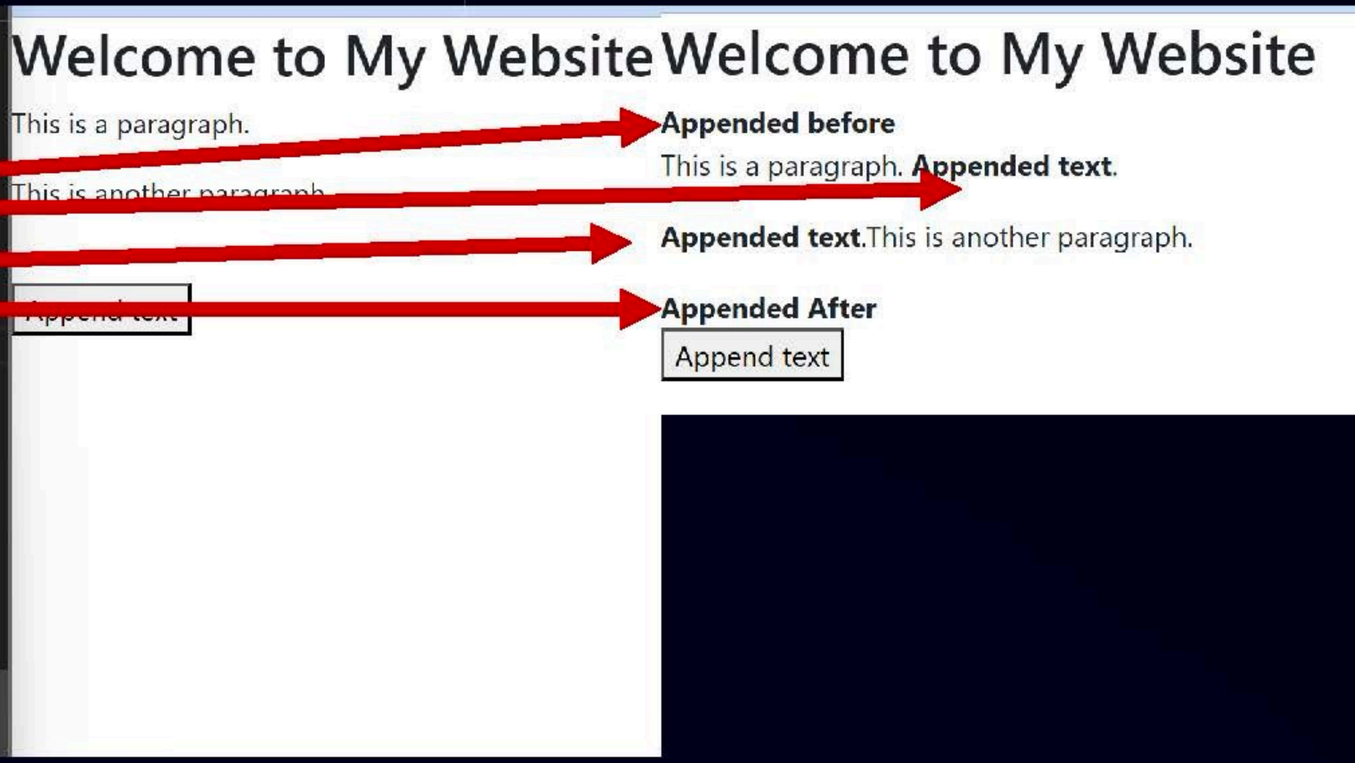
```
<body>
    <h1>Welcome to My Website</h1>
    <p>This is a paragraph of text.</p><br>
    <button>hide</button>

    <script >
        $(document).ready(function(){
        $("button").click(function(){
            $("p").hide("slow", function(){
            alert("The paragraph is now hidden");
            });
        });
        });
    </script>


</body>
```



jQuery Add Content

.append( )	Inserts content at the end of the selected elements
.prepend( )	Inserts content at the beginning of the selected elements
.after( )	Inserts content after the selected elements
.before( )	Inserts content before the selected elements

```
<body>
  <script>
    $(document).ready(function(){
    $("#btn1").click(function(){
        $("#p1").before("<b>Appended before</b>")
        $("#p1").append(" <b>Appended text</b>.");
        $("#p2").prepend(" <b>Appended text</b>.");
        $("#p2").after("<b>Appended After</b>")

    });
    });
</script>

<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>

<br><button id="btn1">Append text</button>

</body>
```



jQuery Remove Content

.remove()	Removes the selected element and its children
.empty()	Removes the children of the selected element

jQuery CSS

.addClass( )	Adds one or more classes to the selected elements
.removeClass( )	Removes one or more classes from the selected element
.toggleClass( )	Toggles between adding/removing classes
.css( )	Sets or returns the style attribute

AJAX

…a set of **techniques** used to **create more dynamic and responsive web applications** by **enabling asynchronous communication** with a server **without requiring a full page reload**

Dynamic Content Updates: **Updating** specific **parts of a web page** with new data fetched from the server, *e.g.* loading more items in a list, refreshing a news feed, and displaying search results, **without reloading the entire page**.

Asynchronous Data Exchange: **Sending and retrieving data from a server in the background** (user continues to interact with the page while the data transfer occurs… improves user experience by eliminating the need to wait for a full page refresh)

Form Submissions and Validation: **Submitting form data** to the server and **receiving validation feedback** in real-time, without a full page reload, leading to more immediate and interactive user feedback.

Creating Single-Page Applications (SPAs): Building **applications where the entire web app resides within a single HTML document**, and AJAX is used to dynamically load and display different sections or "pages" of content as needed.

Real-time Interactions: Implementing features where data needs to be continuously exchanged with the server and reflected on the client-side without interruption *e.g.* live chat, notification updates, or collaborative editing

AJAX

Introduction

It is a technique used to enable web pages to **update asynchronously** by **exchanging data with a server** in the background.

This means that it allows web pages to be updated without reloading the entire page.

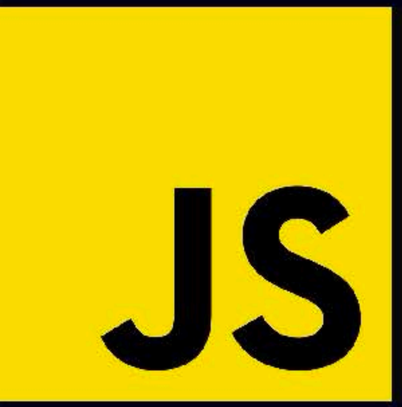
Features

Asynchronous: allows data to be fetched in the background

Partial Page Updates: Instead of refreshing the whole page, AJAX can update parts of a web page

Reduced Server Load: By only sending and receiving the data that’s necessary,

Improved User Experience: AJAX enables smoother interactions.



AJAX & XHR

How Does it Work?

AJAX uses a combination of:

HTML and **CSS** for content and styling.

JavaScript (or **jQuery**) to send and receive data.

XML or **JSON** as a data format (JSON is more commonly used today).

XMLHttpRequest (xhr) object to interact with the server.

AJAX Request Flow

1. A user action (like clicking a button) triggers the JavaScript function.

2.JavaScript creates an XMLHttpRequest (xhr) object.

3.The XMLHttpRequest object sends a request to the server.

4.The server processes the request and sends a response back.

5.JavaScript processes the server's response and updates the web page content.

Use AJAX for API calls

Syntax:

$.ajax({object})

Object contains key value pairs (some are optional)

method: 'GET' , 'POST', 'PUT', 'DELETE'

URL: 'the API link'

datatype: 'json', 'xml', 'html'

data: {for example in json format }

contentType: 'application/json'

success: function(response) { }

error: function(jqXHR, textStatus, errorThrown)

complete: function( ){ }

```
20  <body>
75      <button id="loadData">Load Data</button>
76      <div id="result"></div>
77
78  <script>
79      $(document).ready(function(){
80
81          // GET REQUEST
82          $("#loadData").click(function() {
83              $.ajax({
84                  url: "https://jsonplaceholder.typicode.com/todos/1",
85                  method: "GET",
86                  success: function(response) {
87                      $("#result").append('<h3>'+response.userId+'</h3>');
88                      console.log(response);
89                  },
90                  error: function() {
91                      alert("Failed to load data");
92                  }
93              });
94          });
95      })
96  </script>
97
    </body>
```


```
<div style="text-align:center; ">
    <button id="loadData">Load Data</button>
    <button id="postData">Send Data</button>
    <button id="updateData">Update Data</button>
    <button id="deleteData">Delete Data</button>
</div>
    <div id="result"></div>
```

Use AJAX for API calls to GET, POST, UPDATE, DELETE data from the backend

{JSON} Placeholder

Free fake and reliable API for testing and prototyping.

Powered by JSON Server + LowDB.

Serving ~3 billion requests each month.

```javascript
// GET REQUEST
$("#loadData").click(function() {
    $.ajax({
        url: "https://jsonplaceholder.typicode.com/todos/1",
        method: "GET",
        success: function(response) {
            $("#result").append('<h3>'+response.userId+'</h3>');
            console.log(response);
        },
        error: function() {
            alert("Failed to load data");
        }
    });
});
```

```javascript
// POST REQUEST

$("#postData").click(function() {

    $.ajax({
                url: "https://jsonplaceholder.typicode.com/todos",
                method: "POST",
                data: JSON.stringify({
                    id: 1,
                    title: "Updated Todo",
                    body: "This is the updated content.",
                    userId: 1
                }),
                contentType: "application/json",
                success: function(response) {
                    $("#response").html("<p>Success! " + response.message + "</p>");
                },
                error: function(xhr, status, error) {
                    $("#response").html("<p>Error: " + xhr.responseText + "</p>");
                }
            });
});
```

```javascript
// PUT REQUEST
$("#updateData").click(function() {
    $.ajax({
        url: "https://jsonplaceholder.typicode.com/todos/1",
        method: "PUT",
        data: JSON.stringify({
            id: 1,
            title: "Updated Todo",
            body: "This is the updated content.",
            userId: 1
        }),
        contentType: "application/json; charset=utf-8",
        success: function(response) {
            alert("Todo Updated: " + response.title);
        }
    });
});
```

```javascript
// DELETE REQUEST

$("#deleteData").click(function() {
    $.ajax({
        url: "https://jsonplaceholder.typicode.com/todos/1",
        method: "DELETE",
        success: function() {
            alert("Todo Deleted Successfully");
        }
    });
});
```

Try retrieving more data...

Use each

Use map

What does JSON.stringify do?

NACH DEM SPIEL
IST VOR DEM SPIEL