Chapter 13

Popular JavaScript Libraries and Frameworks

jQuery – Simplifying DOM Manipulation

Example:

$(document).ready(function(){

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

$("#content").fadeToggle();

});

});

**Adding jQuery to your webpage:**

**Example:**

**Google CDN:**

<head>

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

</head>

**CDNJS CDN:**

<head>

<script src=”<https://cdnjs.cloudflare.com/ajax/libs/jquery/3.7.1/jquery.min.js>”> </script>

</head>

Lodash – Utility Functions for JavaScript

Example:

const numbers = [1, 2, 3, 4, 5];

const doubled = \_.map(numbers, (num) => num \* 2);

console.log(doubled); // Output: [2, 4, 6, 8, 10]

Axios – Handling HTTP Requests

Example:

axios.get('https://api.example.com/users')

.then(response => console.log(response.data))

.catch(error => console.error('Error:', error));

Moment.js & Date-fns – Handling Dates and Time

Example (Using Date-fns):

import { format, addDays } from 'date-fns';

const date = new Date();

console.log(format(addDays(date, 5), 'yyyy-MM-dd')); // Output: Future date

Chart.js & D3.js – Data Visualization

Example:

const ctx = document.getElementById('myChart').getContext('2d');

const myChart = new Chart(ctx, {

type: 'bar',

data: {

labels: ['Red', 'Blue', 'Yellow'],

datasets: [{

label: 'Votes',

data: [12, 19, 3],

backgroundColor: ['red', 'blue', 'yellow']

}]

}

});

GSAP – Animations in JavaScript

Example:

gsap.to(".box", { x: 100, duration: 2, ease: "power2.out" });

Three.js – 3D Graphics in JavaScript

Example:

const scene = new THREE.Scene();

const camera = new THREE.PerspectiveCamera(75, window.innerWidth / window.innerHeight, 0.1, 1000);

const renderer = new THREE.WebGLRenderer();

renderer.setSize(window.innerWidth, window.innerHeight);

document.body.appendChild(renderer.domElement);

Library\_demo.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>JavaScript Libraries Demo</title>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/lodash.js/4.17.21/lodash.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/underscore.js/1.13.6/underscore-min.js"></script>

    <script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.29.4/moment.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/date-fns/2.30.0/date-fns.min.js"></script>

    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/d3/7.8.5/d3.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/gsap/3.12.2/gsap.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/animejs/3.2.1/anime.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/three.js/r128/three.min.js"></script>

    <style>

        body { font-family: Arial, sans-serif; text-align: center; }

        .chart-container { width: 300px; height: 300px; margin: auto; }

        .box { width: 100px; height: 100px; background-color: coral; margin: auto; }

        canvas { display: block; margin: auto; }

    </style>

</head>

<body>

    <h1>JavaScript Libraries Demo</h1>

    <div id="user-data"></div>

    <p id="date"></p>

    <div class="chart-container">

        <canvas id="chart"></canvas>

    </div>

    <svg width="400" height="100"></svg>

    <div class="box"></div>

    <div id="3d-container"></div>

    <script>

        // jQuery AJAX Request

        $(document).ready(() => {

            $.get("https://randomuser.me/api/", (data) => {

                let user = data.results[0];

                $("#user-data").html(`<h3>${user.name.first} ${user.name.last}</h3>`);

            });

        });

        // Axios HTTP Request

        axios.get("https://randomuser.me/api/").then(response => {

            let user = response.data.results[0];

            console.log("Axios User:", user);

        });

        // Fetch API HTTP Request

        fetch("https://randomuser.me/api/")

            .then(response => response.json())

            .then(data => console.log("Fetch API User:", data.results[0]));

        // Lodash & Underscore usage

        console.log("Lodash Chunk:", \_.chunk([1, 2, 3, 4, 5], 2));

        console.log("Underscore Shuffle:", \_.shuffle([1, 2, 3, 4, 5]));

        // Moment.js & Date-fns

        document.getElementById("date").innerText = "Date: " + moment().format("MMMM Do YYYY, h:mm:ss a");

        // Chart.js Visualization

        new Chart(document.getElementById("chart"), {

            type: 'bar',

            data: {

                labels: ["Red", "Blue", "Yellow"],

                datasets: [{ data: [10, 20, 30], backgroundColor: ["red", "blue", "yellow"] }]

            }

        });

        // D3.js Visualization

        d3.select("svg").append("circle")

            .attr("cx", 50).attr("cy", 50).attr("r", 40).style("fill", "green");

        // GSAP Animation

        gsap.to(".box", { duration: 2, x: 200, backgroundColor: "blue", repeat: -1, yoyo: true });

        // Anime.js Animation

        anime({ targets: ".box", rotate: '1turn', duration: 2000, easing: "easeInOutSine", loop: true });

        // Three.js 3D Cube

        let scene = new THREE.Scene();

        let camera = new THREE.PerspectiveCamera(75, 300 / 300, 0.1, 1000);

        let renderer = new THREE.WebGLRenderer();

        renderer.setSize(300, 300);

        document.getElementById("3d-container").appendChild(renderer.domElement);

        let geometry = new THREE.BoxGeometry();

        let material = new THREE.MeshBasicMaterial({ color: 0x00ff00 });

        let cube = new THREE.Mesh(geometry, material);

        scene.add(cube);

        camera.position.z = 5;

        function animate() {

            requestAnimationFrame(animate);

            cube.rotation.x += 0.01;

            cube.rotation.y += 0.01;

            renderer.render(scene, camera);

        }

        animate();

    </script>

</body>

</html>

**Installing Libraries**

**Example: Including jQuery & Lodash via CDN**

<!DOCTYPE html>

<html lang="en">

<head>

<title>Library Integration Example</title>

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/lodash.js/4.17.21/lodash.min.js"></script>

</head>

<body>

<button id="btn">Click Me</button>

<script>

$(document).ready(function() {

$('#btn').click(() => alert("Hello, jQuery!"));

});

console.log(\_.chunk([1, 2, 3, 4, 5], 2)); // Lodash chunking function

</script>

</body>

</html>

**Can be imported into JavaScript modules**

Example usage in an **ES Module (.js file)**:

import $ from 'jquery';

import \_ from 'lodash';

$(document).ready(() => {

console.log("jQuery is ready!");

});

console.log(\_.shuffle([1, 2, 3, 4])); // Lodash function

**Example: Using Axios and Chart.js in a React Component**

import React, { useEffect, useState } from 'react';

import axios from 'axios';

import { Bar } from 'react-chartjs-2';

import 'chart.js/auto';

const Dashboard = () => {

const [data, setData] = useState([]);

useEffect(() => {

axios.get('https://jsonplaceholder.typicode.com/posts')

.then(response => setData(response.data.slice(0, 5)));

}, []);

const chartData = {

labels: data.map(item => item.id),

datasets: [{

label: 'Post Length',

data: data.map(item => item.body.length),

backgroundColor: 'rgba(54, 162, 235, 0.5)',

}],

};

return <Bar data={chartData} />;

};

export default Dashboard;

**Integrating Libraries in a Node.js Project**

For back-end projects using Node.js + Express, install dependencies with npm:

npm install express axios moment lodash

**Example: Using Axios & Moment.js in an Express Server**

const express = require('express');

const axios = require('axios');

const moment = require('moment');

const app = express();

app.get('/time', (req, res) => {

res.json({ currentTime: moment().format('YYYY-MM-DD HH:mm:ss') });

});

app.get('/posts', async (req, res) => {

const response = await axios.get('https://jsonplaceholder.typicode.com/posts');

res.json(response.data.slice(0, 5));

});

app.listen(3000, () => console.log("Server running on port 3000"));

**Best Practices for Managing Dependencies**

**Example:**

"dependencies": {

"axios": "^1.5.0",

"moment": "^2.29.4"

},

"devDependencies": {

"eslint": "^8.30.0"

}

**Use .gitignore to Exclude node\_modules/**

# .gitignore

node\_modules/

.env

**Don't commit node\_modules/, use npm install after cloning the project.**

Example: Instead of using Lodash for simple array methods, use built-in JavaScript methods:

// Using Lodash

import \_ from 'lodash';

console.log(\_.shuffle([1, 2, 3, 4]));

// Using Vanilla JS

console.log([1, 2, 3, 4].sort(() => Math.random() - 0.5));

Optimize Performance with Tree Shaking

If you're using Webpack or Vite, make sure you're importing only what you need.

// BAD: Importing everything

import \_ from "lodash";

console.log(\_.chunk([1, 2, 3, 4, 5], 2));

// GOOD: Importing only required function

import { chunk } from "lodash";

console.log(chunk([1, 2, 3, 4, 5], 2));

This reduces bundle size and improves load time.

Managing Dependencies in Your Project

Example: package.json

"dependencies": {

"axios": "^1.5.0",

"date-fns": "^2.30.0"

},

"devDependencies": {

"eslint": "^8.30.0",

"webpack": "^5.88.0"

}

**Code Example (React Component):**

import React, { useState } from "react";

function Counter() {

const [count, setCount] = useState(0);

return (

<div>

<h2>Counter: {count}</h2>

<button onClick={() => setCount(count + 1)}>Increment</button>

</div>

);

}

export default Counter;

**React Example:**

* Create the following components inside the src directory:   Home.jsx

import React from 'react';

const Home = () => {

  return (

    <div>

      <h2>Home Page</h2>

      <p>Welcome to the Home Page!</p>

    </div>

  );

};

export default Home;

* Create the following components inside the src directory:   About.jsx

import React from 'react';

const About = () => {

  return (

    <div>

      <h2>About Page</h2>

      <p>This is the About Page.</p>

    </div>

  );

};

export default About;

* Create the following components inside the src directory:   Contact.jsx

import React from 'react';

const Contact = () => {

  return (

    <div>

      <h2>Contact Page</h2>

      <p>Get in touch with us!</p>

    </div>

  );

};

export default Contact;

**Step 4: Set Up Routing in the App Component**

* Edit src/App.jsx to include routing:

import React from 'react';

import { BrowserRouter as Router, Route, Routes, Link } from 'react-router-dom';

import Home from './Home';

import About from './About';

import Contact from './Contact';

function App() {

  return (

    <Router>

      <div className="App">

        <nav>

          <ul>

            <li>

              <Link to="/">Home</Link>

            </li>

            <li>

              <Link to="/about">About</Link>

            </li>

            <li>

              <Link to="/contact">Contact</Link>

            </li>

          </ul>

        </nav>

        <Routes>

          <Route path="/" element={<Home />} />

          <Route path="/about" element={<About />} />

          <Route path="/contact" element={<Contact />} />

        </Routes>

      </div>

    </Router>

  );

}

export default App;

**Step 5: Implement Data Binding**

* Create a new component called DataBinding.jsx in the src directory:

import React, { useState } from 'react';

const DataBinding = () => {

  const [name, setName] = useState('');

  const [greeting, setGreeting] = useState('');

  const handleChange = (e) => {

    setName(e.target.value);

  };

  const handleSubmit = (e) => {

    e.preventDefault();

    setGreeting(`Hello, ${name}!`);

  };

  return (

    <div>

      <h2>Data Binding Example</h2>

      <form onSubmit={handleSubmit}>

        <input

          type="text"

          value={name}

          onChange={handleChange}

          placeholder="Enter your name"

        />

        <button type="submit">Greet</button>

      </form>

      {greeting && <p>{greeting}</p>}

    </div>

  );

}; export default DataBinding;

* Add a route for DataBinding in App.jsx:

import DataBinding from './DataBinding';

// Add this in the <nav> section

<li>

  <Link to="/databinding">Data Binding</Link>

</li>

// Add this in the <Routes> section

<Route path="/databinding" element={<DataBinding />} />

**Step 6: Start the Development Server**

* Run the development server as shown in Figure 13.7, to see your application in action:

npm run dev

The application will be available at http://localhost:5173

**Code Example (Angular Component):**

import { Component } from '@angular/core';

@Component({

selector: 'app-counter',

template: `<h2>Counter: {{count}}</h2>

<button (click)="increment()">Increment</button>`

})

export class CounterComponent {

count = 0;

increment() {

this.count++;

}

}

**Code Example (Vue Component):**

<template>

<div>

<h2>Counter: {{ count }}</h2>

<button @click="count++">Increment</button>

</div>

</template>

<script>

export default {

data() {

return { count: 0 };

}

};

</script>

**Backend JavaScript Frameworks**

**Code Example (Basic Server):**

const http = require('http');

const server = http.createServer((req, res) => {

res.writeHead(200, { 'Content-Type': 'text/plain' });

res.end('Hello, World!');

});

server.listen(3000, () => console.log('Server running on port 3000'));

**Express.js**

**Code Example (Express Server):**

const express = require('express');

const app = express();

app.get('/', (req, res) => res.send('Hello, Express!'));

app.listen(3000, () => console.log('Server running on port 3000'));

**Code Example (Next.js API Route):**

export default function handler(req, res) {

res.status(200).json({ message: "Hello from Next.js API!" });

}

**Other Specialized JavaScript Frameworks**

**Svelte**

**Code Example:**

<script>

let count = 0;

</script>

<h2>Counter: {count}</h2>

<button on:click={() => count++}>Increment</button>

**NestJS (Backend Framework for Node.js)**

**Code Example (NestJS Controller):**

import { Controller, Get } from '@nestjs/common';

@Controller()

export class AppController {

@Get()

getHello(): string {

return 'Hello, NestJS!';

}

}