Day	Topic	Definition & Uses
1	Functions and Scope	Functions are blocks of reusable code. Scope defines the visibility and accessibility of variables. Used for organizing code and controlling variable access.
2	Arrays and Array Methods	Arrays store multiple values. Methods like push, pop, map, and filter provide powerful ways to manipulate arrays. Used for handling collections of data.
3	Objects and Object-oriented Programming (OOP)	Objects group data and functions. OOP principles like encapsulation, inheritance, and polymorphism help structure code. Used for building modular and scalable applications.
4	DOM Manipulation	Document Object Model manipulation changes HTML content dynamically. Used for creating interactive and dynamic web pages.
5	Events and Event Handling	Events are user actions (clicks, keypress). Event handling enables responding to these actions. Used for creating responsive web applications.
6	Asynchronous JavaScript (Callbacks, Promises)	Asynchronous programming handles tasks concurrently. Callbacks, Promises, and async/await are used to manage asynchronous operations. Essential for handling I/O operations efficiently.
7	AJAX and Fetch API	Asynchronous JavaScript and XML (AJAX) enables data exchange with a server without refreshing the page. Fetch API simplifies making HTTP requests. Used for efficient data retrieval.
8	Error Handling and Debugging	Handling errors gracefully and debugging techniques are crucial for robust applications. Used for identifying and fixing issues in code.
9	ES6 Features (Arrow Functions, let/const, etc.)	ES6 introduced many features like arrow functions, let/const for variable declaration, destructuring, etc. Enhances code readability and maintainability.
10	Closures and Callbacks	Closures allow functions to retain access to variables from their outer scope. Callbacks are functions passed as arguments to other functions. Used for advanced control flow and handling asynchronous operations.
11	Prototypes and Inheritance	Prototypes and inheritance are key concepts in JavaScript's OOP model. Used for creating and extending object types.
12	JavaScript Modules (CommonJS, ES6 Modules)	Modules help organize and structure code. CommonJS and ES6 Modules define how to modularize JavaScript code. Used for building scalable and maintainable applications.
13	Promises and Async/Await	Promises represent a value that might be available now or in the future. Async/Await simplifies asynchronous code, making it more readable. Essential for managing asynchronous tasks in a synchronous style.
14	Webpack and Module Bundling	Webpack is a module bundler for JavaScript applications. It bundles and optimizes code, improving performance. Used for building and bundling complex applications.
15	Local Storage and Cookies	Local Storage and Cookies store data on the client-side. Used for persisting data between sessions and enhancing user experience.
16	JSON and RESTful APIs	JSON (JavaScript Object Notation) is a data interchange format. RESTful APIs use HTTP methods for communication between the client and server. Essential for data exchange in web development.
17	Testing with Jest or Mocha/Chai	Testing frameworks like Jest or Mocha/Chai ensure code reliability. Used for automating tests and verifying the correctness of code.
18	Node.js and npm	Node.js enables server-side JavaScript. npm is the package manager for Node.js. Used for building scalable and efficient server-side applications.
19	Express.js (Building a simple server)	Express.js is a web application framework for Node.js. Used for building server-side applications and APIs.
20	Middleware in Express.js	Middleware functions process incoming requests before reaching the route handler. Used for adding functionality to Express.js applications.
21	Authentication and Authorization (JWT)	Authentication verifies user identity, and authorization controls access to resources. JSON Web Tokens (JWT) are commonly used for secure token-based authentication.
22	WebSockets	WebSockets provide full-duplex communication channels over a single, long-lived connection. Used for real-time communication in web applications.
23	Reactive Programming (RxJS)	Reactive Programming involves handling asynchronous data streams. RxJS is a popular library for reactive programming in JavaScript. Used for managing complex asynchronous workflows.
24	GraphQL Basics	GraphQL is a query language for APIs. It provides a more efficient and flexible alternative to traditional REST APIs. Used for efficiently fetching and updating data in client-server communication.
25	Building a Full-Stack Application (Frontend + Backend)	Integrating frontend and backend development to create a complete web application. Applies all learned concepts to create a real-world project.

Day	Topic	Definition & Uses
26	Design Patterns (Singleton, Observer, etc.)	Design patterns are reusable solutions to common problems in software design. They provide templates for solving problems in a way that's both tested and documented. Used for creating scalable and maintainable code.
27	Web Security (CORS, HTTPS, Content Security Policy)	Understanding and implementing security measures to protect web applications. CORS (Cross-Origin Resource Sharing), HTTPS (Hypertext Transfer Protocol Secure), and Content Security Policy are essential for secure web development.
28	Progressive Web Apps (PWAs)	PWAs are web applications that provide a native app-like experience. They work offline, offer push notifications, and provide an engaging user experience. Used for building modern and responsive web applications.
29	WebAssembly	WebAssembly (Wasm) is a binary instruction format that enables high-performance execution on web browsers. Used for running code at near-native speed in web applications.
30	Serverless Architecture (AWS Lambda, Azure Functions)	Serverless architecture abstracts server management, allowing developers to focus on writing code. AWS Lambda and Azure Functions are examples of serverless computing. Used for building scalable and cost-efficient applications.