1️⃣ Core JavaScript topics (for *both* frontend & backend)

🔹 Basics

- Variables: `var`, `let`, `const`
- Data types: string, number, boolean, null, undefined, object, array, symbol
- Operators: arithmetic, comparison, logical
- Conditionals: `if`, `else`, `switch`
- Loops: `for`, `while`, `do...while`, `for...of`, `for...in`
- Functions:
    - Function declaration
    - Function expression
    - Arrow functions `() => {}`

🔹 Working with data

- Arrays: methods like `.push()`, `.pop()`, `.slice()`, `.splice()`
- Array higher-order functions: `.map()`, `.filter()`, `.reduce()`, `.find()`, `.some()`, `.every()`
- Objects: creation, updating, nested access
- JSON: parse/stringify

🔹 Scope & context

- Global vs local scope
- Block scope (`let` / `const`)
- The `this` keyword (how it changes with arrow functions)

🔹 ES6+ / Modern JS (very important)

- Destructuring
- Spread & rest `...`
- Template literals `` `Hello ${name}` ``
- Default parameters
- Modules: `import` / `export` (and CommonJS: `require`)
- Optional chaining `obj?.prop`
- Nullish coalescing `??`

🔹 Functions & patterns

- Closures
- Higher-order functions
- Callback functions
- Currying (advanced)
- Pure functions vs impure

🔹 Asynchronous JS

- Callbacks
- Promises
- `async` / `await`
- Fetch API basics
- Error handling in async code

🔹 Advanced topics

- Event loop & call stack
- Memory management & garbage collection (basic)
- Prototypes & inheritance
- Classes & constructor functions
- The `new` keyword
- `bind`, `call`, `apply`
- Symbol & iterator basics
- Generators (optional, advanced)
- Debouncing & throttling (frontend performance)

2️⃣ Extra topics for React (frontend)

- JSX syntax & why it works

- Component types: functional vs class components (React now prefers functional)
- Using arrow functions in props / event handlers
- Using `.map()` to render lists
- Conditional rendering:
  - `&&` operator
  - ternary expressions
- Updating immutable state: using spread/rest
- Event handling: `onClick={() => doSomething()}`
- Destructuring props & state
- Module system & bundlers (basic: Webpack, Vite)
- ES Modules (`import` / `export`) in depth
- Async/await for data fetching
- Fetch API / Axios
- LocalStorage / SessionStorage basics
- Memoization: `useMemo`, `useCallback` (React hooks)
- Understanding closures inside hooks (very important for `useEffect`)

---

3 Extra topics for backend (Node.js + Express)

- Node.js environment:
  - `global`, `__dirname`, `__filename`
  - `process` & environment variables
- Node built-in modules: `fs`, `path`, `os`, `crypto`, `http`
- CommonJS modules: `require`, `module.exports`
- EventEmitter basics
- Streams & buffers (files, uploads)
- Asynchronous patterns in Node:
  - Promises, async/await, callbacks
- Package management with npm / yarn
- Working with JSON & file system
- Error handling in async code
- Express.js:
  - Middleware functions & chaining
  - Routing & route parameters
  - Request & response objects (`req`, `res`)
  - Serving static files
  - Error handling middleware
- Connecting to databases (using drivers or ORMs)
- REST API patterns
- Structuring a Node project: modular folders, controllers, services

---

Summary:

Part	Topics
Core JS	Variables, data types, functions, scope, this, ES6+ (destructuring, spread), async (promises, async/await), modules, classes, advanced patterns
Frontend (React)	JSX, props, state, `.map()`, conditional rendering, immutable updates, events, hooks & closures, module bundlers
Backend (Node/Express)	Node APIs, CommonJS, streams, EventEmitter, middleware, routing, error handling, async patterns, file system

Part	Topics