Session	ClassName	Topics
1	Intro to JavaScript	What is JavaScript and its role Setting up environment (VS Code, Node.js) Variables (let) Data types (number, string, boolean) console.log() and comments Basic arithmetic operators (+, -, *, /, %)
2	Arithmetic Operators (Implementation)	Order of operations (PEMDAS) Compound assignments (+=, -=, *=, /=) Math object basics (Math.round, Math.floor, Math.ceil, toFixed, parseInt(basics), parseFloat, NaN, Infinity
3	Conditions - 1	Comparison operators (==, ===, !=, !==, <, >, <=, >=) Truthy and falsy values Logical operators (&&, ||, !) Short-circuit if, else if, else statements
4	Conditions - 2	if, else if, else statements Multiple if Nested If
5	Conditions - 3	Ternary and Switch
6	Loops - 1	Unary operators (++, --) Pre vs post increment for loop basics

7	Loops - 2	while loop basics
		do while Loop
		break and continue

8	Loops - 3	Loops Implementation and Problem solving Class

9	Functions	Function declaration vs expression Parameters and arguments Return statement Function calling/invocation Simple function examples (add, multiply, etc.)
10	Scope	Global scope Function scope Block scope (let vs var), TDZ Scope chain basics Variable shadowing Hoisting basic
11	Arrays - 1	Array declaration and initialization Accessing elements (indexing) length property Basic iteration with loops for of loop push(), pop(), shift, unshift, splice()

12	Arrays - 2	slice(), indexOf(), includes()
		concat(), reverse()

13	Arrays - 3	Shallow and Deep Copy Spread Operator structuredClone const with Array Array destructuring Array.flat()

14	String Basics	- What is a string - String literals (single, double, backticks) - Escape characters - Template literals and interpolation - length property - Accessing characters: [] and charAt() - Immutability concept - String concatenation (+) - Basic iteration (for loop , for of loop)
15	String Searching & Manipulation	- indexOf(), lastIndexOf() - includes() - startsWith(), endsWith() - slice() - - substring() - mention briefly - toUpperCase(), toLowerCase(), ASCII - replace(), replaceAll() - split() and join()
16	String Utilities	trim(), trimStart(), trimEnd() - padStart(), padEnd() - charCodeAt() , fromCharCode()
17	Objects - 1	Object literals Properties and methods Dot vs bracket notation Dynamic Keys Adding/deleting properties Object.keys(), Object.values()
18	Objects - 2	Nested objects Arrays of objects Object destructuring basics
19	Searching	Linear Search Problem on Linear Search
20	Sorting	.sort() Method sort menthod on Object Implementation

Session	ClassName	Topics
1	Nested Loops	Nested Loops with for Loop and While Loop Introduction to Star Pattern
2	Star Pattern	Star Pattern Problem Discussion
3	Star Pattern Day 2	Continue with Pattern based Question
3	2 D Matrix	2 D Matrix
4	Nested Arrays and String	2 D Matrix Nested Array Question (subArray)
5	Nested Arrays and String	Question Practice and Doubt Focus on Nested Array Arrays with Nested Loops String with Nested Loops
6	Time Complexity	Big O notation basics O(1), O(n), O(n^2) with examples Space complexity introduction Analyzing simple loops Best/worst/average case
7	Binary Search	Binary search concept Implementation on sorted arrays Finding boundaries Time complexity advantage Common variations
8	Recursion	Base case and recursive case Call stack visualization Simple examples: factorial, power Tracing recursive calls
9	Recursion	Basic Recursion Problems Array/string recursion Helper method recursion Pure recursion Fibonacci, reverse string

10	Recursion	Tree-like recursion Backtracking basics Permutations concept Recursion optimization (memoization intro)
11	Basic Sorting	Bubble sort Selection sort Insertion sort Time complexities
12	Merge Sort	Divide and conquer concept Merge function implementation mergeSort recursion Time/space complexity When to use
13	Quick Sort	Pivot selection Partition implementation quickSort recursion Best/worst cases In-place sorting
14	Callabck and Hofs	Functions as first-class citizens Callback pattern deep dive Creating higher-order functions Closure with callbacks
15	Map, Filter and Reduce	map() - transformation filter() - selection reduce() - aggregation Chaining methods

Session	ClassName	Topics
1	HTML Foundations	What is HTML and Why Do We Use It HTML Document Structure: <!DOCTYPE html>, <html>, <head>, <body> Using Headings: h1 to h6 Creating Subheadings with h2 Working with div as a Container Adding Text Content with p Tag Hyperlinks with Anchor Tag (a) and href Attribute Replacing Text Content in Elements Understanding Opening and Closing Tags Responsive and Semantic HTML Basics Using ID and Class Attributes Creating a Simple Webpage Layout Best Practices for Clean HTML Code

2	HTML in Practice	Creating a Webpage with Headings and Text Writing Paragraphs with h1, p, and b Elements Building a Semantic Navigation Bar with Links History Blog with Headings, Paragraphs, and Bold Text Adding Images Inside a Webpage Working with Ordered and Unordered Lists Using Custom Data Attributes (data-ns-test) Creating a Page with Title and Paragraph

3	HTML Tables & Forms 2	HTML Form Structure and Method Input Types: Text, Email, Number, Date, Password, URL Labels, IDs, and Required Fields Checkbox, Radio Buttons, and Select (Single/Multi) Submit Buttons and Form Submission Basic Form Styling with CSS HTML Table Structure: Rows, Columns, and Cells Responsive and Semantic HTML Basics Semantic Tables with thead, tbody, and tfoot Table Attributes: colspan, rowspan, classes, and IDs Size Chart Example: E-Commerce Style Table with thead and tbody Meta Tags and Their Purpose

4	GitHub Introduction	What is Git and Why Do We Need Version Control Difference Between Git and GitHub Git as a Decentralized Version Control System Initializing a Repository: git init Configuring User Details: git config (name and email) Tracking and Cleaning Files: git add, git commit, git clean Fetching and Cloning Repositories: git fetch vs git clone Committing Changes with Messages: git commit -m Tagging Commits with git tag

5	CSS Fundamentals and Bo	What is CSS and Why Do We Use It Ways to Add CSS: Inline, Internal, and External Selectors and Properties in CSS Colors, Fonts, and Text Styling Basics CSS Units: px, %, em, rem, vh, vw Understanding the Box Model: Content, Padding, Border, Margin Box Sizing: content-box vs border-box Display Property: block, inline, inline-block Positioning and Alignment Basics (static, relative, absolute, fixed) Practical Examples: Styling a Card Using Box Model CSS rulesets CSS variables

6	CSS Flexbox	What is Flexbox and Why Do We Use It Enabling Flexbox with display: flex Main Axis vs Cross Axis in Flexbox Flex Direction: row, row-reverse, column, column-reverse Justify Content: aligning items along the main axis Align Items: aligning items along the cross axis Align Self: overriding alignment for a single item Flex Wrap and Controlling Overflow Flex Grow, Flex Shrink, and Flex Basis (the flex shorthand) Practical Layouts with Flexbox: Navbars, Cards, and Grids

7	CSS Grid	What is CSS Grid and when to use it (vs Flexbox) Create a grid container: display: grid and basic properties Define tracks: grid-template-columns, grid-template-rows, and fr units Named layouts with grid-template-areas Place items using grid-column, grid-row, and grid-area Alignment & spacing: gap, justify-/align-items, and justify-/align-content Responsive grid techniques: minmax(), auto-fit/auto-fill, and media queries Styling grid children (size, padding, margin, background) — ties to Hello Ac Selectors, pseudo-classes and pseudo-elements (:nth-child, ::before, ::afte Hands-on exercises: submit button hover, block vs inline demo, and size-cl
8	ChatGPT Clone Project OF	Media Queries for Responsive Design — @media (max-width: 768px) Pseudo-Classes (:hover, :first-child, :nth-child) in Chat UI Pseudo-Elements (::before, ::after, ::first-line) for Styling Messages Styling the First Line of a Paragraph with p::first-line Using @keyframes for Typing Animations and Message Fade-In Background Styling with Linear and Radial Gradients (linear-gradient, radia Responsive Layout Techniques: Flexbox + Grid with Media Queries Mobile Optimization with <meta name="viewport" content="width=device-w

9	Amazing Designs	Introduction to Sass and Its Advantages over CSS Using Sass Variables for Colors, Sizes, and Fonts Nesting Selectors in Sass for Cleaner Code Creating and Using Mixins for Reusable Styles Generating Styles with Sass Loops (@for, @each) Optimizing CSS with Sass Functions (lighten, darken, etc.) Flexbox Layout Fundamentals (display: flex, alignment, flex-grow) Responsive Design with Media Queries Animations with @keyframes and transform Applying Transitions and Hover Effects for Interactivity Background Styling with Linear and Repeating Gradients Responsive Typography with clamp() and Relative Units Positioning Elements with Absolute and Relative Positioning Structuring Reusable Components with Modifier Classes

11	CSS Frameworks: Bootstra	Introduction to CSS Frameworks: Why Bootstrap and Tailwind Setting Up a Project with Bootstrap or Tailwind Typography Utilities: Headings, Font Sizes, Font Weights, Text Colors Spacing Utilities: Margin (m-, mx-, my-), Padding (p-, px-, py-) Flexbox Utilities: d-flex, flex, justify-content-*, items-center Grid System: Rows, Columns, and Responsive Breakpoints Backgrounds and Borders: Colors, Radius, Shadows, and Opacity Buttons and Links: Styling, Variants, and States (hover, active) Layout Helpers: container, max-w, w-full, h-screen Responsive Design: Breakpoints (sm:, md:, lg:) and Utility Responsiveness Building Components: Navbars, Cards, and Modals Exploring Documentation: How to Read & Apply Classes from Docs

Session	ClassName	Topics	Notes
1	DOM Basics & Element Selection	What is DOM - Document Object Model introduction Browser Developer Console basics document object exploration querySelector and querySelectorAll Selecting by ID: querySelector('#id') Selecting by class: querySelector('.class') Selecting by tag: querySelector('div') Selecting multiple elements: querySelectorAll('.class') CSS selector combinations: querySelector('div.class') Complex selectors practice Getting and Setting Content textContent - plain text manipulation innerHTML - HTML content manipulation Differences and when to use each Security considerations briefly Reading content from elements Basic Property Manipulation Changing element properties element.id, element.className Working with attributes getAttribute, setAttribute	Suggested Practice Projects: Interactive text manipulator Quote generator that changes existing elements About page where user can edit their info FAQ page with dynamic answers Product card information updater Key Learning Outcomes: Select any element using various methods Modify existing page content Understand difference between textContent and innerHTML

2	Event Basics & Simple Interactions	addEventListener Introduction Click event handling Why addEventListener over onclick Basic event handler functions Multiple listeners concept Building Interactive Elements State management with variables Updating DOM based on actions Working with numbers in DOM Increment/decrement patterns Random Numbers and Logic Math.random() for dynamic content Conditional rendering Basic game mechanics Score keeping patterns	Suggested Practice Projects: Counter with increment/decrement/reset Simple calculator with basic operations Dice rolling game Number guessing game Click reaction time tester Color switcher/theme changer Like button with count Voting system Key Learning Outcomes: Handle click events confidently Maintain state in JavaScript Update DOM based on user actions Create simple interactive features

3	Forms & Multi-User Interactions	Form Basics Getting input values Form submission prevention (preventDefault) Reading different input types Clearing inputs after use Working with Multiple Inputs Collecting user data Storing in arrays/objects Displaying user inputs Form validation basics Element States Enabling/disabling elements Showing/hiding elements Managing multiple UI states Visual feedback patterns Array Operations for UI Storing multiple entries Finding min/max values Filtering and sorting basics Displaying lists from arrays	Suggested Practice Projects: Multi-player competition game User registration system Survey/feedback form Guest book/comment system Team score tracker Raffle/lottery system Tournament bracket generator Expense splitter Key Learning Outcomes: Handle form submissions properly Manage multiple user inputs Work with arrays for data storage Control element states dynamically

4	Creating & Modifying DOM Elements	Creating Elements document.createElement() Setting properties on new elements Building elements before adding to DOM Creating complex element structures Adding Elements to DOM appendChild() - adding at the end append() vs appendChild() insertBefore() - specific position prepend() - adding at beginning Removing Elements element.remove() removeChild() method Clearing all children When to remove vs hide Dynamic List Management Adding items dynamically Removing specific items Clearing all items Counting elements	Suggested Practice Projects: Todo list application Shopping list manager Note-taking app Dynamic FAQ builder Comment system Playlist creator Contact list manager Recipe ingredient list Key Learning Outcomes: Create elements programmatically Add elements at specific positions Remove elements cleanly Build dynamic lists

5	Advanced Forms & Validation	Input Types Deep Dive Text, email, password, number inputs Textareas for longer content Date and time inputs File input basics Selection Controls Checkboxes (single and multiple) Radio button groups Select dropdowns (single/multiple) Getting selected values Form Validation Patterns Required field checking Length validation (min/max) Pattern matching (email/phone) Custom validation rules Real-time vs submit validation Validation Feedback Showing/hiding error messages Field-level vs form-level errors Submit button enable/disable Success confirmations Form Events 'input' event for real-time 'change' event 'blur' and 'focus' events	Suggested Practice Projects: Registration form with validation Multi-step form wizard Survey with different question types Password strength checker Booking/reservation form Job application form Product review form Settings/preferences panel Key Learning Outcomes: Handle all form input types Implement comprehensive validation Provide clear user feedback Create professional forms

6	DOM Traversal & Navigation	Parent Navigation parentElement property closest() method Finding ancestor elements Use cases for going up Child Navigation children vs childNodes firstElementChild, lastElementChild Accessing specific children Iterating through children Sibling Navigation nextElementSibling previousElementSibling Finding all siblings Practical sibling operations Complex Navigation Patterns Finding related elements Working with nested structures Table/list traversal Tree structure navigation	Suggested Practice Projects: Nested todo list with subtasks File explorer interface Organization chart navigator Nested comments/threads Table with row operations Menu with submenus Family tree visualizer Breadcrumb navigation Key Learning Outcomes: Navigate DOM tree in any direction Find related elements efficiently Work with complex nested structures Implement edit-in-place features

7	Objects & State Management	Why Objects? Problems They Solve Global variable pollution Organizing related data Namespace management Encapsulation benefits Object Literals Creating objects Properties and methods Nested objects Arrays of objects The 'this' Keyword 'this' in object methods Context and binding Common pitfalls Arrow functions vs regular functions State Management Patterns Single source of truth State object pattern Update and render cycle Configuration objects	Suggested Practice Projects: Multi-widget dashboard Game with player stats Shopping cart state Settings manager Multi-timer application Calendar with events Budget tracker Quiz game with scores Key Learning Outcomes: Use objects to organize code Manage application state Understand 'this' context Avoid global variable issues

8	Constructor Functions & Instances	Constructor Functions Creating object blueprints Naming conventions The 'new' keyword Constructor vs regular function Instance Properties Setting initial values Default parameters Each instance independence Property initialization Instance Methods Adding methods to instances Shared behavior patterns Method calling syntax Building reusable components Prototype Introduction (Brief) (Guide students to research on it) Prototype concept Adding shared methods Memory efficiency Prototype chain basics	Suggested Practice Projects: Multiple counter instances Timer/stopwatch system Notification manager Modal/popup factory Tab component system Slider/carousel creator Poll/voting widgets Alert/toast messages Key Learning Outcomes: Create reusable object blueprints Instantiate multiple independent objects Understand prototype basics Build component-like structures

9	ES6 Classes Fundamentals	Class Syntax Class declaration Constructor method Instance methods Class vs constructor function Properties and Methods Defining properties Method syntax Getters and setters Static methods basics Private Fields Private properties with # Encapsulation Public vs private When to use private Creating Instances Using new with classes Multiple instances Configuration options Connecting to DOM	Suggested Practice Projects: Accordion component class Modal/dialog class Dropdown menu class Tooltip system Tab navigation class Gallery component Form validator class Pagination component Key Learning Outcomes: Write modern ES6 classes Create reusable components Implement encapsulation Build UI component library

10	Event Delegation & Dynamic Content	Dynamic Content Problems Events on new elements Direct binding limitations Memory considerations Scalability issues Event Bubbling & Capturing Event propagation phases target vs currentTarget stopPropagation() Event flow understanding this Keyword Again: Call, Apply, Bind Delegation Pattern Parent listeners Target identification matches() method closest() for delegation Delegation Benefits Performance advantages Dynamic content handling Cleaner code Memory efficiency	Suggested Practice Projects: Dynamic shopping cart Product listing with filters Data table with actions Comment system with replies Gallery with dynamic images Task board with categories Chat interface Dynamic form builder Key Learning Outcomes: Implement event delegation properly Handle dynamic content events Optimize event handling Build scalable interfaces

Session	ClassName	Topics	Notes
1	Async Fundamentals & Timers	Project: Timer-based Applications Core Topics to Cover: Sync vs Async Introduction What is synchronous code? Blocking behavior demonstration Need for asynchronous programming Real-world examples setTimeout Deep Dive Basic syntax and usage Multiple setTimeout examples Execution order understanding clearTimeout and its importance Common use cases setInterval Mastery Continuous execution patterns Creating timers and clocks clearInterval usage Interval vs recursive setTimeout Memory considerations Practical Patterns Building countdown timers Slideshow creation	Suggested Practice Projects: Digital clock Countdown timer Quiz timer Auto-save feature Notification system Traffic light simulator Typing animation Session timeout Key Learning Outcomes: Understand async need Master setTimeout/setInterval Build timer-based features Handle cleanup properly

2	Callbacks & Callback Hell	Project: Multi-step Data Processor Core Topics to Cover: Understanding Callbacks Functions as arguments Callback execution flow Async callbacks with setTimeout Real-world callback examples Callback Patterns Error-first callbacks Success and error handlers Nested callbacks Sequential operations Callback Hell Problem The pyramid of doom Why it happens Readability issues Debugging difficulties Maintenance problems Managing Callback Hell Named functions approach Modular code structure Early returns Helper functions	Suggested Practice Projects: File processing simulator Multi-step form handler Sequential API caller Data validation pipeline User registration flow Order processing system Animation sequence Key Learning Outcomes: Write callback functions Understand callback hell Handle errors in callbacks Structure code better

3	Promises - The Solution	Project: Promise-based API Handler Core Topics to Cover: Promise Fundamentals What is a Promise? Three states: pending, fulfilled, rejected Promise constructor resolve and reject Using Promises .then() for success .catch() for errors .finally() for cleanup Promise chaining Callback to Promise Conversion Converting setTimeout to Promise Converting callback hell to promise chain Before and after comparison Promisify pattern Real Use Cases API call simulation Data loading Form submission	Suggested Practice Projects: Promise-based timer utilities Login system with promises Data fetcher with states Image preloader Sequential task runner Retry mechanism Cache system Key Learning Outcomes: Create and use Promises Convert callbacks to Promises Chain promises effectively Handle errors properly

4	Fetch API & Real APIs	Project: Complete API Integration Core Topics to Cover: Fetch API Basics Introduction to fetch() GET requests Understanding Response object Parsing JSON data Working with Real APIs Free APIs for practice Handling API responses Error handling Loading states POST, PUT, DELETE Sending data with fetch Headers configuration Body formatting Different content types Practical Implementation CORS basics API keys handling Network error handling Timeout implementation	Suggested Practice Projects: Weather app News aggregator GitHub user finder Currency converter Recipe finder Movie database Quote generator Todo API integration Key Learning Outcomes: Make API requests confidently Handle all HTTP methods Parse and display data Handle errors gracefully

5	Promise Methods & Parallel Operations	Project: Multi-Source Data Aggregator Core Topics to Cover: Promise.all() Parallel execution concept When to use Promise.all Handling multiple APIs Failure behavior Promise.race() First to complete wins Use cases (timeout, fastest server) Practical examples Promise.allSettled() Getting all results regardless Handling mixed success/failure When to use over Promise.all Promise.any Sequential vs Parallel Performance comparison When to use which Resource considerations Real-world scenarios	Suggested Practice Projects: Multi-API dashboard Bulk data processor Image gallery loader Parallel validator Server health checker Data migration tool Report generator Key Learning Outcomes: Use all Promise methods Optimize parallel operations Handle partial failures Choose right approach

6	Async/Await - Modern Approach	Project: Modern Async Application Core Topics to Cover: Async/Await Basics async keyword await keyword Syntactic sugar over Promises Making async look sync Error Handling try-catch blocks Error propagation Multiple error handling strategies finally blocks Converting Promise Code From .then() to await Maintaining same functionality When to use which approach Mixing promises and async/await Common Patterns Sequential await Parallel with Promise.all Conditional async operations Loop considerations	Suggested Practice Projects: User management system File upload handler Data sync service Report builder Backup system Migration script API wrapper library Key Learning Outcomes: Write clean async/await code Handle errors effectively Convert promise chains Avoid common pitfalls

7	Throttling & Debouncing	Project: Optimized Search Interface Core Topics to Cover: Understanding Throttling What is throttling? Limiting function calls Implementation with setTimeout Use cases (scroll, resize) Understanding Debouncing What is debouncing? Delay until stop Implementation pattern Use cases (search, save) Implementation Details Building from scratch Using clearTimeout Closure usage this context handling Real-World Applications Search optimization Form auto-save Scroll performance API call reduction	Suggested Practice Projects: Search with debounce Auto-save editor Infinite scroll Resize handler Real-time validation Analytics tracker Performance monitor Key Learning Outcomes: Implement throttle/debounce Optimize performance Reduce API calls Improve user experience

8	Execution Context & Call Stack	Project: Execution Visualizer Core Topics to Cover: Global Execution Context Creation phase Execution phase Global object and this Variable environment Function Execution Context Creation when function called Local variables Arguments object Scope chain Call Stack LIFO principle Stack frames Stack overflow Recursive functions Hoisting & Scope Variable hoisting Function hoisting Temporal dead zone Block scope vs function scope	Suggested Practice Projects: Call stack tracer Execution order predictor Scope analyzer Hoisting quiz Context debugger Memory visualizer Key Learning Outcomes: Understand execution flow Predict code execution Debug scope issues Understand hoisting

9	Event Loop Deep Dive	Project: Event Loop Simulator Core Topics to Cover: Event Loop Mechanism Single-threaded nature Call stack review Web APIs / Node APIs Callback queue Microtasks vs Macrotasks Task queue (setTimeout, setInterval) Microtask queue (Promises) Execution priority Real examples Complete Flow Code execution path When callbacks execute Promise execution timing Starvation scenarios Practical Implications UI blocking Performance optimization Debugging async code Common gotchas	Suggested Practice Projects: Event loop visualizer Execution order quiz Performance analyzer Task scheduler Priority queue system Animation coordinator Key Learning Outcomes: Understand event loop completely Predict execution order Debug timing issues Optimize performance

10	Essentail Concept before React	Part 1: ES6 Modules (1 hour) - export/import syntax - Named vs default - Organizing projects Part 2: NPM Ecosystem (1 hour) - npm init, install - package.json - node_modules - NPX for tools axios with and without NPM. Part 3: Simple Build Setup (1 hour) - Parcel or Vite - Dev server - Building project - Deploying	
11	Final Project	Chose One project from Doc or Equivalent Project You are free to chose your own project but in that case do prepare a doc of your project and share it to learningteam@acciojob.com so we could review it. Do one project completely end to end (100%) in class , and students will chose another project from this sheet to do on their own, do share the list with students.	Async Module Project:

Session	ClassName	Topics	Notes
1	SQL Basics	Core Topics: Database concepts, tables, relationships Basic SQL: SELECT, INSERT, UPDATE, DELETE WHERE, ORDER BY, LIMIT Data types and constraints	
2	SQL Joins & Relationships	Primary/Foreign keys INNER JOIN, LEFT JOIN GROUP BY, HAVING Aggregate functions (COUNT, SUM, AVG) Simple subqueries	
3	MongoDB Fundamentals	NoSQL concepts MongoDB CRUD operations Basic queries and filtering When to use SQL vs NoSQL Simple data modeling	
4	Express.js Fundamentals	Core Topics: What is Express? Creating a server Routing basics (GET, POST) Request/Response cycle Middleware concept Cors	Use Javascript to Call these apis: Suggested Projects: Calculator API Welcome API Info API Mock data API
5	Express CRUD - In Memory	Core Topics: RESTful principles CRUD operations (no database) Request body, params, query Status codes Error handling basics Postman testing	Use Javascript to Call these apis: Suggested Projects: Todo API (array) Notes API Contacts API Shopping cart API

6	React Setup & JSX	Core Topics: React with Vite setup JSX in depth Components introduction Rendering elements Project structure	Suggested Projects: Portfolio header Card components Navigation bar Hero section
7	Components & Props	Core Topics: Functional components Props in detail Children props Prop destructuring Default props	Suggested Projects: Product cards User profiles Blog previews Testimonials Gallery items
8	State Basics (useState)	useState introduction State with primitives State With Object Event handling Forms with input	Suggested Projects: Counter variations Toggle switches Color picker Temperature converter Tabs component
9	Complex State Management	Complex State Update Multiple states Controlled and Uncontrolled	Suggested Projects: Todo list Contact manager Shopping list Expense tracker Quiz application

10	Lifting State Up	Lifting state pattern Parent-child communication Sibling components Props vs State Data flow	Suggested Projects: Shopping cart Seat selector Multi-step form Filter system Game scoreboard
11	useEffect Basics	useEffect introduction Dependency array Common use cases API calls introduction Loading states	Suggested Projects: Timer component Data fetcher Window resize tracker Local storage sync Title updater
12	useEffect Advanced	Core Topics: Cleanup functions Multiple useEffects Race conditions Async in useEffect Common pitfalls	Suggested Projects: Countdown timer with cleanup Search with debounce Auto-save form Clock with cleanup API call cancellation

13	Project Sessions	Minor Project	Project Options: E-commerce Product Page Product gallery with image selection Size/color options Add to cart functionality Cart sidebar with items Total calculation All in one page, no routing Task Management Dashboard Add/edit/delete tasks Filter by status Search functionality Drag between columns (without library) All state management Restaurant Menu & Order System Menu with categories Add items to order Quantity management Order summary Total with tax calculation

14	React Router	Core Topics: React Router setup Routes and Links Dynamic routing useParams, useNavigate Nested routes Protected routes basics	Suggested Projects: Portfolio site Blog with pages E-commerce routing Dashboard layout Documentation site
15	Context API Basics	Prop drilling problem Creating context Provider pattern useContext hook When to use context	Suggested Projects: Theme switcher Language selector User context Settings manager
16	Context API Advanced	Multiple contexts Context composition Performance considerations Context vs props useRef	Suggested Projects: Auth context Cart context Notification system Modal manager Form context
17	Use Ref	useRef for DOM access useRef for persistent values When to use useRef vs useState Common useRef patterns	UseRef Part: Video player controls Form focus manager Scroll controller Canvas drawing Stopwatch

18	Authentication Project - Frontend	Full Project Session Build Complete Auth UI: Login/Signup forms Password validation Remember me checkbox Forgot password flow Protected routes Profile page Logout functionality Using Context for auth state Using useRef for form focus Use Localstorage to Store Token API: Mock API provided and localStorage	
19	Basic Backend Authentication	User registration endpoint Password hashing with bcrypt Login endpoint Simple token generation (UUID) Verify token middleware Protected routes Connect to frontend project	
20	Cookies Introduction	Problems with simple tokens (can't verify, no expiry) What is JWT - self-contained tokens JWT structure explained Replace simple token with JWT Add expiration Verify without database lookup	

21	Sessions & Advanced Auth	Problems with localStorage (XSS) What are cookies? express cookie-parser Convert token to cookie storage httpOnly flag Cookie vs localStorage comparison Update frontend to work with cookies CORS with credentials	
22	JWT Implementation	Problems with simple tokens JWT structure and benefits Replace UUID with JWT Token expiration JWT in cookies Verify JWT middleware Update auth project with JWT	Skipping: Passport and Social Logins , Firebase , Clerk etc We could share a youtube video if they want to explore them.
23	File Uploads & Email	Topics: Multer for file uploads Profile picture upload Image validation (size, type) Storing image paths/URLs Basic email sending (nodemailer) Welcome email after registration Simple email templates	Practice (45 min): Add profile picture to auth project Send welcome email on signup and Email Validation
24	Remaining React Topics & Advanced Patterns	useReducer deep dive useMemo for expensive calculations useCallback for function optimization React.lazy and Suspense Error Boundaries Code splitting basics forwardRef (briefly)	Practice: Optimize existing components Implement useReducer for complex state
25	Project Day 1	Frontend Of Projects	
26	Project Day 2	Frontend Of Projects	
27	Project Day 3	Backend of Project	

28	Project Day 4	Backend of Projects	
29	Redux Core Concepts	Redux in vanilla JS Store, actions, reducers Subscribe, dispatch Pure Redux examples	Counter with History Increment/decrement Undo/redo functionality Action log display Todo List (No UI) Console-based Add, remove, toggle Filter actions Shopping Cart Add items Update quantities Calculate total Apply discount

30	Redux with React	React-Redux integration useSelector, useDispatch Provider setup Same examples in React	Counter with History + UI Visual counter History timeline Time travel debugging Todo App with Filters Full UI Multiple filters Stats display Shopping Cart with UI Product cards Cart sidebar Real-time total

31	Redux Toolkit	Modern Redux with RTK Slices Simplified syntax Better developer experience	Advanced Counter Multiple counters with RTK Async increment (setTimeout) Step configuration Todo with Categories createSlice for todos createSlice for categories Related data management Mini E-commerce Products slice Cart slice Wishlist slice Filter slice

32	Redux Toolkit With Apis	createAsyncThunk deep dive Handling loading, success, error states Extra reducers API call patterns Error handling in Redux	Mini Projects: Weather App with Redux Fetch weather data Loading states Error handling Cache results User Search with Redux GitHub API integration Search with debounce Pagination Loading indicators News Feed Fetch articles Infinite scroll Categories Refresh functionality
33	Apply Redux to Project	Replace Context with Redux	
34	Node.js Core Modules	File System (fs) - read, write, streams Path module for file paths Crypto for hashing/encryption OS module for system info Readline for CLI inputs URL and querystring modules	Mini Projects: File manager CLI tool Log analyzer Bulk file processor CSV reader/writer

35	Advanced Node.js	Streams and pipes in depth Child processes (spawn, exec, fork) Clusters for multi-core Event emitters pattern Worker threads basics	Mini Projects: Video processing with child process Multi-core server Custom event system Stream-based file upload
36	Backend Jobs & Scheduling	Cron jobs with node-cron Background job processing Queue systems (Bull basics) Scheduled tasks (newsletters, cleanup) Long-running tasks handling	Mini Projects: Email newsletter scheduler Database backup automation Expired data cleanup Report generation system Scoket io and chat app => Youtube video recsource if any one needs it
		Provide YouTube videos for: for those who want to explore: TypeScript basics Testing introduction Next.js overview GraphQL basics Docker basics Socket.io	