Admissions & Support: support@mockattempt.com
Phone / WhatsApp: +91 95400 07917
Course Code: NFSC • **Updated:** 04 Nov 2026
https://mockattempt.com/training/nodejs-full-stack-engineering-program

NFSC • FULL STACK DEVELOPMENT **BEGINNER TO EXPERT • JOB-GUARANTEED***

100% PLACEMENT GUARANTEED* CAREER PROGRAM (6-MO REFUND*)

Node.js, TypeScript & Modern Full Stack Engineering (MERN / PERN)

Complete Zero-to-Expert HTML5, CSS3, Core JS, TypeScript 5 & DSA to React 19, Next.js 15, Node.js, NestJS, Dual DB (PostgreSQL + MongoDB), Redis, Docker & AWS | 100% Placement Guaranteed* (6 Months Fees Refund*)

Duration: 28w (420h) • Weekend Only (2h/Day) (448 Hrs) • **Modules:** 22 • **Live Topics:** 86 • **Subtopics:** 688+ • **Labs & Tasks:** 172 • **Mocks / AI Vivas:** 172 / 172 • **Projects:** 22 Module + 1 Live Client Prod

All-Inclusive Direct Fee (Incl. 18% GST • Zero Hidden Charges): Total Payable: ₹29,999 (Net Tuition ₹25,423 + Included 18% GST ₹4,576) • **Split Option:** ₹4,500/mo × 6 (0% Split) • **Effective ROI:** ~₹67/Guided Hr (~₹349/Live Topic)

Faculty: 100% Active Working IT Professionals • **Weekend-Only Live Batch (7:00 AM – 8:00 PM IST Flexible Slots):** Weekend-Only Career Live Batch (Sat – Sun • 2 Hrs/Day • 7:00 AM – 8:00 PM) (Saturday & Sunday Only (2 Hours Live Session / Day) • 07:00 AM – 08:00 PM IST (Students Can Choose Any 2-Hour Weekend Slot Based on Working Professional Trainer Availability) • 28 Weeks (448 Guided Live + Lab Hours))

✓ **Trainer LeetCode & HackerRank Mentorship (JavaScript ES6+, TypeScript 5 & SQL): Live trainer support to crack LeetCode & HackerRank problems, optimize complexity, and handle edge cases.**

✓ **6-Month Placement Guarantee* + Online AI Assessment (OA) Support: Complete 100% assigned repositories & score ≥90% across all Topic Mock Tests & Voice AI Interviews → 100+ verified IT job openings within 6 months with active Senior Trainer coaching to crack employer Online AI Assessments (OA), or 100% net tuition refund*.**

Core Technologies & Engineering Stack: HTML5, CSS3 & Core JS From Zero • TypeScript 5 & JS/TS DSA • React 19 & Next.js 15 App Router • Node.js 22, Express & NestJS • SQL From Scratch & PostgreSQL 16 • Prisma / Drizzle Type-Safe ORMs • MongoDB 7 & Aggregations • Redis 7, BullMQ & Socket.IO • JWT / OAuth2 & OWASP Security • Vitest, Testcontainers & Playwright • Docker, Nginx & AWS Cloud • GitHub Actions CI/CD & System Design

1. PROGRAM MASTERY OUTCOMES & ELIGIBILITY

End-to-End Engineering Outcomes:

1. Start from Day-1 zero coding experience and master every foundation, framework, database, cloud, and system design layer up to expert level
2. Build and ship 22 real-world production GitHub repositories + 1 Live Client Production Cloud Deployment (zero dummy to-do apps)
3. Achieve ≥90% mastery across every Topic Mock Test and Spoken MockAttempt AI Technical Interview (unlimited free re-attempts allowed below 90%)
4. Earn Dual Barcode-Verifiable Credentials: Course Excellence Certificate + Official Project Internship Confirmation Letter
5. Unlock 6-Month Placement Support guaranteeing 100 curated relevant IT job openings — backed by our 100% Tuition Fee Refund* policy
6. Crack LeetCode and HackerRank problems in JavaScript ES6+, TypeScript 5 & SQL with live trainer mentorship on algorithmic/query patterns, edge cases, and optimal complexity.
7. Receive active Senior Trainer coaching during your 6-month (180-day) Placement Tenure to crack employer Online AI Assessments (OA), timed coding rounds, and automated screening tests.
8. End-to-end strict TypeScript 5 across React 19, Next.js 15 App Router (SSR/RSC), Express, and enterprise NestJS modules.

Eligibility & Prerequisites: Zero prior programming or IT experience required — Module 1 starts from Day-1 setup, logic building, and plain-English fundamentals • Basic computer literacy and commitment to attend Weekend-Only Live Sessions (Saturday & Sunday, 2 Hours/Day between 7:00 AM – 8:00 PM IST taught by active Working IT Professionals) plus guided weekday real-repository sprints • Dedication to complete 100% of assigned tasks and re-attempt assessments until crossing ≥90% mastery • Ideal for B.Tech / BCA / MCA / B.Sc IT students, non-CS beginners, career switchers, and working developers upgrading to 2026 production standards. • Dedication to write original code in your GitHub portfolio and complete the mentor-guided Live Client Production Sprint.

2. DETAILED ZERO-TO-EXPERT MODULE, TOPIC & SUBTOPIC SYLLABUS (22 MODULES • 86 LIVE TOPICS • 688+ SUBTOPICS)

Progressive Zero-to-Expert Mastery Stages: Modules 1–5: Stage 1: Day-1 Foundations & Core Mental Models (20 Topics / 160 Subtopics) • Modules 6–12: Stage 2: Applied Engineering, Data & API Workflows (28 Topics / 224 Subtopics) • Modules 13–17: Stage 3: Internals, Concurrency, Security & Performance (20 Topics / 160 Subtopics) • Modules 18–22: Stage 4: Cloud Scale, System Design & ≥90% Placement Gate (18 Topics / 144 Subtopics)

MODULE 1 OF 22 • STAGE 1 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Web Foundations From Scratch: Internet Protocols, Linux CLI, Git & Semantic HTML5

Beginner-to-expert real-repository module covering Web Foundations From Scratch: Internet Protocols, Linux CLI, Git & Semantic HTML5 with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ How the Internet Works (DNS, TCP/IP, TLS, HTTP/1.1 vs HTTP/2/3), VS Code Setup & Browser DevTools — Purpose: Inspect live network waterfalls, headers, and status codes like a professional web engineer from Day 1. • ✓ Linux Terminal Navigation, Shell Essentials & Professional Git / GitHub Branching & PR Workflows — Purpose: Manage clean feature branches, resolve merge conflicts calmly, and maintain recruiter-ready GitHub repositories. • ✓ Semantic HTML5 Document Architecture, Accessible Forms, Native Validation, Tables & Media Elements — Purpose: Structure clean, screen-reader-friendly, SEO-semantic web pages without div-soup anti-patterns. • ✓ WCAG 2.2 Accessibility Landmarks, ARIA Attributes, Technical SEO Meta Tags & Core Web Vitals Basics — Purpose: Build inclusive web foundations that rank strongly on Google and work seamlessly with keyboard navigation.

1.1 How the Internet Works (DNS, TCP/IP, TLS, HTTP/1.1 vs HTTP/2/3), VS Code Setup & Browser DevTools

[90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for How the Internet Works (DNS, TCP/IP, TLS, HTTP/1.1 v... • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply How the Internet Works (DNS, TCP/IP, TLS, HTTP/1.1 vs HTTP/2/3), VS Code Setup & Browser DevTools in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Next Topic 1.2: Linux Terminal Navigation, Shell Essentials & Professional Git / GitHub Branching & PR Workflows

1.2 Linux Terminal Navigation, Shell Essentials & Professional Git / GitHub Branching & PR Workflows

[90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Linux Terminal Navigation • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Linux Terminal Navigation, Shell Essentials & Professional Git / GitHub Branching & PR Workflows in real Node.js, TypeScript & Modern Full Stack E...

What Topic Is Next → Next Topic 1.3: Semantic HTML5 Document Architecture, Accessible Forms, Native Validation, Tables & Media Elements

1.3 Semantic HTML5 Document Architecture, Accessible Forms, Native Validation, Tables & Media Elements

[90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Semantic HTML5 Document Architecture • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Semantic HTML5 Document Architecture, Accessible Forms, Native Validation, Tables & Media Elements in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Next Topic 1.4: WCAG 2.2 Accessibility Landmarks, ARIA Attributes, Technical SEO Meta Tags & Core Web Vitals Basics

1.4 WCAG 2.2 Accessibility Landmarks, ARIA Attributes, Technical SEO Meta Tags & Core Web Vitals Basics

[90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for WCAG 2.2 Accessibility Landmarks • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply WCAG 2.2 Accessibility Landmarks, ARIA Attributes, Technical SEO Meta Tags & Core Web Vitals Basics in real Node.js, TypeScript & Modern Full Stac...

What Topic Is Next → Module 1 Capstone (scaleflow-m01-web-html5-git-foundations) → Next Topic 2.1: CSS3 Cascade, Specificity, Box Model, Custom Properties (CSS Variables) & Modern Viewport Units

Module 1 Hands-On Capstone Project (100 Marks): scaleflow-m01-web-html5-git-foundations — Architect, implement, test, and push the production GitHub repository 'scaleflow-m01-web-html5-git-foundations' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 1...

MODULE 2 OF 22 • STAGE 1 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Modern CSS3 Mastery, Flexbox, CSS Grid, Animations & Tailwind CSS Systems

Beginner-to-expert real-repository module covering Modern CSS3 Mastery, Flexbox, CSS Grid, Animations & Tailwind CSS Systems with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ CSS3 Cascade, Specificity, Box Model, Custom Properties (CSS Variables) & Modern Viewport Units — Purpose: Eliminate layout guesswork and specificity wars using predictable CSS custom tokens and box sizing. • ✓ 1D & 2D Responsive Layouts with Flexbox, CSS Grid, Container Queries & Mobile-First Media Queries — Purpose: Construct complex responsive dashboards, Bento grids, and adaptive navigation bars across all screen sizes. • ✓ Micro-Interactions, GPU-Accelerated Transitions, Keyframe Animations & Light / Dark Theme Tokens — Purpose: Deliver smooth 60fps UI polish without triggering expensive browser layout thrashing. • ✓ Utility-First Tailwind CSS Architecture, Reusable Design Tokens & Accessible Component Primitives — Purpose: Build cohesive, enterprise-grade SaaS UI kits 3x faster using structured Tailwind utility patterns.

2.1 CSS3 Cascade, Specificity, Box Model, Custom Properties (CSS Variables) & Modern Viewport Units [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for CSS3 Cascade • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply CSS3 Cascade, Specificity, Box Model, Custom Properties (CSS Variables) & Modern Viewport Units in real Node.js, TypeScript & Modern Full Stack En...

What Topic Is Next → Next Topic 2.2: 1D & 2D Responsive Layouts with Flexbox, CSS Grid, Container Queries & Mobile-First Media Queries

2.2 1D & 2D Responsive Layouts with Flexbox, CSS Grid, Container Queries & Mobile-First Media Queries [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for 2D Responsive Layouts with Flexbox • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply 1D & 2D Responsive Layouts with Flexbox, CSS Grid, Container Queries & Mobile-First Media Queries in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Next Topic 2.3: Micro-Interactions, GPU-Accelerated Transitions, Keyframe Animations & Light / Dark Theme Tokens

2.3 Micro-Interactions, GPU-Accelerated Transitions, Keyframe Animations & Light / Dark Theme Tokens [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Micro-Interactions • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Micro-Interactions, GPU-Accelerated Transitions, Keyframe Animations & Light / Dark Theme Tokens in real Node.js, TypeScript & Modern Full Stack E...

What Topic Is Next → Next Topic 2.4: Utility-First Tailwind CSS Architecture, Reusable Design Tokens & Accessible Component Primitives

2.4 Utility-First Tailwind CSS Architecture, Reusable Design Tokens & Accessible Component Primitives [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Utility-First Tailwind CSS Architecture • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Utility-First Tailwind CSS Architecture, Reusable Design Tokens & Accessible Component Primitives in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Module 2 Capstone (scaleflow-m02-css3-tailwind-design-system) → Next Topic 3.1: JavaScript Values, Types, Coercion Rules, let/const Temporal Dead Zone, Operators & Control Flow

Module 2 Hands-On Capstone Project (100 Marks): scaleflow-m02-css3-tailwind-design-system — Architect, implement, test, and push the production GitHub repository 'scaleflow-m02-css3-tailwind-design-system' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module...

MODULE 3 OF 22 • STAGE 1 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Core JavaScript From Zero: Syntax, Functions, Closures, Objects & DOM Events

Beginner-to-expert real-repository module covering Core JavaScript From Zero: Syntax, Functions, Closures, Objects & DOM Events with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ JavaScript Values, Types, Coercion Rules, let/const Temporal Dead Zone, Operators & Control Flow — Purpose: Avoid classic JavaScript equality and coercion bugs by mastering strict primitive and reference semantics. • ✓ Functions, Lexical Scope, Closures, Higher-Order Functions & Functional Array Pipelines (map/filter/reduce) — Purpose: Encapsulate private state with closures and transform complex API datasets declaratively without mutation bugs. • ✓ Objects, Prototypal Inheritance, ES6 Classes, this Binding Rules (call/apply/bind), Destructuring & Modules — Purpose: Master JavaScript's object model and organize multi-file ES Module applications cleanly. • ✓ DOM Tree Manipulation, Event Bubbling / Delegation, Debounce / Throttle, Web Storage & Form Handling — Purpose: Build fast interactive client experiences with zero memory leaks or laggy scroll/input handlers.

3.1 JavaScript Values, Types, Coercion Rules, let/const Temporal Dead Zone, Operators & Control Flow [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... •

Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for JavaScript Values • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply JavaScript Values, Types, Coercion Rules, let/const Temporal Dead Zone, Operators & Control Flow in real Node.js, TypeScript & Modern Full Stack E...

What Topic Is Next → Next Topic 3.2: Functions, Lexical Scope, Closures, Higher-Order Functions & Functional Array Pipelines (map/filter/reduce)

3.2 Functions, Lexical Scope, Closures, Higher-Order Functions & Functional Array Pipelines (map/filter/reduce)

[90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Functions • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Functions, Lexical Scope, Closures, Higher-Order Functions & Functional Array Pipelines (map/filter/reduce) in real Node.js, TypeScript & Modern F...

What Topic Is Next → Next Topic 3.3: Objects, Prototypal Inheritance, ES6 Classes, this Binding Rules (call/apply/bind), Destructuring & Modules

3.3 Objects, Prototypal Inheritance, ES6 Classes, this Binding Rules (call/apply/bind), Destructuring & Modules

[90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Objects • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Objects, Prototypal Inheritance, ES6 Classes, this Binding Rules (call/apply/bind), Destructuring & Modules in real Node.js, TypeScript & Modern F...

What Topic Is Next → Next Topic 3.4: DOM Tree Manipulation, Event Bubbling / Delegation, Debounce / Throttle, Web Storage & Form Handling

3.4 DOM Tree Manipulation, Event Bubbling / Delegation, Debounce / Throttle, Web Storage & Form Handling

[90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for DOM Tree Manipulation • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply DOM Tree Manipulation, Event Bubbling / Delegation, Debounce / Throttle, Web Storage & Form Handling in real Node.js, TypeScript & Modern Full Sta...

What Topic Is Next → Module 3 Capstone (scaleflow-m03-core-javascript-dom) → Next Topic 4.1: V8 Call Stack, Heap, Web APIs, Macrotask Queue vs Microtask Queue & Event Loop Execution Order

Module 3 Hands-On Capstone Project (100 Marks): scaleflow-m03-core-javascript-dom — Architect, implement, test, and push the production GitHub repository 'scaleflow-m03-core-javascript-dom' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 3: Core...

MODULE 4 OF 22 • STAGE 1 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Asynchronous JavaScript, V8 Event Loop, Fetch API & Browser Performance

Beginner-to-expert real-repository module covering Asynchronous JavaScript, V8 Event Loop, Fetch API & Browser Performance with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ V8 Call Stack, Heap, Web APIs, Macrotask Queue vs Microtask Queue & Event Loop Execution Order — Purpose: Predict and debug complex async execution timing and prevent main-thread UI freezes. • ✓ Promises From Scratch: Chaining, Error Propagation, Promise.all / allSettled / race & async/await — Purpose: Orchestrate parallel and sequential async workflows with clean cancellation (AbortController) and retries. • ✓ REST API Integration with Fetch, Headers, CORS Preflight, Streaming Responses & Resilient Error UI — Purpose: Consume live HTTP APIs defensively with timeouts, structured error recovery, and skeleton loading states. • ✓ ES2024+ Features, Map/Set/WeakMap, RegExp, Intl Formatting (INR/UTC) & Client Memory Leak Profiling — Purpose: Format currencies/dates accurately and hunt detached DOM nodes using Chrome Memory DevTools.

4.1 V8 Call Stack, Heap, Web APIs, Macrotask Queue vs Microtask Queue & Event Loop Execution Order

[90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for V8 Call Stack • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply V8 Call Stack, Heap, Web APIs, Macrotask Queue vs Microtask Queue & Event Loop Execution Order in real Node.js, TypeScript & Modern Full Stack Eng...

What Topic Is Next → Next Topic 4.2: Promises From Scratch: Chaining, Error Propagation, Promise.all / allSettled / race & async/await

4.2 Promises From Scratch: Chaining, Error Propagation, Promise.all / allSettled / race & async/await

[90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Promises From Scratch • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Promises From Scratch: Chaining, Error Propagation, Promise.all / allSettled / race & async/await in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Next Topic 4.3: REST API Integration with Fetch, Headers, CORS Preflight, Streaming Responses & Resilient Error UI

4.3 REST API Integration with Fetch, Headers, CORS Preflight, Streaming Responses & Resilient Error UI [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for REST API Integration with Fetch • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply REST API Integration with Fetch, Headers, CORS Preflight, Streaming Responses & Resilient Error UI in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Next Topic 4.4: ES2024+ Features, Map/Set/WeakMap, RegExp, Intl Formatting (INR/UTC) & Client Memory Leak Profiling

4.4 ES2024+ Features, Map/Set/WeakMap, RegExp, Intl Formatting (INR/UTC) & Client Memory Leak Profiling [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for ES2024+ Features • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply ES2024+ Features, Map/Set/WeakMap, RegExp, Intl Formatting (INR/UTC) & Client Memory Leak Profiling in real Node.js, TypeScript & Modern Full Stac...

What Topic Is Next → Module 4 Capstone (scaleflow-m04-async-js-v8-runtime) → Next Topic 5.1: TypeScript 5 Compiler Config (strict: true), Structural Typing, Interfaces, Tuples, Enums & Type Narrowing

Module 4 Hands-On Capstone Project (100 Marks): scaleflow-m04-async-js-v8-runtime — Architect, implement, test, and push the production GitHub repository 'scaleflow-m04-async-js-v8-runtime' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 4: Asyn...

MODULE 5 OF 22 • STAGE 1 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Strict TypeScript 5 Mastery & Data Structures / Algorithms (DSA) in TypeScript

Beginner-to-expert real-repository module covering Strict TypeScript 5 Mastery & Data Structures / Algorithms (DSA) in TypeScript with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ TypeScript 5 Compiler Config (strict: true), Structural Typing, Interfaces, Tuples, Enums & Type Narrowing — Purpose: Eradicate undefined-is-not-a-function runtime crashes across frontend and backend codebases. • ✓ Advanced Generics, keyof / typeof, Mapped & Conditional Types, Utility Types & Zod Schema Inference — Purpose: Design self-validating end-to-end API contracts where runtime Zod schemas generate static TypeScript types. • ✓ DSA Part 1 in TypeScript: Big-O Analysis, Two Pointers, Sliding Window, Stacks, Queues & Hash Maps — Purpose: Solve array, string, interval, and caching interview problems in clean, strongly typed TypeScript. • ✓ DSA Part 2 in TypeScript: Linked Lists, Binary Trees, Heaps, Tries, Graph BFS/DFS & Dynamic Programming — Purpose: Clear product-startup and MNC coding rounds with solid algorithmic mastery in your primary stack language.

5.1 TypeScript 5 Compiler Config (strict: true), Structural Typing, Interfaces, Tuples, Enums & Type Narrowing [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for TypeScript 5 Compiler Config (strict: true) • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply TypeScript 5 Compiler Config (strict: true), Structural Typing, Interfaces, Tuples, Enums & Type Narrowing in real Node.js, TypeScript & Modern Fu...

What Topic Is Next → Next Topic 5.2: Advanced Generics, keyof / typeof, Mapped & Conditional Types, Utility Types & Zod Schema Inference

5.2 Advanced Generics, keyof / typeof, Mapped & Conditional Types, Utility Types & Zod Schema Inference [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Advanced Generics • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Advanced Generics, keyof / typeof, Mapped & Conditional Types, Utility Types & Zod Schema Inference in real Node.js, TypeScript & Modern Full Stac...

What Topic Is Next → Next Topic 5.3: DSA Part 1 in TypeScript: Big-O Analysis, Two Pointers, Sliding Window, Stacks, Queues & Hash Maps

5.3 DSA Part 1 in TypeScript: Big-O Analysis, Two Pointers, Sliding Window, Stacks, Queues & Hash Maps [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for DSA Part 1 in TypeScript • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply DSA Part 1 in TypeScript: Big-O Analysis, Two Pointers, Sliding Window, Stacks, Queues & Hash Maps in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Next Topic 5.4: DSA Part 2 in TypeScript: Linked Lists, Binary Trees, Heaps, Tries, Graph BFS/DFS & Dynamic Programming

5.4 DSA Part 2 in TypeScript: Linked Lists, Binary Trees, Heaps, Tries, Graph BFS/DFS & Dynamic Programming [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for DSA Part 2 in TypeScript • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply DSA Part 2 in TypeScript: Linked Lists, Binary Trees, Heaps, Tries, Graph BFS/DFS & Dynamic Programming in real Node.js, TypeScript & Modern Full...

What Topic Is Next → Module 5 Capstone (scaleflow-m05-typescript5-dsa-core) → Next Topic 6.1: React 19 Rendering Mental Model, Virtual DOM Reconciliation, JSX, Props & One-Way Data Flow

Module 5 Hands-On Capstone Project (100 Marks): scaleflow-m05-typescript5-dsa-core — Architect, implement, test, and push the production GitHub repository 'scaleflow-m05-typescript5-dsa-core' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 5: Str...

MODULE 6 OF 22 • STAGE 2 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

React 19 Fundamentals, Hooks Internals, Component Patterns & Forms

Beginner-to-expert real-repository module covering React 19 Fundamentals, Hooks Internals, Component Patterns & Forms with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ React 19 Rendering Mental Model, Virtual DOM Reconciliation, JSX, Props & One-Way Data Flow — Purpose: Think in declarative UI state machines rather than brittle imperative DOM mutations. • ✓ Core Hooks Deep Dive: useState, useReducer, useEffect Cleanup Rules, useRef, useId & Custom Hooks — Purpose: Extract complex stateful logic into clean, testable custom hooks without stale-closure or infinite-loop bugs. • ✓ Compound Components, Controlled vs Uncontrolled Inputs, React Hook Form + Zod & Accessible Modals — Purpose: Build multi-step enterprise onboarding and checkout forms with instant accessible validation feedback. • ✓ React Rendering Performance: Profiler Flamegraphs, Virtualized Lists, Code Splitting & Suspense Boundaries — Purpose: Keep 10,000-row tables and rich analytics dashboards silky smooth on mid-range mobile devices.

6.1 React 19 Rendering Mental Model, Virtual DOM Reconciliation, JSX, Props & One-Way Data Flow [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for React 19 Rendering Mental Model • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply React 19 Rendering Mental Model, Virtual DOM Reconciliation, JSX, Props & One-Way Data Flow in real Node.js, TypeScript & Modern Full Stack Engine...

What Topic Is Next → Next Topic 6.2: Core Hooks Deep Dive: useState, useReducer, useEffect Cleanup Rules, useRef, useId & Custom Hooks

6.2 Core Hooks Deep Dive: useState, useReducer, useEffect Cleanup Rules, useRef, useId & Custom Hooks [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Core Hooks Deep Dive • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Core Hooks Deep Dive: useState, useReducer, useEffect Cleanup Rules, useRef, useId & Custom Hooks in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Next Topic 6.3: Compound Components, Controlled vs Uncontrolled Inputs, React Hook Form + Zod & Accessible Modals

6.3 Compound Components, Controlled vs Uncontrolled Inputs, React Hook Form + Zod & Accessible Modals [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Compound Components • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Compound Components, Controlled vs Uncontrolled Inputs, React Hook Form + Zod & Accessible Modals in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Next Topic 6.4: React Rendering Performance: Profiler Flamegraphs, Virtualized Lists, Code Splitting & Suspense Boundaries

6.4 React Rendering Performance: Profiler Flamegraphs, Virtualized Lists, Code Splitting & Suspense Boundaries [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for React Rendering Performance • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply React Rendering Performance: Profiler Flamegraphs, Virtualized Lists, Code Splitting & Suspense Boundaries in real Node.js, TypeScript & Modern Fu...

What Topic Is Next → Module 6 Capstone (scaleflow-m06-react19-hooks-architecture) → Next Topic 7.1: Multi-Role Client Routing, Protected Auth Guards, Lazy Route Bundles & URL Search-Param State

Module 6 Hands-On Capstone Project (100 Marks): scaleflow-m06-react19-hooks-architecture — Architect, implement, test, and push the production GitHub repository 'scaleflow-m06-react19-hooks-architecture' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module...

MODULE 7 OF 22 • STAGE 2 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Enterprise React State Management, TanStack Query & Automated UI Testing

Beginner-to-expert real-repository module covering Enterprise React State Management, TanStack Query & Automated UI Testing with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Multi-Role Client Routing, Protected Auth Guards, Lazy Route Bundles & URL Search-Param State — Purpose: Make filters, tabs, pagination, and modals deep-linkable, shareable, and role-protected. • ✓ Server-State Mastery with TanStack Query v5: Caching, Stale-While-Revalidate, Infinite Scroll & Mutations — Purpose: Replace hundreds of lines of manual useEffect fetch boilerplate with resilient declarative cache sync. • ✓ Scalable Client State with Zustand / Redux Toolkit Slices, Optimistic Rollbacks & Real-Time Toast Sync — Purpose: Coordinate complex shopping-cart, workspace, and multi-panel editor state predictably. • ✓ Frontend Testing Pyramid: Vitest Unit Suites, React Testing Library User Flows & Mock Service Worker (MSW) — Purpose: Verify real user clicks, keyboard flows, and API error states without brittle implementation-detail tests.

7.1 Multi-Role Client Routing, Protected Auth Guards, Lazy Route Bundles & URL Search-Param State [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Multi-Role Client Routing • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Multi-Role Client Routing, Protected Auth Guards, Lazy Route Bundles & URL Search-Param State in real Node.js, TypeScript & Modern Full Stack Engi...

What Topic Is Next → Next Topic 7.2: Server-State Mastery with TanStack Query v5: Caching, Stale-While-Revalidate, Infinite Scroll & Mutations

7.2 Server-State Mastery with TanStack Query v5: Caching, Stale-While-Revalidate, Infinite Scroll & Mutations [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Server-State Mastery with TanStack Query v5 • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Server-State Mastery with TanStack Query v5: Caching, Stale-While-Revalidate, Infinite Scroll & Mutations in real Node.js, TypeScript & Modern Ful...

What Topic Is Next → Next Topic 7.3: Scalable Client State with Zustand / Redux Toolkit Slices, Optimistic Rollbacks & Real-Time Toast Sync

7.3 Scalable Client State with Zustand / Redux Toolkit Slices, Optimistic Rollbacks & Real-Time Toast Sync [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Scalable Client State with Zustand / Redux Toolkit S... • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Scalable Client State with Zustand / Redux Toolkit Slices, Optimistic Rollbacks & Real-Time Toast Sync in real Node.js, TypeScript & Modern Full S...

What Topic Is Next → Next Topic 7.4: Frontend Testing Pyramid: Vitest Unit Suites, React Testing Library User Flows & Mock Service Worker (MSW)

7.4 Frontend Testing Pyramid: Vitest Unit Suites, React Testing Library User Flows & Mock Service Worker (MSW) [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Frontend Testing Pyramid • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Frontend Testing Pyramid: Vitest Unit Suites, React Testing Library User Flows & Mock Service Worker (MSW) in real Node.js, TypeScript & Modern Fu...

What Topic Is Next → Module 7 Capstone (scaleflow-m07-react-state-tanstack-vitest) → Next Topic 8.1: Next.js 15 App Router File Conventions, Nested Layouts, Loading Skeletons & Error Boundaries

Module 7 Hands-On Capstone Project (100 Marks): scaleflow-m07-react-state-tanstack-vitest — Architect, implement, test, and push the production GitHub repository 'scaleflow-m07-react-state-tanstack-vitest' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module...

MODULE 8 OF 22 • STAGE 2 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Next.js 15 App Router, React Server Components, SSR/ISR & Edge Middleware

Beginner-to-expert real-repository module covering Next.js 15 App Router, React Server Components, SSR/ISR & Edge Middleware with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Next.js 15 App Router File Conventions, Nested Layouts, Loading Skeletons & Error Boundaries — Purpose: Structure modern hybrid web applications with instant route transitions and isolated failure recovery. • ✓ React Server Components (RSC) vs Client Islands, Streaming SSR, Static Generation (SSG) & ISR Caching — Purpose: Deliver near-zero client JS bundles and sub-second Largest Contentful Paint (LCP) for public pages. • ✓ Validated Next.js Server

Actions, Route Handlers, Edge Auth Middleware & Dynamic Metadata / Sitemaps — Purpose: Mutate backend state securely with progressive enhancement and rank storefronts on Google Search. • ✓ Image / Font Optimization, Content Security Policy (CSP) Headers & Production Standalone Builds — Purpose: Achieve 95+ Lighthouse scores across Performance, Accessibility, Best Practices, and SEO.

8.1 Next.js 15 App Router File Conventions, Nested Layouts, Loading Skeletons & Error Boundaries [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Next.js 15 App Router File Conventions • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Next.js 15 App Router File Conventions, Nested Layouts, Loading Skeletons & Error Boundaries in real Node.js, TypeScript & Modern Full Stack Engin...

What Topic Is Next → Next Topic 8.2: React Server Components (RSC) vs Client Islands, Streaming SSR, Static Generation (SSG) & ISR Caching

8.2 React Server Components (RSC) vs Client Islands, Streaming SSR, Static Generation (SSG) & ISR Caching [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for React Server Components (RSC) vs Client Islands • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply React Server Components (RSC) vs Client Islands, Streaming SSR, Static Generation (SSG) & ISR Caching in real Node.js, TypeScript & Modern Full St...

What Topic Is Next → Next Topic 8.3: Validated Next.js Server Actions, Route Handlers, Edge Auth Middleware & Dynamic Metadata / Sitemaps

8.3 Validated Next.js Server Actions, Route Handlers, Edge Auth Middleware & Dynamic Metadata / Sitemaps [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Validated Next.js Server Actions • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Validated Next.js Server Actions, Route Handlers, Edge Auth Middleware & Dynamic Metadata / Sitemaps in real Node.js, TypeScript & Modern Full Sta...

What Topic Is Next → Next Topic 8.4: Image / Font Optimization, Content Security Policy (CSP) Headers & Production Standalone Builds

8.4 Image / Font Optimization, Content Security Policy (CSP) Headers & Production Standalone Builds [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Image / Font Optimization • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Image / Font Optimization, Content Security Policy (CSP) Headers & Production Standalone Builds in real Node.js, TypeScript & Modern Full Stack En...

What Topic Is Next → Module 8 Capstone (scaleflow-m08-nextjs15-fullstack-web) → Next Topic 9.1: Node.js Architecture From Scratch: V8 + Libuv Thread Pool, CommonJS vs ESM, process & Environment Config

Module 8 Hands-On Capstone Project (100 Marks): scaleflow-m08-nextjs15-fullstack-web — Architect, implement, test, and push the production GitHub repository 'scaleflow-m08-nextjs15-fullstack-web' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 8: N...

MODULE 9 OF 22 • STAGE 2 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Node.js 22 Runtime Internals, Libuv Event Loop, Streams & CLI Tooling

Beginner-to-expert real-repository module covering Node.js 22 Runtime Internals, Libuv Event Loop, Streams & CLI Tooling with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Node.js Architecture From Scratch: V8 + Libuv Thread Pool, CommonJS vs ESM, process & Environment Config — Purpose: Understand how Node.js handles 20,000+ concurrent I/O connections on a single JavaScript thread. • ✓ EventEmitter Patterns, Buffer Binary Data, fs/promises File System & Path Security (Path Traversal Defense) — Purpose: Process binary uploads and local files safely without blocking the event loop or exposing host directories. • ✓ Readable, Writable & Transform Streams, pipeline() Backpressure & Memory-Safe Multi-GB CSV/Log Processing — Purpose: Stream 5 GB datasets in under 40 MB of RAM using proper Node.js backpressure control. • ✓ Worker Threads for CPU-Bound Tasks, Child Processes, Graceful SIGTERM Shutdown & Node Diagnostic Profiling — Purpose: Offload heavy hashing/PDF generation to worker pools and drain in-flight HTTP sockets cleanly on deploy.

9.1 Node.js Architecture From Scratch: V8 + Libuv Thread Pool, CommonJS vs ESM, process & Environment Config [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Node.js Architecture From Scratch • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Node.js Architecture From Scratch: V8 + Libuv Thread Pool, CommonJS vs ESM, process & Environment Config in real Node.js, TypeScript & Modern Full...

What Topic Is Next → Next Topic 9.2: EventEmitter Patterns, Buffer Binary Data, fs/promises File System & Path Security (Path Traversal Defense)

9.2 EventEmitter Patterns, Buffer Binary Data, fs/promises File System & Path Security (Path Traversal Defense) [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for EventEmitter Patterns • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply EventEmitter Patterns, Buffer Binary Data, fs/promises File System & Path Security (Path Traversal Defense) in real Node.js, TypeScript & Modern F...

What Topic Is Next → Next Topic 9.3: Readable, Writable & Transform Streams, pipeline() Backpressure & Memory-Safe Multi-GB CSV/Log Processing

9.3 Readable, Writable & Transform Streams, pipeline() Backpressure & Memory-Safe Multi-GB CSV/Log Processing [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Readable • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Readable, Writable & Transform Streams, pipeline() Backpressure & Memory-Safe Multi-GB CSV/Log Processing in real Node.js, TypeScript & Modern Ful...

What Topic Is Next → Next Topic 9.4: Worker Threads for CPU-Bound Tasks, Child Processes, Graceful SIGTERM Shutdown & Node Diagnostic Profiling

9.4 Worker Threads for CPU-Bound Tasks, Child Processes, Graceful SIGTERM Shutdown & Node Diagnostic Profiling [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Worker Threads for CPU-Bound Tasks • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Worker Threads for CPU-Bound Tasks, Child Processes, Graceful SIGTERM Shutdown & Node Diagnostic Profiling in real Node.js, TypeScript & Modern Fu...

What Topic Is Next → Module 9 Capstone (scaleflow-m09-nodejs22-runtime-streams) → Next Topic 10.1: HTTP Server Internals, Express Routing, Modular Controllers/Services & Typed Request/Response Contracts

Module 9 Hands-On Capstone Project (100 Marks): scaleflow-m09-nodejs22-runtime-streams — Architect, implement, test, and push the production GitHub repository 'scaleflow-m09-nodejs22-runtime-streams' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 9:...

MODULE 10 OF 22 • STAGE 2 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Express.js REST API Engineering, Middleware Pipelines & OpenAPI Contracts

Beginner-to-expert real-repository module covering Express.js REST API Engineering, Middleware Pipelines & OpenAPI Contracts with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ HTTP Server Internals, Express Routing, Modular Controllers/Services & Typed Request/Response Contracts — Purpose: Architect clean 3-layer (Controller -> Service -> Repository) Node.js TypeScript services with zero spaghetti routes. • ✓ Custom Middleware Chains: Correlation IDs, Zod Request Validation, Multer S3 Streaming & Centralized Errors — Purpose: Validate params, query strings, and JSON bodies at the edge and return RFC 9457 ProblemDetail responses. • ✓ Pagination (Offset vs Cursor), Filtering, Sorting, Idempotency Keys & Webhook HMAC Verification — Purpose: Build Stripe/Razorpay-grade resilient APIs that never double-charge customers on network retries. • ✓ OpenAPI 3.1 / Swagger UI Generation, Structured Pino JSON Logging & Health / Readiness Endpoints — Purpose: Publish live API contracts and emit structured millisecond-latency request logs ready for cloud aggregators.

10.1 HTTP Server Internals, Express Routing, Modular Controllers/Services & Typed Request/Response Contracts [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for HTTP Server Internals • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply HTTP Server Internals, Express Routing, Modular Controllers/Services & Typed Request/Response Contracts in real Node.js, TypeScript & Modern Full...

What Topic Is Next → Next Topic 10.2: Custom Middleware Chains: Correlation IDs, Zod Request Validation, Multer S3 Streaming & Centralized Errors

10.2 Custom Middleware Chains: Correlation IDs, Zod Request Validation, Multer S3 Streaming & Centralized Errors [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Custom Middleware Chains • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Custom Middleware Chains: Correlation IDs, Zod Request Validation, Multer S3 Streaming & Centralized Errors in real Node.js, TypeScript & Modern F...

What Topic Is Next → Next Topic 10.3: Pagination (Offset vs Cursor), Filtering, Sorting, Idempotency Keys & Webhook HMAC Verification

10.3 Pagination (Offset vs Cursor), Filtering, Sorting, Idempotency Keys & Webhook HMAC Verification [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Pagination (Offset vs Cursor) • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Pagination (Offset vs Cursor), Filtering, Sorting, Idempotency Keys & Webhook HMAC Verification in real Node.js, TypeScript & Modern Full Stack En...

What Topic Is Next → Next Topic 10.4: OpenAPI 3.1 / Swagger UI Generation, Structured Pino JSON Logging & Health / Readiness Endpoints

10.4 OpenAPI 3.1 / Swagger UI Generation, Structured Pino JSON Logging & Health / Readiness Endpoints [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for OpenAPI 3.1 / Swagger UI Generation • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply OpenAPI 3.1 / Swagger UI Generation, Structured Pino JSON Logging & Health / Readiness Endpoints in real Node.js, TypeScript & Modern Full Stack E...

What Topic Is Next → Module 10 Capstone (scaleflow-m10-expressjs-rest-platform) → Next Topic 11.1: NestJS Modules, Controllers, Injectable Providers, Scopes & Inversion of Control (IoC) Container

Module 10 Hands-On Capstone Project (100 Marks): scaleflow-m10-expressjs-rest-platform — Architect, implement, test, and push the production GitHub repository 'scaleflow-m10-expressjs-rest-platform' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 10:...

MODULE 11 OF 22 • STAGE 2 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Enterprise Backend Architecture with NestJS, Dependency Injection & Clean DDD

Beginner-to-expert real-repository module covering Enterprise Backend Architecture with NestJS, Dependency Injection & Clean DDD with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ NestJS Modules, Controllers, Injectable Providers, Scopes & Inversion of Control (IoC) Container — Purpose: Master the enterprise TypeScript backend framework favored by global FinTech and B2B SaaS engineering teams. • ✓ NestJS Pipes, Guards, Interceptors, Custom Decorators & Exception Filters in Production — Purpose: Enforce authentication, RBAC permissions, response serialization, and audit logging declaratively. • ✓ Hexagonal / Clean Domain-Driven Design (DDD), Typed ConfigModule Validation & Lifecycle Hooks — Purpose: Keep core domain business rules 100% framework-independent and fail fast if env variables are missing. • ✓ Microservice Transports, Scheduled Cron Jobs, Health Terminus Checks & Modular Monolith Boundaries — Purpose: Structure large multi-team TypeScript backends that scale cleanly from modular monolith to microservices.

11.1 NestJS Modules, Controllers, Injectable Providers, Scopes & Inversion of Control (IoC) Container [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for NestJS Modules • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply NestJS Modules, Controllers, Injectable Providers, Scopes & Inversion of Control (IoC) Container in real Node.js, TypeScript & Modern Full Stack E...

What Topic Is Next → Next Topic 11.2: NestJS Pipes, Guards, Interceptors, Custom Decorators & Exception Filters in Production

11.2 NestJS Pipes, Guards, Interceptors, Custom Decorators & Exception Filters in Production [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for NestJS Pipes • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply NestJS Pipes, Guards, Interceptors, Custom Decorators & Exception Filters in Production in real Node.js, TypeScript & Modern Full Stack Engineerin...

What Topic Is Next → Next Topic 11.3: Hexagonal / Clean Domain-Driven Design (DDD), Typed ConfigModule Validation & Lifecycle Hooks

11.3 Hexagonal / Clean Domain-Driven Design (DDD), Typed ConfigModule Validation & Lifecycle Hooks [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Hexagonal / Clean Domain-Driven Design (DDD) • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Hexagonal / Clean Domain-Driven Design (DDD), Typed ConfigModule Validation & Lifecycle Hooks in real Node.js, TypeScript & Modern Full Stack Engi...

What Topic Is Next → Next Topic 11.4: Microservice Transports, Scheduled Cron Jobs, Health Terminus Checks & Modular Monolith Boundaries

11.4 Microservice Transports, Scheduled Cron Jobs, Health Terminus Checks & Modular Monolith Boundaries [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Microservice Transports • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Microservice Transports, Scheduled Cron Jobs, Health Terminus Checks & Modular Monolith Boundaries in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Module 11 Capstone (scaleflow-m11-nestjs-clean-architecture) → Next Topic 12.1: Relational Modeling From Scratch, 1NF–3NF Normalization, Foreign Keys & Core SQL (SELECT, JOIN, GROUP BY)

Module 11 Hands-On Capstone Project (100 Marks): scaleflow-m11-nestjs-clean-architecture — Architect, implement, test, and push the production GitHub repository 'scaleflow-m11-nestjs-clean-architecture' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 1...

MODULE 12 OF 22 • STAGE 2 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Relational Databases & SQL From Scratch to PostgreSQL 16 + Prisma / Drizzle (PERN)

Beginner-to-expert real-repository module covering Relational Databases & SQL From Scratch to PostgreSQL 16 + Prisma / Drizzle (PERN) with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Relational Modeling From Scratch, 1NF–3NF Normalization, Foreign Keys & Core SQL (SELECT, JOIN, GROUP BY) — Purpose: Build rock-solid SQL foundations before writing a single ORM line so you never fear raw database queries. • ✓ Advanced SQL: CTEs, Window Functions, ACID Isolation Levels, Row Locks (FOR UPDATE) & EXPLAIN ANALYZE — Purpose: Eliminate race conditions on wallet/inventory balances and tune slow queries using composite B-Tree indexes. • ✓ Type-Safe ORM Engineering with Prisma & Drizzle ORM: Relations, Transactions & N+1 Query Elimination — Purpose: Enjoy 100% autocomplete-safe SQL queries in TypeScript while maintaining full control over generated SQL. • ✓ Versioned SQL Migrations, Connection Pooling (PgBouncer) & Zero-Downtime Expand-Contract Schema Rollouts — Purpose: Deploy schema evolutions safely under live traffic without exhausting PostgreSQL max_connections.

12.1 Relational Modeling From Scratch, 1NF–3NF Normalization, Foreign Keys & Core SQL (SELECT, JOIN, GROUP BY) [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Relational Modeling From Scratch • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Relational Modeling From Scratch, 1NF–3NF Normalization, Foreign Keys & Core SQL (SELECT, JOIN, GROUP BY) in real Node.js, TypeScript & Modern Ful...

What Topic Is Next → Next Topic 12.2: Advanced SQL: CTEs, Window Functions, ACID Isolation Levels, Row Locks (FOR UPDATE) & EXPLAIN ANALYZE

12.2 Advanced SQL: CTEs, Window Functions, ACID Isolation Levels, Row Locks (FOR UPDATE) & EXPLAIN ANALYZE [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Advanced SQL • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Advanced SQL: CTEs, Window Functions, ACID Isolation Levels, Row Locks (FOR UPDATE) & EXPLAIN ANALYZE in real Node.js, TypeScript & Modern Full St...

What Topic Is Next → Next Topic 12.3: Type-Safe ORM Engineering with Prisma & Drizzle ORM: Relations, Transactions & N+1 Query Elimination

12.3 Type-Safe ORM Engineering with Prisma & Drizzle ORM: Relations, Transactions & N+1 Query Elimination [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Type-Safe ORM Engineering with Prisma • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Type-Safe ORM Engineering with Prisma & Drizzle ORM: Relations, Transactions & N+1 Query Elimination in real Node.js, TypeScript & Modern Full Sta...

What Topic Is Next → Next Topic 12.4: Versioned SQL Migrations, Connection Pooling (PgBouncer) & Zero-Downtime Expand-Contract Schema Rollouts

12.4 Versioned SQL Migrations, Connection Pooling (PgBouncer) & Zero-Downtime Expand-Contract Schema Rollouts [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Versioned SQL Migrations • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Versioned SQL Migrations, Connection Pooling (PgBouncer) & Zero-Downtime Expand-Contract Schema Rollouts in real Node.js, TypeScript & Modern Full...

What Topic Is Next → Module 12 Capstone (scaleflow-m12-postgresql-prisma-drizzle) → Next Topic 13.1: NoSQL vs SQL Trade-Offs, BSON Types, Embedding vs Referencing & Polymorphic Catalog Schema Design

Module 12 Hands-On Capstone Project (100 Marks): scaleflow-m12-postgresql-prisma-drizzle — Architect, implement, test, and push the production GitHub repository 'scaleflow-m12-postgresql-prisma-drizzle' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 1...

MODULE 13 OF 22 • STAGE 3 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

MongoDB 7 From Scratch, Document Modeling & Aggregation Pipelines (MERN)

Beginner-to-expert real-repository module covering MongoDB 7 From Scratch, Document Modeling & Aggregation Pipelines (MERN) with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ NoSQL vs SQL Trade-Offs, BSON Types, Embedding vs Referencing & Polymorphic Catalog Schema Design — Purpose: Know exactly when to choose PostgreSQL ACID tables versus flexible MongoDB document collections. • ✓ MongoDB CRUD Operations, Atomic Operators ($inc, $push, $addToSet), Bulk Writes & Schema Validation — Purpose: Mutate nested documents atomically at high throughput while enforcing JSON Schema collection rules. • ✓ Multi-Stage Aggregation Pipelines ($match, $lookup, $unwind, $group, $facet) & Time-Series Analytics — Purpose: Compute real-time revenue funnels, faceted product filters, and cohort metrics directly inside MongoDB. • ✓ Compound / ESR Indexing Rules, Covered Queries, Multi-Document ACID Transactions & Replica Set Failover — Purpose: Optimize slow collection scans using Explain plans and run MongoDB Atlas clusters with high availability.

13.1 NoSQL vs SQL Trade-Offs, BSON Types, Embedding vs Referencing & Polymorphic Catalog Schema Design [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for NoSQL vs SQL Trade-Offs • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply NoSQL vs SQL Trade-Offs, BSON Types, Embedding vs Referencing & Polymorphic Catalog Schema Design in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Next Topic 13.2: MongoDB CRUD Operations, Atomic Operators ($inc, $push, $addToSet), Bulk Writes & Schema Validation

13.2 MongoDB CRUD Operations, Atomic Operators ($inc, $push, $addToSet), Bulk Writes & Schema Validation [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for MongoDB CRUD Operations • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply MongoDB CRUD Operations, Atomic Operators ($inc, $push, $addToSet), Bulk Writes & Schema Validation in real Node.js, TypeScript & Modern Full Stac...

What Topic Is Next → Next Topic 13.3: Multi-Stage Aggregation Pipelines ($match, $lookup, $unwind, $group, $facet) & Time-Series Analytics

13.3 Multi-Stage Aggregation Pipelines ($match, $lookup, $unwind, $group, $facet) & Time-Series Analytics [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Multi-Stage Aggregation Pipelines ($match, $lookup,... • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Multi-Stage Aggregation Pipelines ($match, $lookup, $unwind, $group, $facet) & Time-Series Analytics in real Node.js, TypeScript & Modern Full Sta...

What Topic Is Next → Next Topic 13.4: Compound / ESR Indexing Rules, Covered Queries, Multi-Document ACID Transactions & Replica Set Failover

13.4 Compound / ESR Indexing Rules, Covered Queries, Multi-Document ACID Transactions & Replica Set Failover [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Compound / ESR Indexing Rules • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Compound / ESR Indexing Rules, Covered Queries, Multi-Document ACID Transactions & Replica Set Failover in real Node.js, TypeScript & Modern Full...

What Topic Is Next → Module 13 Capstone (scaleflow-m13-mongodb7-mern-aggregations) → Next Topic 14.1: Password Security (Argon2id), Session Cookies vs Stateless JWTs, HttpOnly / SameSite / Secure Flags

Module 13 Hands-On Capstone Project (100 Marks): scaleflow-m13-mongodb7-mern-aggregations — Architect, implement, test, and push the production GitHub repository 'scaleflow-m13-mongodb7-mern-aggregations' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module...

MODULE 14 OF 22 • STAGE 3 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Full-Stack Security: JWT Rotation, OAuth2 / Passkeys, Multi-Tenant RBAC & OWASP

Beginner-to-expert real-repository module covering Full-Stack Security: JWT Rotation, OAuth2 / Passkeys, Multi-Tenant RBAC & OWASP with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Password Security (Argon2id), Session Cookies vs Stateless JWTs, HttpOnly / SameSite / Secure Flags — Purpose: Build authentication flows immune to XSS token theft and credential-stuffing attacks. • ✓ Refresh Token Rotation, Reuse Detection, OAuth2 / OIDC Social SSO (Google/GitHub) & 2FA TOTP / OTP — Purpose: Implement enterprise login, device session revocation, and multi-factor verification end-to-end. • ✓ Multi-Tenant Organization Switching, Role &

Permission Matrices (RBAC / ABAC) & Row-Level Guards — Purpose: Ensure strict workspace isolation across B2B SaaS APIs, Next.js Server Actions, and database queries. • ✓ Helmet CSP, CORS Allowlists, CSRF Protection, S3 Presigned Upload Guards & OWASP Top-10 Audits — Purpose: Harden every API and frontend surface against injection, SSRF, BOLA, and prototype pollution.

14.1 Password Security (Argon2id), Session Cookies vs Stateless JWTs, HttpOnly / SameSite / Secure Flags [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Password Security (Argon2id) • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Password Security (Argon2id), Session Cookies vs Stateless JWTs, HttpOnly / SameSite / Secure Flags in real Node.js, TypeScript & Modern Full Stac...

What Topic Is Next → Next Topic 14.2: Refresh Token Rotation, Reuse Detection, OAuth2 / OIDC Social SSO (Google/GitHub) & 2FA TOTP / OTP

14.2 Refresh Token Rotation, Reuse Detection, OAuth2 / OIDC Social SSO (Google/GitHub) & 2FA TOTP / OTP [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Refresh Token Rotation • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Refresh Token Rotation, Reuse Detection, OAuth2 / OIDC Social SSO (Google/GitHub) & 2FA TOTP / OTP in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Next Topic 14.3: Multi-Tenant Organization Switching, Role & Permission Matrices (RBAC / ABAC) & Row-Level Guards

14.3 Multi-Tenant Organization Switching, Role & Permission Matrices (RBAC / ABAC) & Row-Level Guards [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Multi-Tenant Organization Switching • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Multi-Tenant Organization Switching, Role & Permission Matrices (RBAC / ABAC) & Row-Level Guards in real Node.js, TypeScript & Modern Full Stack E...

What Topic Is Next → Next Topic 14.4: Helmet CSP, CORS Allowlists, CSRF Protection, S3 Presigned Upload Guards & OWASP Top-10 Audits

14.4 Helmet CSP, CORS Allowlists, CSRF Protection, S3 Presigned Upload Guards & OWASP Top-10 Audits [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Helmet CSP • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Helmet CSP, CORS Allowlists, CSRF Protection, S3 Presigned Upload Guards & OWASP Top-10 Audits in real Node.js, TypeScript & Modern Full Stack Eng...

What Topic Is Next → Module 14 Capstone (scaleflow-m14-auth-security-owasp) → Next Topic 15.1: Redis 7 Strings, Hashes, Sorted Sets, Streams, Cache-Aside Invalidation & Lua Sliding-Window Rate Limiters

Module 14 Hands-On Capstone Project (100 Marks): scaleflow-m14-auth-security-owasp — Architect, implement, test, and push the production GitHub repository 'scaleflow-m14-auth-security-owasp' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 14: Ful...

MODULE 15 OF 22 • STAGE 3 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Redis 7 Caching, BullMQ Background Queues & Real-Time WebSockets (Socket.IO)

Beginner-to-expert real-repository module covering Redis 7 Caching, BullMQ Background Queues & Real-Time WebSockets (Socket.IO) with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Redis 7 Strings, Hashes, Sorted Sets, Streams, Cache-Aside Invalidation & Lua Sliding-Window Rate Limiters — Purpose: Serve hot catalog/session reads in <1ms and throttle abusive IPs or API keys atomically. • ✓ Distributed Background Jobs with BullMQ: Retries, Priorities, Delayed Schedules, Flows & Dead-Letter Queues — Purpose: Process invoice PDFs, transactional emails, and webhook retries reliably outside HTTP request threads. • ✓ Bidirectional Real-Time Engineering with WebSockets & Socket.IO: Rooms, Presence, Typing & Ack Delivery — Purpose: Build live collaborative boards, instant chat, and real-time order-tracking dashboards. • ✓ Scaling WebSockets Horizontally Across Multiple Node Pods with Redis Pub/Sub Adapters & GenAI SSE Streams — Purpose: Broadcast live events seamlessly across load-balanced containers and stream AI copilot responses.

15.1 Redis 7 Strings, Hashes, Sorted Sets, Streams, Cache-Aside Invalidation & Lua Sliding-Window Rate Limiters [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Redis 7 Strings • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Redis 7 Strings, Hashes, Sorted Sets, Streams, Cache-Aside Invalidation & Lua Sliding-Window Rate Limiters in real Node.js, TypeScript & Modern Fu...

What Topic Is Next → Next Topic 15.2: Distributed Background Jobs with BullMQ: Retries, Priorities, Delayed Schedules, Flows & Dead-Letter Queues

15.2 Distributed Background Jobs with BullMQ: Retries, Priorities, Delayed Schedules, Flows & Dead-Letter Queues [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Distributed Background Jobs with BullMQ • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Distributed Background Jobs with BullMQ: Retries, Priorities, Delayed Schedules, Flows & Dead-Letter Queues in real Node.js, TypeScript & Modern F...

What Topic Is Next → Next Topic 15.3: Bidirectional Real-Time Engineering with WebSockets & Socket.IO: Rooms, Presence, Typing & Ack Delivery

15.3 Bidirectional Real-Time Engineering with WebSockets & Socket.IO: Rooms, Presence, Typing & Ack Delivery [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Bidirectional Real-Time Engineering with WebSockets • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Bidirectional Real-Time Engineering with WebSockets & Socket.IO: Rooms, Presence, Typing & Ack Delivery in real Node.js, TypeScript & Modern Full...

What Topic Is Next → Next Topic 15.4: Scaling WebSockets Horizontally Across Multiple Node Pods with Redis Pub/Sub Adapters & GenAI SSE Streams

15.4 Scaling WebSockets Horizontally Across Multiple Node Pods with Redis Pub/Sub Adapters & GenAI SSE Streams [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Scaling WebSockets Horizontally Across Multiple Node... • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Scaling WebSockets Horizontally Across Multiple Node Pods with Redis Pub/Sub Adapters & GenAI SSE Streams in real Node.js, TypeScript & Modern Ful...

What Topic Is Next → Module 15 Capstone (scaleflow-m15-redis-bullmq-websockets) → Next Topic 16.1: Testing Strategy for Full-Stack TypeScript: Unit vs Integration vs Contract vs End-to-End Pyramid

Module 15 Hands-On Capstone Project (100 Marks): scaleflow-m15-redis-bullmq-websockets — Architect, implement, test, and push the production GitHub repository 'scaleflow-m15-redis-bullmq-websockets' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 15:...

MODULE 16 OF 22 • STAGE 3 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

End-to-End Automated Testing: Vitest, Supertest, Testcontainers & Playwright E2E

Beginner-to-expert real-repository module covering End-to-End Automated Testing: Vitest, Supertest, Testcontainers & Playwright E2E with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Testing Strategy for Full-Stack TypeScript: Unit vs Integration vs Contract vs End-to-End Pyramid — Purpose: Focus testing effort where real production regressions happen instead of chasing vanity mocks. • ✓ Backend API Integration Suites with Vitest, Supertest & Real PostgreSQL / MongoDB / Redis Testcontainers — Purpose: Spin up real isolated databases in CI to verify auth, SQL transactions, and race conditions accurately. • ✓ Cross-Browser End-to-End (E2E) Automation with Playwright: Auth StorageState, Stripe/Razorpay Flows & Traces — Purpose: Verify signup, checkout, RBAC permissions, and real-time updates across Chromium, Firefox, and WebKit. • ✓ Load & Concurrency Benchmarking with k6 / Autocannon: Detecting Event-Loop Lag & Connection Leaks — Purpose: Prove your Node.js/NestJS APIs sustain target RPS and p95<80ms before shipping to customers.

16.1 Testing Strategy for Full-Stack TypeScript: Unit vs Integration vs Contract vs End-to-End Pyramid [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Testing Strategy for Full-Stack TypeScript • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Testing Strategy for Full-Stack TypeScript: Unit vs Integration vs Contract vs End-to-End Pyramid in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Next Topic 16.2: Backend API Integration Suites with Vitest, Supertest & Real PostgreSQL / MongoDB / Redis Testcontainers

16.2 Backend API Integration Suites with Vitest, Supertest & Real PostgreSQL / MongoDB / Redis Testcontainers [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Backend API Integration Suites with Vitest • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Backend API Integration Suites with Vitest, Supertest & Real PostgreSQL / MongoDB / Redis Testcontainers in real Node.js, TypeScript & Modern Full...

What Topic Is Next → Next Topic 16.3: Cross-Browser End-to-End (E2E) Automation with Playwright: Auth StorageState, Stripe/Razorpay Flows & Traces

16.3 Cross-Browser End-to-End (E2E) Automation with Playwright: Auth StorageState, Stripe/Razorpay Flows & Traces [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Cross-Browser End-to-End (E2E) Automation with Playw... • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Cross-Browser End-to-End (E2E) Automation with Playwright: Auth StorageState, Stripe/Razorpay Flows & Traces in real Node.js, TypeScript & Modern...

What Topic Is Next → Next Topic 16.4: Load & Concurrency Benchmarking with k6 / Autocannon: Detecting Event-Loop Lag & Connection Leaks

16.4 Load & Concurrency Benchmarking with k6 / Autocannon: Detecting Event-Loop Lag & Connection Leaks [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Load • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Load & Concurrency Benchmarking with k6 / Autocannon: Detecting Event-Loop Lag & Connection Leaks in real Node.js, TypeScript & Modern Full Stack...

What Topic Is Next → Module 16 Capstone (scaleflow-m16-automated-testing-e2e) → Next Topic 17.1: Docker From Scratch: Multi-Stage Node 22 Alpine / Distroless Dockerfiles, pnpm Layer Caching & Non-Root

Module 16 Hands-On Capstone Project (100 Marks): scaleflow-m16-automated-testing-e2e — Architect, implement, test, and push the production GitHub repository 'scaleflow-m16-automated-testing-e2e' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 16: E...

MODULE 17 OF 22 • STAGE 3 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Docker Containerization, Nginx Reverse Proxies & AWS Cloud Infrastructure

Beginner-to-expert real-repository module covering Docker Containerization, Nginx Reverse Proxies & AWS Cloud Infrastructure with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Docker From Scratch: Multi-Stage Node 22 Alpine / Distroless Dockerfiles, pnpm Layer Caching & Non-Root — Purpose: Build minimal, fast-booting production images free from devDependencies and OS bloat. • ✓ Docker Compose Full-Stack Orchestration & Nginx Reverse Proxy (TLS, Gzip/Brotli, WebSocket Upgrade) — Purpose: Run Next.js, NestJS, PostgreSQL, MongoDB, Redis, and Nginx together with production parity. • ✓ AWS Cloud Core: IAM Least Privilege, Multi-AZ VPC, Security Groups, EC2, ALB & Amazon ECR Registries — Purpose: Architect secure AWS network topologies and deploy load-balanced containerized Node/Next workloads. • ✓ AWS Managed Data & Edge: RDS PostgreSQL, S3 + CloudFront CDN, Route 53 TLS DNS & AWS Secrets Manager — Purpose: Serve global assets from CloudFront edge PoPs and safeguard production databases with automated backups.

17.1 Docker From Scratch: Multi-Stage Node 22 Alpine / Distroless Dockerfiles, pnpm Layer Caching & Non-Root [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Docker From Scratch • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Docker From Scratch: Multi-Stage Node 22 Alpine / Distroless Dockerfiles, pnpm Layer Caching & Non-Root in real Node.js, TypeScript & Modern Full...

What Topic Is Next → Next Topic 17.2: Docker Compose Full-Stack Orchestration & Nginx Reverse Proxy (TLS, Gzip/Brotli, WebSocket Upgrade)

17.2 Docker Compose Full-Stack Orchestration & Nginx Reverse Proxy (TLS, Gzip/Brotli, WebSocket Upgrade) [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Docker Compose Full-Stack Orchestration • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Docker Compose Full-Stack Orchestration & Nginx Reverse Proxy (TLS, Gzip/Brotli, WebSocket Upgrade) in real Node.js, TypeScript & Modern Full Stac...

What Topic Is Next → Next Topic 17.3: AWS Cloud Core: IAM Least Privilege, Multi-AZ VPC, Security Groups, EC2, ALB & Amazon ECR Registries

17.3 AWS Cloud Core: IAM Least Privilege, Multi-AZ VPC, Security Groups, EC2, ALB & Amazon ECR Registries [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for AWS Cloud Core • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply AWS Cloud Core: IAM Least Privilege, Multi-AZ VPC, Security Groups, EC2, ALB & Amazon ECR Registries in real Node.js, TypeScript & Modern Full Sta...

What Topic Is Next → Next Topic 17.4: AWS Managed Data & Edge: RDS PostgreSQL, S3 + CloudFront CDN, Route 53 TLS DNS & AWS Secrets Manager

17.4 AWS Managed Data & Edge: RDS PostgreSQL, S3 + CloudFront CDN, Route 53 TLS DNS & AWS Secrets Manager [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for AWS Managed Data • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply AWS Managed Data & Edge: RDS PostgreSQL, S3 + CloudFront CDN, Route 53 TLS DNS & AWS Secrets Manager in real Node.js, TypeScript & Modern Full Sta...

What Topic Is Next → Module 17 Capstone (scaleflow-m17-docker-aws-cloud) → Next Topic 18.1: GitHub Actions CI/CD Pipelines: Typecheck, Lint, Testcontainers, Playwright, Trivy Scan & Zero-Downtime Deploy

Module 17 Hands-On Capstone Project (100 Marks): scaleflow-m17-docker-aws-cloud — Architect, implement, test, and push the production GitHub repository 'scaleflow-m17-docker-aws-cloud' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 17: Docker...

MODULE 18 OF 22 • STAGE 4 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

GitHub Actions DevSecOps CI/CD, Terraform Basics & OpenTelemetry Observability

Beginner-to-expert real-repository module covering GitHub Actions DevSecOps CI/CD, Terraform Basics & OpenTelemetry Observability with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ GitHub Actions CI/CD Pipelines: Typecheck, Lint, Testcontainers, Playwright, Trivy Scan & Zero-Downtime Deploy — Purpose: Automate every verification gate and roll out new container revisions onto AWS via OIDC with zero SSH keys. • ✓ Infrastructure as Code Foundations with Terraform: Reproducible AWS Networking, S3, ECR & ECS Services — Purpose: Eliminate ClickOps configuration drift by defining cloud resources cleanly in version-controlled HCL. • ✓ OpenTelemetry Distributed Tracing Across Next.js, NestJS/Express, PostgreSQL, MongoDB & Redis — Purpose: Follow slow user requests end-to-end across SSR, REST APIs, database queries, and BullMQ jobs. • ✓ Prometheus Metrics, Grafana Dashboards, Pino Log Aggregation, Sentry & On-Call Incident Runbooks — Purpose: Monitor heap usage, event-loop lag, HTTP 5xx spikes, and queue depth with instant alerting.

18.1 GitHub Actions CI/CD Pipelines: Typecheck, Lint, Testcontainers, Playwright, Trivy Scan & Zero-Downtime Deploy [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for GitHub Actions CI/CD Pipelines • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply GitHub Actions CI/CD Pipelines: Typecheck, Lint, Testcontainers, Playwright, Trivy Scan & Zero-Downtime Deploy in real Node.js, TypeScript & Moder...

What Topic Is Next → Next Topic 18.2: Infrastructure as Code Foundations with Terraform: Reproducible AWS Networking, S3, ECR & ECS Services

18.2 Infrastructure as Code Foundations with Terraform: Reproducible AWS Networking, S3, ECR & ECS Services [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Infrastructure as Code Foundations with Terraform • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Infrastructure as Code Foundations with Terraform: Reproducible AWS Networking, S3, ECR & ECS Services in real Node.js, TypeScript & Modern Full S...

What Topic Is Next → Next Topic 18.3: OpenTelemetry Distributed Tracing Across Next.js, NestJS/Express, PostgreSQL, MongoDB & Redis

18.3 OpenTelemetry Distributed Tracing Across Next.js, NestJS/Express, PostgreSQL, MongoDB & Redis [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for OpenTelemetry Distributed Tracing Across Next.js • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply OpenTelemetry Distributed Tracing Across Next.js, NestJS/Express, PostgreSQL, MongoDB & Redis in real Node.js, TypeScript & Modern Full Stack Engi...

What Topic Is Next → Next Topic 18.4: Prometheus Metrics, Grafana Dashboards, Pino Log Aggregation, Sentry & On-Call Incident Runbooks

18.4 Prometheus Metrics, Grafana Dashboards, Pino Log Aggregation, Sentry & On-Call Incident Runbooks [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Prometheus Metrics • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Prometheus Metrics, Grafana Dashboards, Pino Log Aggregation, Sentry & On-Call Incident Runbooks in real Node.js, TypeScript & Modern Full Stack E...

What Topic Is Next → Module 18 Capstone (scaleflow-m18-cicd-otel-observability) → Next Topic 19.1: Low-Level Design (LLD) & Machine Coding Rounds: Trello Board, Rate Limiter, Cart Engine & Pub/Sub Bus

Module 18 Hands-On Capstone Project (100 Marks): scaleflow-m18-cicd-otel-observability — Architect, implement, test, and push the production GitHub repository 'scaleflow-m18-cicd-otel-observability' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 18:...

MODULE 19 OF 22 • STAGE 4 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Full-Stack System Design (LLD + HLD) & Performance Engineering Interviews

Beginner-to-expert real-repository module covering Full-Stack System Design (LLD + HLD) & Performance Engineering Interviews with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Low-Level Design (LLD) & Machine Coding Rounds: Trello Board, Rate Limiter, Cart Engine & Pub/Sub Bus — Purpose: Excel in both frontend UI machine-coding rounds and backend TypeScript OOP design rounds. • ✓ Frontend System Design (HLD): Collaborative Docs, Real-Time Feeds, Video Dashboards & Offline-First Sync — Purpose: Articulate state normalization, optimistic updates, caching, and WebSocket scaling in senior interviews. • ✓ Backend Distributed System Design (HLD): Multi-Tenant SaaS Billing, Flash-Sale Inventory & Notification Cloud — Purpose: Design fault-tolerant PERN/MERN distributed systems handling 50k+ concurrent users without overselling. • ✓ Database Sharding, Read Replicas, CDN Edge Caching & Live Mock System Design Whiteboard Defense — Purpose: Defend SQL vs NoSQL trade-offs, consistency models, and bottleneck mitigations confidently.

19.1 Low-Level Design (LLD) & Machine Coding Rounds: Trello Board, Rate Limiter, Cart Engine & Pub/Sub Bus [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Low-Level Design (LLD) • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Low-Level Design (LLD) & Machine Coding Rounds: Trello Board, Rate Limiter, Cart Engine & Pub/Sub Bus in real Node.js, TypeScript & Modern Full St...

What Topic Is Next → Next Topic 19.2: Frontend System Design (HLD): Collaborative Docs, Real-Time Feeds, Video Dashboards & Offline-First Sync

19.2 Frontend System Design (HLD): Collaborative Docs, Real-Time Feeds, Video Dashboards & Offline-First Sync [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Frontend System Design (HLD) • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Frontend System Design (HLD): Collaborative Docs, Real-Time Feeds, Video Dashboards & Offline-First Sync in real Node.js, TypeScript & Modern Full...

What Topic Is Next → Next Topic 19.3: Backend Distributed System Design (HLD): Multi-Tenant SaaS Billing, Flash-Sale Inventory & Notification Cloud

19.3 Backend Distributed System Design (HLD): Multi-Tenant SaaS Billing, Flash-Sale Inventory & Notification Cloud [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Backend Distributed System Design (HLD) • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Backend Distributed System Design (HLD): Multi-Tenant SaaS Billing, Flash-Sale Inventory & Notification Cloud in real Node.js, TypeScript & Modern...

What Topic Is Next → Next Topic 19.4: Database Sharding, Read Replicas, CDN Edge Caching & Live Mock System Design Whiteboard Defense

19.4 Database Sharding, Read Replicas, CDN Edge Caching & Live Mock System Design Whiteboard Defense [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Database Sharding • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Database Sharding, Read Replicas, CDN Edge Caching & Live Mock System Design Whiteboard Defense in real Node.js, TypeScript & Modern Full Stack En...

What Topic Is Next → Module 19 Capstone (scaleflow-m19-fullstack-system-design) → Next Topic 20.1: End-to-End Delivery of ScaleFlow Dual-DB (PERN + MERN) Commerce, Billing & Real-Time Collaboration Cloud

Module 19 Hands-On Capstone Project (100 Marks): scaleflow-m19-fullstack-system-design — Architect, implement, test, and push the production GitHub repository 'scaleflow-m19-fullstack-system-design' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 19:...

MODULE 20 OF 22 • STAGE 4 • 21 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

ScaleFlow Live Client Production Cutover, Portfolio Defense & ≥90% Placement Sprint

Beginner-to-expert real-repository module covering ScaleFlow Live Client Production Cutover, Portfolio Defense & ≥90% Placement Sprint with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ End-to-End Delivery of ScaleFlow Dual-DB (PERN + MERN) Commerce, Billing & Real-Time Collaboration Cloud — Purpose: Ship Next.js 15, NestJS, PostgreSQL ledgers, MongoDB catalogs, Redis, and WebSockets in one live platform. • ✓ Playwright E2E Verification, OWASP Security Audit, k6 Load Test & Live AWS HTTPS Production Cutover — Purpose: Verify zero critical vulnerabilities, green E2E suites, and resilient auto-recovery under simulated load. • ✓ ATS-

Optimized Full-Stack TypeScript Resume, 20-Repo GitHub Portfolio & Live Architecture Walkthrough — Purpose: Stand out in product-company shortlists with deployed URLs, clean TypeScript PRs, and Internship credentials. • ✓ Timed DSA + Stack Assessments, Spoken MockAttempt AI Viva Loops & ≥90% Placement Guarantee Activation — Purpose: Achieve ≥90% across all topic mocks and AI vivas (unlimited re-attempts included) to unlock 100 job openings in 6 months.

20.1 End-to-End Delivery of ScaleFlow Dual-DB (PERN + MERN) Commerce, Billing & Real-Time Collaboration Cloud [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for End-to-End Delivery of ScaleFlow Dual-DB (PERN + MER... • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply End-to-End Delivery of ScaleFlow Dual-DB (PERN + MERN) Commerce, Billing & Real-Time Collaboration Cloud in real Node.js, TypeScript & Modern Full...

What Topic Is Next → Next Topic 20.2: Playwright E2E Verification, OWASP Security Audit, k6 Load Test & Live AWS HTTPS Production Cutover

20.2 Playwright E2E Verification, OWASP Security Audit, k6 Load Test & Live AWS HTTPS Production Cutover [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Playwright E2E Verification • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Playwright E2E Verification, OWASP Security Audit, k6 Load Test & Live AWS HTTPS Production Cutover in real Node.js, TypeScript & Modern Full Stac...

What Topic Is Next → Next Topic 20.3: ATS-Optimized Full-Stack TypeScript Resume, 20-Repo GitHub Portfolio & Live Architecture Walkthrough

20.3 ATS-Optimized Full-Stack TypeScript Resume, 20-Repo GitHub Portfolio & Live Architecture Walkthrough [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for ATS-Optimized Full-Stack TypeScript Resume • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply ATS-Optimized Full-Stack TypeScript Resume, 20-Repo GitHub Portfolio & Live Architecture Walkthrough in real Node.js, TypeScript & Modern Full Sta...

What Topic Is Next → Next Topic 20.4: Timed DSA + Stack Assessments, Spoken MockAttempt AI Viva Loops & ≥90% Placement Guarantee Activation

20.4 Timed DSA + Stack Assessments, Spoken MockAttempt AI Viva Loops & ≥90% Placement Guarantee Activation [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic NFSC Implementation • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full • Runnable Code Implementation (TYPESCRIPT) • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Timed DSA + Stack Assessments • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Timed DSA + Stack Assessments, Spoken MockAttempt AI Viva Loops & ≥90% Placement Guarantee Activation in real Node.js, TypeScript & Modern Full St...

What Topic Is Next → Module 20 Capstone (scaleflow-m20-live-client-production-capstone) → Next Topic 21.1: NestJS Domain Modules, Guards, Interceptors, Zod/Class-Validator DTOs & OpenAPI

Module 20 Hands-On Capstone Project (100 Marks): scaleflow-m20-live-client-production-capstone — Architect, implement, test, and push the production GitHub repository 'scaleflow-m20-live-client-production-capstone' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Mo...

MODULE 21 OF 22 • STAGE 4 • 14 LIVE + LAB HOURS • 3 TOPICS (24 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Enterprise NestJS Modular Monolith / Microservices, Dual-DB & BullMQ Queues

Self-explanatory list-and-bullet production mastery module covering Enterprise NestJS Modular Monolith / Microservices, Dual-DB & BullMQ Queues in Node.js, TypeScript & Modern Full Stack Engineering (MERN / PERN), paired with hands-on labs, capstone repository, Topic Mock Tests, and Spoken MockAttempt AI Viva.

Objectives: ✓ NestJS Domain Modules, Guards, Interceptors, Zod/Class-Validator DTOs & OpenAPI — Purpose: Enforce clean architectural boundaries and 100% typed contracts across large TypeScript teams. • ✓ Dual-Database Mastery: Transactional PostgreSQL Ledgers + High-Write MongoDB Streams — Purpose: Use ACID relational tables and flexible document collections together where each excels. • ✓ BullMQ Redis Job Pipelines, WebSockets Gateways & Sliding-Window Rate Limiting — Purpose: Process retries, scheduled jobs, and live collaborative events reliably at scale. • ✓ Build, verify, and document 'Type-Safe NestJS + PostgreSQL + MongoDB + Redis BullMQ Event Engine Repository' with clean Git commits and edge-case protection. • ✓ Clear 100% of Topic Timed Mock Tests and Spoken Voice + MockAttempt AI Technical Interviews.

21.1 NestJS Domain Modules, Guards, Interceptors, Zod/Class-Validator DTOs & OpenAPI [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Security Hygiene & Data Safety Rules for NestJS Domain Modules, Guards, Interceptors, Zod/Class • Performance Optimization, Complexity & Latency Impact of NestJS Domain Modules, Guards, Interceptors, Zod/Class • Live Debugging, Stack • Runnable Code Implementation (TYPESCRIPT) • Hands • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for NestJS Domain Modules • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply NestJS Domain Modules, Guards, Interceptors, Zod/Class-Validator DTOs & OpenAPI in real Node.js, TypeScript & Modern Full Stack Engineering (MERN...

What Topic Is Next → Next Topic 21.2: Dual-Database Mastery: Transactional PostgreSQL Ledgers + High-Write MongoDB Streams

21.2 Dual-Database Mastery: Transactional PostgreSQL Ledgers + High-Write MongoDB Streams [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Security Hygiene & Data Safety Rules for Dual • Performance Optimization, Complexity & Latency Impact of Dual • Live Debugging, Stack • Runnable Code Implementation (TYPESCRIPT) • Hands • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Dual-Database Mastery • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Dual-Database Mastery: Transactional PostgreSQL Ledgers + High-Write MongoDB Streams in real Node.js, TypeScript & Modern Full Stack Engineering (...

What Topic Is Next → Next Topic 21.3: BullMQ Redis Job Pipelines, WebSockets Gateways & Sliding-Window Rate Limiting

21.3 BullMQ Redis Job Pipelines, WebSockets Gateways & Sliding-Window Rate Limiting [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Security Hygiene & Data Safety Rules for BullMQ Redis Job Pipelines, WebSockets Gateways & Sliding • Performance Optimization, Complexity & Latency Impact of BullMQ Redis Job Pipelines, WebSockets Gateways & Sliding • Live Debugging, Stack • Runnable Code Implementation (TYPESCRIPT) • Hands • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for BullMQ Redis Job Pipelines • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply BullMQ Redis Job Pipelines, WebSockets Gateways & Sliding-Window Rate Limiting in real Node.js, TypeScript & Modern Full Stack Engineering (MERN /...

What Topic Is Next → Module 21 Capstone (Type-Safe NestJS + PostgreSQL + MongoDB + Redis BullMQ Event Engine Repository) → Next Topic 22.1: Next.js 15 Server Components, Streaming Suspense, Caching & Core Web Vitals Tuning

Module 21 Hands-On Capstone Project (100 Marks): Type-Safe NestJS + PostgreSQL + MongoDB + Redis BullMQ Event Engine Repository — Architect, implement, test, and document 'Type-Safe NestJS + PostgreSQL + MongoDB + Redis BullMQ Event Engine Repository' applying every bullet-wise engineering concept from Module 21 (Enter...

MODULE 22 OF 22 • STAGE 4 • 14 LIVE + LAB HOURS • 3 TOPICS (24 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Next.js 15 App Router, Cloud CI/CD & ≥90% Placement Gate System Design

Self-explanatory list-and-bullet production mastery module covering Next.js 15 App Router, Cloud CI/CD & ≥90% Placement Gate System Design in Node.js, TypeScript & Modern Full Stack Engineering (MERN / PERN), paired with hands-on labs, capstone repository, Topic Mock Tests, and Spoken MockAttempt AI Viva.

Objectives: ✓ Next.js 15 Server Components, Streaming Suspense, Caching & Core Web Vitals Tuning — Purpose: Ship SEO-dominant, fast-hydrating full-stack TypeScript web platforms. • ✓ Playwright/Vitest Automated Suites, Multi-Stage Dockerfiles & AWS HTTPS CI/CD — Purpose: Gate every pull request with automated E2E/API tests and zero-downtime cloud deploys. • ✓ MERN/PERN Full Stack System Design, Live Machine Coding & Placement Defense — Purpose: Ace live React/Next.js machine coding, Node event-loop deep dives, and system design rounds. • ✓ Build, verify, and document 'Next.js 15 RSC + Server Actions + NestJS Docker AWS Production Repository' with clean Git commits and edge-case protection. • ✓ Clear 100% of Topic Timed Mock Tests and Spoken Voice + MockAttempt AI Technical Interviews.

22.1 Next.js 15 Server Components, Streaming Suspense, Caching & Core Web Vitals Tuning [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Security Hygiene & Data Safety Rules for Next.js 15 Server Components, Streaming Suspense, Caching & Core Web Vital... • Performance Optimization, Complexity & Latency Impact of Next.js 15 Server Components, Streaming Suspense, Caching.... • Live Debugging, Stack • Runnable Code Implementation (TYPESCRIPT) • Hands • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Next.js 15 Server Components • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Next.js 15 Server Components, Streaming Suspense, Caching & Core Web Vitals Tuning in real Node.js, TypeScript & Modern Full Stack Engineering (ME...

What Topic Is Next → Next Topic 22.2: Playwright/Vitest Automated Suites, Multi-Stage Dockerfiles & AWS HTTPS CI/CD

22.2 Playwright/Vitest Automated Suites, Multi-Stage Dockerfiles & AWS HTTPS CI/CD [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Security Hygiene & Data Safety Rules for Playwright/Vitest Automated Suites, Multi • Performance Optimization, Complexity & Latency Impact of Playwright/Vitest Automated Suites, Multi • Live Debugging, Stack • Runnable Code Implementation (TYPESCRIPT) • Hands • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for Playwright/Vitest Automated Suites • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply Playwright/Vitest Automated Suites, Multi-Stage Dockerfiles & AWS HTTPS CI/CD in real Node.js, TypeScript & Modern Full Stack Engineering (MERN /...

What Topic Is Next → Next Topic 22.3: MERN/PERN Full Stack System Design, Live Machine Coding & Placement Defense

22.3 MERN/PERN Full Stack System Design, Live Machine Coding & Placement Defense [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Security Hygiene & Data Safety Rules for MERN/PERN Full Stack System Design, Live Machine Coding & Placement Defens... • Performance Optimization, Complexity & Latency Impact of MERN/PERN Full Stack System Design, Live Machine Coding &.... • Live Debugging, Stack • Runnable Code Implementation (TYPESCRIPT) • Hands • Idiomatic NFSC Implementation: strict TypeScript contracts, non-blocking I/O, idempotency keys & Redis cachin... • Node.js 22 V8 + Libuv Event Loop, Streams & React/Next.js Full-Stack Execution & Memory Lifecycle for MERN/PERN Full Stack System Design • Production Hardening & Security: HttpOnly rotating JWTs, Zod boundary schemas, Helmet & OWASP BOLA guards

Purpose: To understand and apply MERN/PERN Full Stack System Design, Live Machine Coding & Placement Defense in real Node.js, TypeScript & Modern Full Stack Engineering (MERN / PE...

What Topic Is Next → Module 22 Capstone → End-of-Course Live Client Project: ScaleFlow Dual-Database (PERN + MERN) Real-Time Commerce, Billing & Collaboration Cloud

Module 22 Hands-On Capstone Project (100 Marks): Next.js 15 RSC + Server Actions + NestJS Docker AWS Production Repository — Architect, implement, test, and document 'Next.js 15 RSC + Server Actions + NestJS Docker AWS Production Repository' applying every bullet-wise engineering concept from Module 22 (Next.js 15...

3. LIVE CLIENT PRODUCTION PROJECT & CAREER SCOPE

End-of-Course Live Client Production Project + Official Internship Confirmation Letter

Project: ScaleFlow Dual-Database (PERN + MERN) Real-Time Commerce, Billing & Collaboration Cloud • **Confirmed Role:** Full Stack Node.js, NestJS, Next.js 15 & Cloud Engineering Resident

Domain & Production Infra: High-Growth B2B SaaS & Real-Time Commerce Client (Node.js 22 + NestJS + TypeScript 5 + Next.js 15 + PostgreSQL + MongoDB + Redis + AWS)

- Built modular NestJS & Express TypeScript APIs backed by PostgreSQL ACID ledgers and MongoDB catalog aggregations.
- Engineered Next.js 15 & React 19 Server/Client interfaces with WebSockets live sync and optimistic mutations.
- Implemented rotating HttpOnly JWT security, Redis Lua sliding-window rate limiters, and BullMQ background queues.
- Shipped containerized multi-service builds via GitHub Actions CI/CD onto AWS ALB + ECS/EC2 HTTPS infrastructure.

Target Job Roles: Full Stack Engineer — MERN / PERN (TypeScript), Next.js 15 & Node.js / NestJS Product Engineer, Software Development Engineer — Full Stack, Backend & Real-Time Systems Developer

Realistic Annual CTC Scope: **Placement-Guaranteed Graduate / SDE-1 (0–1 Yr):** ₹6.0 LPA – ₹13.5 LPA **Senior Full Stack TypeScript SDE-2 (1–3 Yrs):** ₹14.0 LPA – ₹27.0 LPA **Staff Full Stack & Cloud Architect (3–6+ Yrs):** ₹28.0 LPA – ₹54.0+ LPA

Optional Worthwhile Global Certifications (100% Exam-Success Mentorship Available): **AWS Certified Developer – Associate (DVA-C02):** ₹10.5 LPA – ₹24.0 LPA (+25–35% Cloud CTC Boost) • **OpenJS Node.js Application Developer (JSNAD):** ₹12.0 LPA – ₹27.0 LPA (Hands-On Coding Exam) • **MongoDB Certified Associate Developer (Node.js) (C100DEV):** ₹10.0 LPA – ₹23.0 LPA (Official Vendor Badge)

MockAttempt Academy Pvt. Ltd. • Email: support@mockattempt.com • Phone/WhatsApp: +91 95400 07917 • Web: https://mockattempt.com/training/nodejs-full-stack-engineering-program

Try 2 Free Topic-Wise Timed Topic Mock Tests & Spoken MockAttempt AI Interviews daily on every topic above before enrolling.