MockAttempt

TECH ACADEMY • OFFICIAL COURSE SYLLABUS BROCHURE

MockAttempt Academy Pvt. Ltd. • Live Architect Mentorship, AI Assessments & Client Production

Admissions & Support: support@mockattempt.com
Phone / WhatsApp: +91 95400 07917
Course Code: JFSC • **Updated:** 05 Nov 2026
https://mockattempt.com/training/java-full-stack-career-program

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

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

Java Full Stack Development & Enterprise Cloud Architecture

Complete Zero-to-Expert Core Java to Java 21 LTS + OOP + DSA + SQL + HTML/CSS/React 19 + Spring Boot 3.3 + Spring AI + Kafka + AWS EKS + DevOps | 100% Placement Guaranteed* (6 Months Fees Refund*)

Duration: 32w (480h) • Weekend Only (2h/Day) (508 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: ₹34,999 (Net Tuition ₹29,660 + Included 18% GST ₹5,339) • **Split Option:** ₹5,333/mo × 6 (0% Split) • **Effective ROI:** ~₹69/Guided Hr (~₹407/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) • 32 Weeks (508 Guided Live + Lab Hours))

✓ **Trainer LeetCode & HackerRank Mentorship (Core Java to Java 21 LTS, 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: Core Java to Java 21 LTS • OOP, Collections & Java DSA • Spring Boot 3.3 & Spring AI • React 19, Kafka & AWS EKS • SQL From Scratch & PostgreSQL 16 • JDBC, Maven & JUnit 5 TDD • HTML5, CSS3, Tailwind & TypeScript 5 • pgvector / Enterprise RAG • Spring Security 6 & OAuth2 • Hibernate JPA & Flyway • Apache Kafka & Sagas • Redis 7 Distributed Caching • Resilience4j Fault Tolerance • Docker & Multi-Stage Builds • Kubernetes (EKS) • AWS Cloud (VPC, ALB, RDS, S3) • Terraform IaC • GitHub Actions CI/CD • OpenTelemetry & 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 Core Java to Java 21 LTS, 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. Full-lifecycle mastery: Java 21 LTS, Spring Boot 3.3, Spring Security 6 OAuth2, Hibernate/PostgreSQL, and React 19 + TypeScript 5.

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 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Programming Foundations, Git, Linux CLI & First Core Java Programs

Beginner-to-expert real-repository module covering Programming Foundations, Git, Linux CLI & First Core Java Programs with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ How Computers Execute Code, JDK 21 / JRE / JVM Setup, IntelliJ IDEA & First Hello World — Purpose: Understand bytecode compilation, install JDK 21 LTS and IntelliJ IDEA from scratch, and run your first verified Java programs. • ✓ Linux Terminal Essentials, Shell Navigation & Professional Git / GitHub Branching Workflows — Purpose: Navigate Linux directories, manage file permissions, and commit clean feature branches and pull requests on GitHub from Day 1. • ✓ Variables, Primitive Data Types, Type Casting, Arithmetic / Bitwise / Logical Operators & Scanner I/O — Purpose: Store numeric, character, and boolean values accurately while avoiding integer overflow and implicit narrowing bugs. • ✓ Control Flow Mastery: if-else, Switch Expressions, for / while / do-while Loops & Nested Pattern Logic — Purpose: Build strong logical thinking through structured branching, dry-run trace tables, and 25+ loop and number/pattern problems.

1.1 How Computers Execute Code, JDK 21 / JRE / JVM Setup, IntelliJ IDEA & First Hello World

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

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for How Computers Execut... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for How Computers Execute Code • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply How Computers Execute Code, JDK 21 / JRE / JVM Setup, IntelliJ IDEA & First Hello World in real Java Full Stack Development & Enterprise Cloud Arc...

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

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

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

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Linux Terminal Essen... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Linux Terminal Essentials • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Linux Terminal Essentials, Shell Navigation & Professional Git / GitHub Branching Workflows in real Java Full Stack Development & Enterprise Cloud...

What Topic Is Next → Next Topic 1.3: Variables, Primitive Data Types, Type Casting, Arithmetic / Bitwise / Logical Operators & Scanner I/O

1.3 Variables, Primitive Data Types, Type Casting, Arithmetic / Bitwise / Logical Operators & Scanner I/O

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

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Variables • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Variables, Primitive Data Types, Type Casting, Arithmetic / Bitwise / Logical Operators & Scanner I/O in real Java Full Stack Development & Enterp...

What Topic Is Next → Next Topic 1.4: Control Flow Mastery: if-else, Switch Expressions, for / while / do-while Loops & Nested Pattern Logic

1.4 Control Flow Mastery: if-else, Switch Expressions, for / while / do-while Loops & Nested Pattern Logic

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

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Control Flow Mastery • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Control Flow Mastery: if-else, Switch Expressions, for / while / do-while Loops & Nested Pattern Logic in real Java Full Stack Development & Enter...

What Topic Is Next → Module 1 Capstone (fincore-m01-java-foundations-cli) → Next Topic 2.1: 1D, 2D & Jagged Arrays: Contiguous Memory Layout, Traversals, Sorting, Searching & Matrix Operations

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

MODULE 2 OF 22 • STAGE 1 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Arrays, Strings, Stack vs Heap Memory, Methods & Recursive Problem Solving

Beginner-to-expert real-repository module covering Arrays, Strings, Stack vs Heap Memory, Methods & Recursive Problem Solving with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ 1D, 2D & Jagged Arrays: Contiguous Memory Layout, Traversals, Sorting, Searching & Matrix Operations — Purpose: Store and manipulate indexed collections in O(1) random access and solve real matrix and subarray problems cleanly. • ✓ Java Strings Deep Dive: String Constant Pool, Immutability, StringBuilder, StringBuffer & Regex Validation — Purpose: Avoid O(N^2) string concatenation traps in loops and validate emails, PANs, mobile numbers, and payloads using regular expressions. • ✓ Modular Methods, Pass-by-Value Mechanics, Method Overloading, Varargs & Stack Frame Execution — Purpose: Decompose complex programs into clean, reusable, single-responsibility Java methods with predictable parameter semantics. • ✓ Stack vs Heap Memory Model, Object Reference Lifecycle, Recursion Call Trees & Base-Case Debugging — Purpose: Trace exactly how local stack frames and heap objects behave during method calls and recursive divide-and-conquer execution.

2.1 1D, 2D & Jagged Arrays: Contiguous Memory Layout, Traversals, Sorting, Searching & Matrix Operations [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Jagged Arrays • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply 1D, 2D & Jagged Arrays: Contiguous Memory Layout, Traversals, Sorting, Searching & Matrix Operations in real Java Full Stack Development & Enterpr...

What Topic Is Next → Next Topic 2.2: Java Strings Deep Dive: String Constant Pool, Immutability, StringBuilder, StringBuffer & Regex Validation

2.2 Java Strings Deep Dive: String Constant Pool, Immutability, StringBuilder, StringBuffer & Regex Validation [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Java Strings Deep Di... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Java Strings Deep Dive • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Java Strings Deep Dive: String Constant Pool, Immutability, StringBuilder, StringBuffer & Regex Validation in real Java Full Stack Development & E...

What Topic Is Next → Next Topic 2.3: Modular Methods, Pass-by-Value Mechanics, Method Overloading, Varargs & Stack Frame Execution

2.3 Modular Methods, Pass-by-Value Mechanics, Method Overloading, Varargs & Stack Frame Execution [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Modular Methods • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Modular Methods, Pass-by-Value Mechanics, Method Overloading, Varargs & Stack Frame Execution in real Java Full Stack Development & Enterprise Clo...

What Topic Is Next → Next Topic 2.4: Stack vs Heap Memory Model, Object Reference Lifecycle, Recursion Call Trees & Base-Case Debugging

2.4 Stack vs Heap Memory Model, Object Reference Lifecycle, Recursion Call Trees & Base-Case Debugging [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Stack vs Heap Memory... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Stack vs Heap Memory Model • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Stack vs Heap Memory Model, Object Reference Lifecycle, Recursion Call Trees & Base-Case Debugging in real Java Full Stack Development & Enterpris...

What Topic Is Next → Module 2 Capstone (fincore-m02-arrays-strings-memory) → Next Topic 3.1: Classes, Objects, Constructors, Constructor Chaining, this Keyword & static Class Members

Module 2 Hands-On Capstone Project (100 Marks): fincore-m02-arrays-strings-memory — Architect, implement, test, and push the production GitHub repository 'fincore-m02-arrays-strings-memory' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 2: Arra...

MODULE 3 OF 22 • STAGE 1 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Complete Object-Oriented Programming (OOP), Packages & SOLID Design Principles

Beginner-to-expert real-repository module covering Complete Object-Oriented Programming (OOP), Packages & SOLID Design Principles with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Classes, Objects, Constructors, Constructor Chaining, this Keyword & static Class Members — Purpose: Model real-world banking accounts, customers, and transactions using clean object blueprints and lifecycle constructors. • ✓ Encapsulation, Access Modifiers (private, default, protected, public), Packages & Data Invariants — Purpose: Protect domain state from invalid external mutation by enforcing validation inside constructors and domain methods. • ✓ Inheritance, super Keyword, Method Overriding, Dynamic Method Dispatch & Runtime Polymorphism — Purpose: Extend shared behavior cleanly and leverage virtual method invocation without brittle type-checking if-else chains. • ✓ Abstract Classes, Interfaces, Composition vs Inheritance & All 5 SOLID Principles in Enterprise Java — Purpose: Architect loosely coupled, extensible business services using interface contracts and Dependency Inversion from first principles.

3.1 Classes, Objects, Constructors, Constructor Chaining, this Keyword & static Class Members [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Classes • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Classes, Objects, Constructors, Constructor Chaining, this Keyword & static Class Members in real Java Full Stack Development & Enterprise Cloud A...

What Topic Is Next → Next Topic 3.2: Encapsulation, Access Modifiers (private, default, protected, public), Packages & Data Invariants

3.2 Encapsulation, Access Modifiers (private, default, protected, public), Packages & Data Invariants [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Encapsulation • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Encapsulation, Access Modifiers (private, default, protected, public), Packages & Data Invariants in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 3.3: Inheritance, super Keyword, Method Overriding, Dynamic Method Dispatch & Runtime Polymorphism

3.3 Inheritance, super Keyword, Method Overriding, Dynamic Method Dispatch & Runtime Polymorphism [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Inheritance • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Inheritance, super Keyword, Method Overriding, Dynamic Method Dispatch & Runtime Polymorphism in real Java Full Stack Development & Enterprise Clo...

What Topic Is Next → Next Topic 3.4: Abstract Classes, Interfaces, Composition vs Inheritance & All 5 SOLID Principles in Enterprise Java

3.4 Abstract Classes, Interfaces, Composition vs Inheritance & All 5 SOLID Principles in Enterprise Java [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Abstract Classes • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Abstract Classes, Interfaces, Composition vs Inheritance & All 5 SOLID Principles in Enterprise Java in real Java Full Stack Development & Enterpr...

What Topic Is Next → Module 3 Capstone (fincore-m03-oop-solid-banking-core) → Next Topic 4.1: Checked vs Unchecked Exceptions, try-catch-finally, try-with-resources & Custom Domain Exceptions

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

MODULE 4 OF 22 • STAGE 1 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Exception Handling, File I/O (NIO.2), Generics & Complete Java Collections Framework

Beginner-to-expert real-repository module covering Exception Handling, File I/O (NIO.2), Generics & Complete Java Collections Framework with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Checked vs Unchecked Exceptions, try-catch-finally, try-with-resources & Custom Domain Exceptions — Purpose: Prevent sudden JVM crashes by handling recoverable I/O errors cleanly and throwing rich business-rule exceptions. • ✓ Java File I/O Streams, Buffered Readers, Serialization & Modern NIO.2 Path / Files Streaming — Purpose: Read and write CSV, JSON, and large audit log files safely with deterministic AutoCloseable resource cleanup. • ✓ Java Generics From Scratch: Type Parameters, Bounded Wildcards (? extends / ? super) & Type Erasure — Purpose: Write reusable, compile-time type-safe repositories and utilities with zero unsafe ClassCastException failures. • ✓ Complete Collections Framework: ArrayList, LinkedList, HashSet, TreeSet, PriorityQueue, ArrayDeque & HashMap Internals — Purpose: Select exact O(1) vs O(log N) collections and master equals/hashCode contracts, load factors, and bucket collision trees.

4.1 Checked vs Unchecked Exceptions, try-catch-finally, try-with-resources & Custom Domain Exceptions [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Checked vs Unchecked... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Checked vs Unchecked Exceptions • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Checked vs Unchecked Exceptions, try-catch-finally, try-with-resources & Custom Domain Exceptions in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 4.2: Java File I/O Streams, Buffered Readers, Serialization & Modern NIO.2 Path / Files Streaming

4.2 Java File I/O Streams, Buffered Readers, Serialization & Modern NIO.2 Path / Files Streaming [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Java File I/O Stream... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs,

@Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Java File I/O Streams • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Java File I/O Streams, Buffered Readers, Serialization & Modern NIO.2 Path / Files Streaming in real Java Full Stack Development & Enterprise Clou...

What Topic Is Next → Next Topic 4.3: Java Generics From Scratch: Type Parameters, Bounded Wildcards (? extends / ? super) & Type Erasure

4.3 Java Generics From Scratch: Type Parameters, Bounded Wildcards (? extends / ? super) & Type Erasure [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Java Generics From S... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Java Generics From Scratch • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Java Generics From Scratch: Type Parameters, Bounded Wildcards (? extends / ? super) & Type Erasure in real Java Full Stack Development & Enterpri...

What Topic Is Next → Next Topic 4.4: Complete Collections Framework: ArrayList, LinkedList, HashSet, TreeSet, PriorityQueue, ArrayDeque & HashMap Internals

4.4 Complete Collections Framework: ArrayList, LinkedList, HashSet, TreeSet, PriorityQueue, ArrayDeque & HashMap Internals [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Complete Collections... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Complete Collections Framework • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Complete Collections Framework: ArrayList, LinkedList, HashSet, TreeSet, PriorityQueue, ArrayDeque & HashMap Internals in real Java Full Stack Dev...

What Topic Is Next → Module 4 Capstone (fincore-m04-collections-generics-io) → Next Topic 5.1: Functional Interfaces (Predicate, Function, Consumer, Supplier), Lambda Expressions & Optional<T>

Module 4 Hands-On Capstone Project (100 Marks): fincore-m04-collections-generics-io — Architect, implement, test, and push the production GitHub repository 'fincore-m04-collections-generics-io' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 4: Ex...

MODULE 5 OF 22 • STAGE 1 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Modern Java 8 to Java 21 LTS: Lambdas, Streams, Multithreading & Virtual Threads

Beginner-to-expert real-repository module covering Modern Java 8 to Java 21 LTS: Lambdas, Streams, Multithreading & Virtual Threads with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Functional Interfaces (Predicate, Function, Consumer, Supplier), Lambda Expressions & Optional<T> — Purpose: Write concise declarative logic and eliminate billion-dollar NullPointerException bugs using Optional pipelines. • ✓ Java Stream API Pipelines, filter / map / flatMap, Collectors, BigDecimal Math & java.time UTC/IST — Purpose: Aggregate enterprise financial ledgers with exact BigDecimal rounding and timezone-safe Instant/ZonedDateTime APIs. • ✓ Classic Multithreading: Thread Lifecycle, synchronized, volatile, ReentrantLock, ExecutorService & CompletableFuture — Purpose: Coordinate multi-core background tasks safely without race conditions, deadlocks, or thread-pool starvation. • ✓ Java 17–21 LTS Mastery: Records, Sealed Classes, Pattern Matching for Switch, G1GC/ZGC & Virtual Threads — Purpose: Upgrade from classic Java to ultra-scalable Project Loom virtual threads and exhaustive immutable domain modeling.

5.1 Functional Interfaces (Predicate, Function, Consumer, Supplier), Lambda Expressions & Optional<T> [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Functional Interface... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Functional Interfaces (Predicate, Function, Consumer... • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Functional Interfaces (Predicate, Function, Consumer, Supplier), Lambda Expressions & Optional<T> in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 5.2: Java Stream API Pipelines, filter / map / flatMap, Collectors, BigDecimal Math & java.time UTC/IST

5.2 Java Stream API Pipelines, filter / map / flatMap, Collectors, BigDecimal Math & java.time UTC/IST [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Java Stream API Pipe... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Java Stream API Pipelines • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Java Stream API Pipelines, filter / map / flatMap, Collectors, BigDecimal Math & java.time UTC/IST in real Java Full Stack Development & Enterpris...

What Topic Is Next → Next Topic 5.3: Classic Multithreading: Thread Lifecycle, synchronized, volatile, ReentrantLock, ExecutorService & CompletableFuture

5.3 Classic Multithreading: Thread Lifecycle, synchronized, volatile, ReentrantLock, ExecutorService & CompletableFuture [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Classic Multithreadi... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs,

@Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Classic Multithreading • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Classic Multithreading: Thread Lifecycle, synchronized, volatile, ReentrantLock, ExecutorService & CompletableFuture in real Java Full Stack Devel...

What Topic Is Next → Next Topic 5.4: Java 17–21 LTS Mastery: Records, Sealed Classes, Pattern Matching for Switch, G1GC/ZGC & Virtual Threads

5.4 Java 17–21 LTS Mastery: Records, Sealed Classes, Pattern Matching for Switch, G1GC/ZGC & Virtual Threads [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Java 17–21 LTS Maste... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Java 17–21 LTS Mastery • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Java 17–21 LTS Mastery: Records, Sealed Classes, Pattern Matching for Switch, G1GC/ZGC & Virtual Threads in real Java Full Stack Development & Ent...

What Topic Is Next → Module 5 Capstone (fincore-m05-modern-java21-concurrency) → Next Topic 6.1: Asymptotic Big-O Time & Space Analysis, Two Pointers, Sliding Window, Prefix Sums & Binary Search

Module 5 Hands-On Capstone Project (100 Marks): fincore-m05-modern-java21-concurrency — Architect, implement, test, and push the production GitHub repository 'fincore-m05-modern-java21-concurrency' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 5:...

MODULE 6 OF 22 • STAGE 2 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Data Structures, Algorithms (DSA) & Coding Round Problem Solving in Java

Beginner-to-expert real-repository module covering Data Structures, Algorithms (DSA) & Coding Round Problem Solving in Java with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Asymptotic Big-O Time & Space Analysis, Two Pointers, Sliding Window, Prefix Sums & Binary Search — Purpose: Optimize brute-force O(N^2) loops into O(N) or O(log N) solutions to clear Round-1 online coding assessments. • ✓ Singly / Doubly Linked Lists, Monotonic Stacks, Queues, Hashing Patterns & Thread-Safe LRU Cache — Purpose: Master pointer manipulation, next-greater-element stack patterns, and high-frequency SDE-1/SDE-2 interview data structures. • ✓ Binary Trees, Binary Search Trees (BST), Heaps / PriorityQueues, Tries & Recursive Backtracking — Purpose: Solve hierarchical tree traversals, top-K streaming problems, prefix autocomplete, and combinatorial search cleanly. • ✓ Graph Algorithms (BFS, DFS, Topological Sort, Dijkstra) & Dynamic Programming (Memoization to Tabulation) — Purpose: Tackle dependency resolution, shortest-path routing, and optimal substructure DP problems asked by top product firms.

6.1 Asymptotic Big-O Time & Space Analysis, Two Pointers, Sliding Window, Prefix Sums & Binary Search [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Asymptotic Big • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Asymptotic Big-O Time • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Asymptotic Big-O Time & Space Analysis, Two Pointers, Sliding Window, Prefix Sums & Binary Search in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 6.2: Singly / Doubly Linked Lists, Monotonic Stacks, Queues, Hashing Patterns & Thread-Safe LRU Cache

6.2 Singly / Doubly Linked Lists, Monotonic Stacks, Queues, Hashing Patterns & Thread-Safe LRU Cache [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Singly / Doubly Link... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Singly / Doubly Linked Lists • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Singly / Doubly Linked Lists, Monotonic Stacks, Queues, Hashing Patterns & Thread-Safe LRU Cache in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 6.3: Binary Trees, Binary Search Trees (BST), Heaps / PriorityQueues, Tries & Recursive Backtracking

6.3 Binary Trees, Binary Search Trees (BST), Heaps / PriorityQueues, Tries & Recursive Backtracking [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Binary Trees • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Binary Trees, Binary Search Trees (BST), Heaps / PriorityQueues, Tries & Recursive Backtracking in real Java Full Stack Development & Enterprise C...

What Topic Is Next → Next Topic 6.4: Graph Algorithms (BFS, DFS, Topological Sort, Dijkstra) & Dynamic Programming (Memoization to Tabulation)

6.4 Graph Algorithms (BFS, DFS, Topological Sort, Dijkstra) & Dynamic Programming (Memoization to Tabulation) [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Graph Algorithms (BF... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Graph Algorithms (BFS,

DFS, Topological Sort, Dijkst... • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Graph Algorithms (BFS, DFS, Topological Sort, Dijkstra) & Dynamic Programming (Memoization to Tabulation) in real Java Full Stack Development & En...

What Topic Is Next → Module 6 Capstone (fincore-m06-java-dsa-problem-solving) → Next Topic 7.1: RDBMS Fundamentals, ER Diagrams, 1NF / 2NF / 3NF Normalization & Core SQL (DDL, DML, Constraints)

Module 6 Hands-On Capstone Project (100 Marks): fincore-m06-java-dsa-problem-solving — Architect, implement, test, and push the production GitHub repository 'fincore-m06-java-dsa-problem-solving' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 6: D...

MODULE 7 OF 22 • STAGE 2 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Relational Databases & SQL From Scratch to PostgreSQL 16 Production Tuning

Beginner-to-expert real-repository module covering Relational Databases & SQL From Scratch to PostgreSQL 16 Production Tuning with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ RDBMS Fundamentals, ER Diagrams, 1NF / 2NF / 3NF Normalization & Core SQL (DDL, DML, Constraints) — Purpose: Design clean relational schemas from scratch with primary keys, foreign keys, unique constraints, and check guards. • ✓ SQL Filtering, INNER / LEFT / RIGHT / FULL JOINs, Subqueries, GROUP BY, HAVING & Analytical Views — Purpose: Write accurate multi-table reporting and reconciliation queries across customers, orders, wallets, and audit logs. • ✓ Advanced SQL: Common Table Expressions (CTEs), Window Functions (ROW_NUMBER, RANK, LAG) & ACID Isolation — Purpose: Compute running balances, cohort rankings, and race-free concurrent financial transfers using SQL transactions. • ✓ PostgreSQL 16 MVCC Internals, B-Tree / GIN JSONB Indexing, EXPLAIN ANALYZE Tuning & pgvector Search — Purpose: Turn 800ms sequential table scans into sub-3ms index scans and store HNSW vector embeddings inside PostgreSQL.

7.1 RDBMS Fundamentals, ER Diagrams, 1NF / 2NF / 3NF Normalization & Core SQL (DDL, DML, Constraints) [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for RDBMS Fundamentals • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply RDBMS Fundamentals, ER Diagrams, 1NF / 2NF / 3NF Normalization & Core SQL (DDL, DML, Constraints) in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 7.2: SQL Filtering, INNER / LEFT / RIGHT / FULL JOINs, Subqueries, GROUP BY, HAVING & Analytical Views

7.2 SQL Filtering, INNER / LEFT / RIGHT / FULL JOINs, Subqueries, GROUP BY, HAVING & Analytical Views [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for SQL Filtering • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply SQL Filtering, INNER / LEFT / RIGHT / FULL JOINs, Subqueries, GROUP BY, HAVING & Analytical Views in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 7.3: Advanced SQL: Common Table Expressions (CTEs), Window Functions (ROW_NUMBER, RANK, LAG) & ACID Isolation

7.3 Advanced SQL: Common Table Expressions (CTEs), Window Functions (ROW_NUMBER, RANK, LAG) & ACID Isolation [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Advanced SQL • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Advanced SQL: Common Table Expressions (CTEs), Window Functions (ROW_NUMBER, RANK, LAG) & ACID Isolation in real Java Full Stack Development & Ent...

What Topic Is Next → Next Topic 7.4: PostgreSQL 16 MVCC Internals, B-Tree / GIN JSONB Indexing, EXPLAIN ANALYZE Tuning & pgvector Search

7.4 PostgreSQL 16 MVCC Internals, B-Tree / GIN JSONB Indexing, EXPLAIN ANALYZE Tuning & pgvector Search [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for PostgreSQL 16 MVCC I... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for PostgreSQL 16 MVCC Internals • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply PostgreSQL 16 MVCC Internals, B-Tree / GIN JSONB Indexing, EXPLAIN ANALYZE Tuning & pgvector Search in real Java Full Stack Development & Enterpri...

What Topic Is Next → Module 7 Capstone (fincore-m07-sql-postgresql-pgvector) → Next Topic 8.1: JDBC Architecture From Scratch: Connections, PreparedStatement, SQL Injection Defense & Batch Updates

Module 7 Hands-On Capstone Project (100 Marks): fincore-m07-sql-postgresql-pgvector — Architect, implement, test, and push the production GitHub repository 'fincore-m07-sql-postgresql-pgvector' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 7: Re...

MODULE 8 OF 22 • STAGE 2 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

JDBC Database Connectivity, Maven / Gradle, JUnit 5 TDD & GoF Design Patterns

Beginner-to-expert real-repository module covering JDBC Database Connectivity, Maven / Gradle, JUnit 5 TDD & GoF Design Patterns with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ JDBC Architecture From Scratch: Connections, PreparedStatement, SQL Injection Defense & Batch Updates — Purpose: Understand exactly how Java talks to relational databases over TCP sockets before touching any ORM framework. • ✓ HikariCP Connection Pooling, DAO / Repository Abstractions & Versioned Flyway SQL Schema Migrations — Purpose: Manage bounded database connection pools and evolve schemas reproducibly across dev, staging, and production. • ✓ Maven & Gradle Build Lifecycles, Dependency Scopes, SLF4J / Logback Logging & JUnit 5 + Mockito TDD — Purpose: Package multi-module Java builds, emit structured logs, and verify business logic with automated unit tests and mocks. • ✓ Gang of Four (GoF) Design Patterns in Java: Singleton, Factory, Builder, Strategy, Observer, Adapter & Proxy — Purpose: Recognize and apply battle-tested creational, structural, and behavioral patterns used inside Spring and enterprise SDKs.

8.1 JDBC Architecture From Scratch: Connections, PreparedStatement, SQL Injection Defense & Batch Updates [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for JDBC Architecture Fr... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for JDBC Architecture From Scratch • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply JDBC Architecture From Scratch: Connections, PreparedStatement, SQL Injection Defense & Batch Updates in real Java Full Stack Development & Enterp...

What Topic Is Next → Next Topic 8.2: HikariCP Connection Pooling, DAO / Repository Abstractions & Versioned Flyway SQL Schema Migrations

8.2 HikariCP Connection Pooling, DAO / Repository Abstractions & Versioned Flyway SQL Schema Migrations [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for HikariCP Connection... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for HikariCP Connection Pooling • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply HikariCP Connection Pooling, DAO / Repository Abstractions & Versioned Flyway SQL Schema Migrations in real Java Full Stack Development & Enterpri...

What Topic Is Next → Next Topic 8.3: Maven & Gradle Build Lifecycles, Dependency Scopes, SLF4J / Logback Logging & JUnit 5 + Mockito TDD

8.3 Maven & Gradle Build Lifecycles, Dependency Scopes, SLF4J / Logback Logging & JUnit 5 + Mockito TDD [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Maven • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Maven & Gradle Build Lifecycles, Dependency Scopes, SLF4J / Logback Logging & JUnit 5 + Mockito TDD in real Java Full Stack Development & Enterpri...

What Topic Is Next → Next Topic 8.4: Gang of Four (GoF) Design Patterns in Java: Singleton, Factory, Builder, Strategy, Observer, Adapter & Proxy

8.4 Gang of Four (GoF) Design Patterns in Java: Singleton, Factory, Builder, Strategy, Observer, Adapter & Proxy [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Gang of Four (GoF) D... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Gang of Four (GoF) Design Patterns in Java • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Gang of Four (GoF) Design Patterns in Java: Singleton, Factory, Builder, Strategy, Observer, Adapter & Proxy in real Java Full Stack Development &...

What Topic Is Next → Module 8 Capstone (fincore-m08-jdbc-maven-junit-patterns) → Next Topic 9.1: How the Web Works (DNS, TCP/TLS, HTTP Methods & Status Codes), Semantic HTML5 Forms & Accessibility

Module 8 Hands-On Capstone Project (100 Marks): fincore-m08-jdbc-maven-junit-patterns — Architect, implement, test, and push the production GitHub repository 'fincore-m08-jdbc-maven-junit-patterns' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 8:...

MODULE 9 OF 22 • STAGE 2 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Web Frontend Foundations From Scratch: HTML5, CSS3, Tailwind CSS & Core JavaScript (ES6+)

Beginner-to-expert real-repository module covering Web Frontend Foundations From Scratch: HTML5, CSS3, Tailwind CSS & Core JavaScript (ES6+) with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ How the Web Works (DNS, TCP/TLS, HTTP Methods & Status Codes), Semantic HTML5 Forms & Accessibility — Purpose: Build semantic, SEO-friendly, keyboard-accessible web documents and understand browser-to-server HTTP lifecycles. • ✓ CSS3 Box Model, Positioning, Flexbox, CSS Grid, Responsive Breakpoints & Tailwind CSS Design Systems — Purpose: Craft pixel-clean mobile, tablet, and desktop layouts rapidly using modern CSS layout primitives and Tailwind utilities. • ✓ Core JavaScript From Zero: let/const, Functions, Closures, Prototypes, Arrays, Objects & ES6+ Modules — Purpose: Master JavaScript execution contexts, lexical scope, array transformations (map/filter/reduce), and clean modular code. • ✓ Browser DOM Manipulation, Event Delegation, Event

Loop, Promises, async/await & Fetch REST Integration — Purpose: Connect interactive frontend UIs dynamically to backend JSON APIs with loading states, retries, and error banners.

9.1 How the Web Works (DNS, TCP/TLS, HTTP Methods & Status Codes), Semantic HTML5 Forms & Accessibility [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for How the Web Works (D... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for How the Web Works (DNS, TCP/TLS, HTTP Methods & Stat... • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply How the Web Works (DNS, TCP/TLS, HTTP Methods & Status Codes), Semantic HTML5 Forms & Accessibility in real Java Full Stack Development & Enterpri...

What Topic Is Next → Next Topic 9.2: CSS3 Box Model, Positioning, Flexbox, CSS Grid, Responsive Breakpoints & Tailwind CSS Design Systems

9.2 CSS3 Box Model, Positioning, Flexbox, CSS Grid, Responsive Breakpoints & Tailwind CSS Design Systems [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for CSS3 Box Model • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply CSS3 Box Model, Positioning, Flexbox, CSS Grid, Responsive Breakpoints & Tailwind CSS Design Systems in real Java Full Stack Development & Enterpr...

What Topic Is Next → Next Topic 9.3: Core JavaScript From Zero: let/const, Functions, Closures, Prototypes, Arrays, Objects & ES6+ Modules

9.3 Core JavaScript From Zero: let/const, Functions, Closures, Prototypes, Arrays, Objects & ES6+ Modules [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Core JavaScript From... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Core JavaScript From Zero • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Core JavaScript From Zero: let/const, Functions, Closures, Prototypes, Arrays, Objects & ES6+ Modules in real Java Full Stack Development & Enterp...

What Topic Is Next → Next Topic 9.4: Browser DOM Manipulation, Event Delegation, Event Loop, Promises, async/await & Fetch REST Integration

9.4 Browser DOM Manipulation, Event Delegation, Event Loop, Promises, async/await & Fetch REST Integration [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Browser DOM Manipula... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Browser DOM Manipulation • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Browser DOM Manipulation, Event Delegation, Event Loop, Promises, async/await & Fetch REST Integration in real Java Full Stack Development & Enter...

What Topic Is Next → Module 9 Capstone (fincore-m09-web-foundations-js-dom) → Next Topic 10.1: TypeScript 5 From Basics to Generics, Discriminated Unions, Utility Types & Zod Schema Validation

Module 9 Hands-On Capstone Project (100 Marks): fincore-m09-web-foundations-js-dom — Architect, implement, test, and push the production GitHub repository 'fincore-m09-web-foundations-js-dom' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 9: Web...

MODULE 10 OF 22 • STAGE 2 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Modern Enterprise Frontend Engineering with Strict TypeScript 5 & React 19

Beginner-to-expert real-repository module covering Modern Enterprise Frontend Engineering with Strict TypeScript 5 & React 19 with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ TypeScript 5 From Basics to Generics, Discriminated Unions, Utility Types & Zod Schema Validation — Purpose: Catch API contract mismatches at compile time and validate untrusted JSON responses before rendering UI components. • ✓ React 19 Core: JSX, Component Composition, Props, State, useEffect, useRef & Reusable Custom Hooks — Purpose: Build modular, predictable single-page application views with clean state lifecycles and zero memory-leaking effects. • ✓ React Router Protected RBAC Routes, Zustand / Context State & TanStack Query Server-State Caching — Purpose: Handle multi-role navigation, pagination, background refetching, and cache invalidation across enterprise portals. • ✓ Accessible Enterprise Forms, Data Tables, SSE Real-Time Token Streaming & Vitest Component Testing — Purpose: Deliver responsive banking and AI operator consoles verified by automated frontend component and interaction suites.

10.1 TypeScript 5 From Basics to Generics, Discriminated Unions, Utility Types & Zod Schema Validation [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for TypeScript 5 From Ba... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for TypeScript 5 From Basics to Generics • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply TypeScript 5 From Basics to Generics, Discriminated Unions, Utility Types & Zod Schema Validation in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 10.2: React 19 Core: JSX, Component Composition, Props, State, useEffect, useRef & Reusable Custom Hooks

10.2 React 19 Core: JSX, Component Composition, Props, State, useEffect, useRef & Reusable Custom Hooks [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for React 19 Core • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply React 19 Core: JSX, Component Composition, Props, State, useEffect, useRef & Reusable Custom Hooks in real Java Full Stack Development & Enterpris...

What Topic Is Next → Next Topic 10.3: React Router Protected RBAC Routes, Zustand / Context State & TanStack Query Server-State Caching

10.3 React Router Protected RBAC Routes, Zustand / Context State & TanStack Query Server-State Caching [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for React Router Protect... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for React Router Protected RBAC Routes • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply React Router Protected RBAC Routes, Zustand / Context State & TanStack Query Server-State Caching in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 10.4: Accessible Enterprise Forms, Data Tables, SSE Real-Time Token Streaming & Vitest Component Testing

10.4 Accessible Enterprise Forms, Data Tables, SSE Real-Time Token Streaming & Vitest Component Testing [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Accessible Enterpris... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Accessible Enterprise Forms • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Accessible Enterprise Forms, Data Tables, SSE Real-Time Token Streaming & Vitest Component Testing in real Java Full Stack Development & Enterpris...

What Topic Is Next → Module 10 Capstone (fincore-m10-react19-typescript-ui) → Next Topic 11.1: Spring Core From First Principles: IoC Container, Constructor Dependency Injection, Bean Scopes & Profiles

Module 10 Hands-On Capstone Project (100 Marks): fincore-m10-react19-typescript-ui — Architect, implement, test, and push the production GitHub repository 'fincore-m10-react19-typescript-ui' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 10: Mod...

MODULE 11 OF 22 • STAGE 2 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Spring Core (IoC / AOP) & Spring Boot 3.3 Enterprise REST API Engineering

Beginner-to-expert real-repository module covering Spring Core (IoC / AOP) & Spring Boot 3.3 Enterprise REST API Engineering with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Spring Core From First Principles: IoC Container, Constructor Dependency Injection, Bean Scopes & Profiles — Purpose: Eliminate tight coupling and wire clean, testable service graphs managed by the Spring ApplicationContext. • ✓ Spring Boot 3.3 Auto-Configuration, @ConfigurationProperties Validation, Spring AOP & Embedded Tomcat — Purpose: Bootstrap production-grade Java services with strongly typed environment configuration and cross-cutting audit aspects. • ✓ RESTful API Design: Controllers, Immutable Record DTOs, Jakarta Bean Validation & RFC 9457 ProblemDetail — Purpose: Expose clean, versioned JSON APIs with strict input validation and standardized machine-readable error contracts. • ✓ Pagination, Dynamic Filtering, ETag Caching, OpenAPI 3 / Swagger UI Docs & Actuator Health Probes — Purpose: Publish interactive API contracts for frontend teams and expose Kubernetes-ready liveness and readiness endpoints.

11.1 Spring Core From First Principles: IoC Container, Constructor Dependency Injection, Bean Scopes & Profiles [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Spring Core From Fir... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Spring Core From First Principles • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Spring Core From First Principles: IoC Container, Constructor Dependency Injection, Bean Scopes & Profiles in real Java Full Stack Development & E...

What Topic Is Next → Next Topic 11.2: Spring Boot 3.3 Auto-Configuration, @ConfigurationProperties Validation, Spring AOP & Embedded Tomcat

11.2 Spring Boot 3.3 Auto-Configuration, @ConfigurationProperties Validation, Spring AOP & Embedded Tomcat [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Spring Boot 3.3 Auto • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Spring Boot 3.3 Auto-Configuration • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Spring Boot 3.3 Auto-Configuration, @ConfigurationProperties Validation, Spring AOP & Embedded Tomcat in real Java Full Stack Development & Enterp...

What Topic Is Next → Next Topic 11.3: RESTful API Design: Controllers, Immutable Record DTOs, Jakarta Bean Validation & RFC 9457 ProblemDetail

11.3 RESTful API Design: Controllers, Immutable Record DTOs, Jakarta Bean Validation & RFC 9457 ProblemDetail [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for RESTful API Design • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply RESTful API Design: Controllers, Immutable Record DTOs, Jakarta Bean Validation & RFC 9457 ProblemDetail in real Java Full Stack Development & Ent...

What Topic Is Next → Next Topic 11.4: Pagination, Dynamic Filtering, ETag Caching, OpenAPI 3 / Swagger UI Docs & Actuator Health Probes

11.4 Pagination, Dynamic Filtering, ETag Caching, OpenAPI 3 / Swagger UI Docs & Actuator Health Probes [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Pagination • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Pagination, Dynamic Filtering, ETag Caching, OpenAPI 3 / Swagger UI Docs & Actuator Health Probes in real Java Full Stack Development & Enterprise...

What Topic Is Next → Module 11 Capstone (fincore-m11-springboot-rest-platform) → Next Topic 12.1: ORM Fundamentals, JPA Entities, @OneToMany / @ManyToOne / @ManyToMany Mappings & FetchType.LAZY

Module 11 Hands-On Capstone Project (100 Marks): fincore-m11-springboot-rest-platform — Architect, implement, test, and push the production GitHub repository 'fincore-m11-springboot-rest-platform' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 11:...

MODULE 12 OF 22 • STAGE 2 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Spring Data JPA, Hibernate 6 Internals, Locking & Testcontainers CI

Beginner-to-expert real-repository module covering Spring Data JPA, Hibernate 6 Internals, Locking & Testcontainers CI with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ ORM Fundamentals, JPA Entities, @OneToMany / @ManyToOne / @ManyToMany Mappings & FetchType.LAZY — Purpose: Map complex relational domain models cleanly without accidental eager cartesian-product table explosions. • ✓ Hibernate 6 Persistence Context, First-Level Cache, Dirty Checking & @Transactional Propagation Rules — Purpose: Control exact SQL flush timing, isolation boundaries, and automatic rollback behavior on business exceptions. • ✓ Solving N+1 Select Storms: JOIN FETCH, @EntityGraph, Record DTO Projections & Keyset Pagination — Purpose: Cut 200+ hidden lazy-loading SQL queries down to a single indexed query on high-traffic API endpoints. • ✓ Optimistic (@Version) vs Pessimistic Locking, JDBC Batching & Real PostgreSQL Testcontainers Suites — Purpose: Prevent concurrent double-spend bugs and verify repositories against real ephemeral PostgreSQL containers in CI.

12.1 ORM Fundamentals, JPA Entities, @OneToMany / @ManyToOne / @ManyToMany Mappings & FetchType.LAZY [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for ORM Fundamentals • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply ORM Fundamentals, JPA Entities, @OneToMany / @ManyToOne / @ManyToMany Mappings & FetchType.LAZY in real Java Full Stack Development & Enterprise C...

What Topic Is Next → Next Topic 12.2: Hibernate 6 Persistence Context, First-Level Cache, Dirty Checking & @Transactional Propagation Rules

12.2 Hibernate 6 Persistence Context, First-Level Cache, Dirty Checking & @Transactional Propagation Rules [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Hibernate 6 Persiste... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Hibernate 6 Persistence Context • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Hibernate 6 Persistence Context, First-Level Cache, Dirty Checking & @Transactional Propagation Rules in real Java Full Stack Development & Enterp...

What Topic Is Next → Next Topic 12.3: Solving N+1 Select Storms: JOIN FETCH, @EntityGraph, Record DTO Projections & Keyset Pagination

12.3 Solving N+1 Select Storms: JOIN FETCH, @EntityGraph, Record DTO Projections & Keyset Pagination [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Solving N+1 Select S... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Solving N+1 Select Storms • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Solving N+1 Select Storms: JOIN FETCH, @EntityGraph, Record DTO Projections & Keyset Pagination in real Java Full Stack Development & Enterprise C...

What Topic Is Next → Next Topic 12.4: Optimistic (@Version) vs Pessimistic Locking, JDBC Batching & Real PostgreSQL Testcontainers Suites

12.4 Optimistic (@Version) vs Pessimistic Locking, JDBC Batching & Real PostgreSQL Testcontainers Suites [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Optimistic (@Version... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Optimistic (@Version) vs Pessimistic Locking • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Optimistic (@Version) vs Pessimistic Locking, JDBC Batching & Real PostgreSQL Testcontainers Suites in real Java Full Stack Development & Enterpri...

What Topic Is Next → Module 12 Capstone (fincore-m12-jpa-hibernate-persistence) → Next Topic 13.1: Authentication vs Authorization, Spring Security 6 SecurityFilterChain & Argon2id / BCrypt Hashing

Module 12 Hands-On Capstone Project (100 Marks): fincore-m12-jpa-hibernate-persistence — Architect, implement, test, and push the production GitHub repository 'fincore-m12-jpa-hibernate-persistence' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 12:...

MODULE 13 OF 22 • STAGE 3 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Spring Security 6, Stateless JWT, OAuth2 / OIDC SSO & OWASP API Defense

Beginner-to-expert real-repository module covering Spring Security 6, Stateless JWT, OAuth2 / OIDC SSO & OWASP API Defense with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Authentication vs Authorization, Spring Security 6 SecurityFilterChain & Argon2id / BCrypt Hashing — Purpose: Secure modern Spring Boot APIs using lambda filter chains and GPU-resistant password storage from scratch. • ✓ Stateless JWT Architecture, Asymmetric RS256 / JWKS Verification & Refresh Token Reuse Detection — Purpose: Authenticate distributed API requests without database session lookups and revoke compromised token families instantly. • ✓ OAuth2 / OpenID Connect (Google / Keycloak SSO), Multi-Tenant Claims & @PreAuthorize Method RBAC — Purpose: Enforce fine-grained role and tenant ownership permissions directly on service methods and REST endpoints. • ✓ CORS, CSRF, Distributed Rate Limiting, Audit Trails & OWASP API Security Top-10 Hardening — Purpose: Block broken object-level authorization (BOLA), mass assignment, credential stuffing, and secret leakage in prod.

13.1 Authentication vs Authorization, Spring Security 6 SecurityFilterChain & Argon2id / BCrypt Hashing [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Authentication vs Au... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Authentication vs Authorization • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Authentication vs Authorization, Spring Security 6 SecurityFilterChain & Argon2id / BCrypt Hashing in real Java Full Stack Development & Enterpris...

What Topic Is Next → Next Topic 13.2: Stateless JWT Architecture, Asymmetric RS256 / JWKS Verification & Refresh Token Reuse Detection

13.2 Stateless JWT Architecture, Asymmetric RS256 / JWKS Verification & Refresh Token Reuse Detection [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Stateless JWT Archit... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Stateless JWT Architecture • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Stateless JWT Architecture, Asymmetric RS256 / JWKS Verification & Refresh Token Reuse Detection in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 13.3: OAuth2 / OpenID Connect (Google / Keycloak SSO), Multi-Tenant Claims & @PreAuthorize Method RBAC

13.3 OAuth2 / OpenID Connect (Google / Keycloak SSO), Multi-Tenant Claims & @PreAuthorize Method RBAC [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for OAuth2 / OpenID Conn... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for OAuth2 / OpenID Connect (Google / Keycloak SSO) • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply OAuth2 / OpenID Connect (Google / Keycloak SSO), Multi-Tenant Claims & @PreAuthorize Method RBAC in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 13.4: CORS, CSRF, Distributed Rate Limiting, Audit Trails & OWASP API Security Top-10 Hardening

13.4 CORS, CSRF, Distributed Rate Limiting, Audit Trails & OWASP API Security Top-10 Hardening [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for CORS • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply CORS, CSRF, Distributed Rate Limiting, Audit Trails & OWASP API Security Top-10 Hardening in real Java Full Stack Development & Enterprise Cloud A...

What Topic Is Next → Module 13 Capstone (fincore-m13-spring-security-oauth2) → Next Topic 14.1: Monolith to Microservices Decomposition, DDD Bounded Contexts & Database-per-Service Governance

Module 13 Hands-On Capstone Project (100 Marks): fincore-m13-spring-security-oauth2 — Architect, implement, test, and push the production GitHub repository 'fincore-m13-spring-security-oauth2' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 13: Sp...

MODULE 14 OF 22 • STAGE 3 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Microservices Architecture, Spring Cloud Gateway & Resilience4j Fault Tolerance

Beginner-to-expert real-repository module covering Microservices Architecture, Spring Cloud Gateway & Resilience4j Fault Tolerance with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Monolith to Microservices Decomposition, DDD Bounded Contexts & Database-per-Service Governance — Purpose: Split growing systems along business capability boundaries without creating a fragile distributed monolith. • ✓ Spring Cloud API Gateway: Non-Blocking Netty Routing, Edge Auth Offload & Per-Tenant Throttling — Purpose: Centralize TLS termination, JWT verification, correlation IDs, and rate limits at the microservice edge. • ✓ Synchronous Service-to-Service Communication: Spring 6 RestClient, @HttpExchange, gRPC & Timeout Budgets — Purpose: Call downstream internal services with typed contracts, strict deadlines, and zero retry-amplification storms. • ✓ Resilience4j Fault Tolerance: Circuit Breakers, Bulkhead Isolation, Jittered Retries & Graceful Fallbacks — Purpose: Keep checkout and banking flows alive even when a third-party KYC or notification provider experiences downtime.

14.1 Monolith to Microservices Decomposition, DDD Bounded Contexts & Database-per-Service Governance [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Monolith to Microser... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Monolith to Microservices Decomposition • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Monolith to Microservices Decomposition, DDD Bounded Contexts & Database-per-Service Governance in real Java Full Stack Development & Enterprise C...

What Topic Is Next → Next Topic 14.2: Spring Cloud API Gateway: Non-Blocking Netty Routing, Edge Auth Offload & Per-Tenant Throttling

14.2 Spring Cloud API Gateway: Non-Blocking Netty Routing, Edge Auth Offload & Per-Tenant Throttling [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Spring Cloud API Gat... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Spring Cloud API Gateway • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Spring Cloud API Gateway: Non-Blocking Netty Routing, Edge Auth Offload & Per-Tenant Throttling in real Java Full Stack Development & Enterprise C...

What Topic Is Next → Next Topic 14.3: Synchronous Service-to-Service Communication: Spring 6 RestClient, @HttpExchange, gRPC & Timeout Budgets

14.3 Synchronous Service-to-Service Communication: Spring 6 RestClient, @HttpExchange, gRPC & Timeout Budgets [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Synchronous Service • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Synchronous Service-to-Service Communication • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Synchronous Service-to-Service Communication: Spring 6 RestClient, @HttpExchange, gRPC & Timeout Budgets in real Java Full Stack Development & Ent...

What Topic Is Next → Next Topic 14.4: Resilience4j Fault Tolerance: Circuit Breakers, Bulkhead Isolation, Jittered Retries & Graceful Fallbacks

14.4 Resilience4j Fault Tolerance: Circuit Breakers, Bulkhead Isolation, Jittered Retries & Graceful Fallbacks [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Resilience4j Fault T... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Resilience4j Fault Tolerance • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Resilience4j Fault Tolerance: Circuit Breakers, Bulkhead Isolation, Jittered Retries & Graceful Fallbacks in real Java Full Stack Development & En...

What Topic Is Next → Module 14 Capstone (fincore-m14-microservices-gateway-resilience) → Next Topic 15.1: Redis 7 Data Structures, Multi-Tier Caching (Caffeine + Redis), TTL Jitter & Redisson Distributed Locks

Module 14 Hands-On Capstone Project (100 Marks): fincore-m14-microservices-gateway-resilience — Architect, implement, test, and push the production GitHub repository 'fincore-m14-microservices-gateway-resilience' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Mod...

MODULE 15 OF 22 • STAGE 3 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Distributed Caching (Redis 7), Event-Driven Apache Kafka & Saga Transactions

Beginner-to-expert real-repository module covering Distributed Caching (Redis 7), Event-Driven Apache Kafka & Saga Transactions with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Redis 7 Data Structures, Multi-Tier Caching (Caffeine + Redis), TTL Jitter & Redisson Distributed Locks — Purpose: Slash read latency below 2ms, prevent cache stampedes, and coordinate cluster-wide idempotency keys safely. • ✓ Apache Kafka From Scratch: KRaft Brokers, Topics, Partition Keys, Consumer Groups & Producer Durability — Purpose: Stream millions of ordered business events per minute with acks=all durability and partition-ordered scaling.

• ✓ Spring for Apache Kafka, Schema Compatibility, Idempotent Consumers, Retries & Dead-Letter Topics (DLT) — Purpose: Process at-least-once event streams without duplicate side effects while isolating poison-pill payloads cleanly. • ✓ Transactional Outbox Pattern & Distributed Sagas (Orchestration + Compensating Rollback Actions) — Purpose: Guarantee eventual consistency across Order, Wallet, and Ledger microservices without brittle two-phase locks.

15.1 Redis 7 Data Structures, Multi-Tier Caching (Caffeine + Redis), TTL Jitter & Redisson Distributed Locks [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Redis 7 Data Structu... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Redis 7 Data Structures • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Redis 7 Data Structures, Multi-Tier Caching (Caffeine + Redis), TTL Jitter & Redisson Distributed Locks in real Java Full Stack Development & Ente...

What Topic Is Next → Next Topic 15.2: Apache Kafka From Scratch: KRaft Brokers, Topics, Partition Keys, Consumer Groups & Producer Durability

15.2 Apache Kafka From Scratch: KRaft Brokers, Topics, Partition Keys, Consumer Groups & Producer Durability [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Apache Kafka From Sc... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Apache Kafka From Scratch • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Apache Kafka From Scratch: KRaft Brokers, Topics, Partition Keys, Consumer Groups & Producer Durability in real Java Full Stack Development & Ente...

What Topic Is Next → Next Topic 15.3: Spring for Apache Kafka, Schema Compatibility, Idempotent Consumers, Retries & Dead-Letter Topics (DLT)

15.3 Spring for Apache Kafka, Schema Compatibility, Idempotent Consumers, Retries & Dead-Letter Topics (DLT) [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Spring for Apache Ka... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Spring for Apache Kafka • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Spring for Apache Kafka, Schema Compatibility, Idempotent Consumers, Retries & Dead-Letter Topics (DLT) in real Java Full Stack Development & Ente...

What Topic Is Next → Next Topic 15.4: Transactional Outbox Pattern & Distributed Sagas (Orchestration + Compensating Rollback Actions)

15.4 Transactional Outbox Pattern & Distributed Sagas (Orchestration + Compensating Rollback Actions) [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Transactional Outbox... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Transactional Outbox Pattern • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Transactional Outbox Pattern & Distributed Sagas (Orchestration + Compensating Rollback Actions) in real Java Full Stack Development & Enterprise...

What Topic Is Next → Module 15 Capstone (fincore-m15-redis-kafka-saga-engine) → Next Topic 16.1: Generative AI Fundamentals for Java Engineers, Spring AI ChatClient & Typed Java Record Structured Outputs

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

MODULE 16 OF 22 • STAGE 3 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Spring AI, Enterprise RAG Pipelines, pgvector Search & Agentic Tool Calling

Beginner-to-expert real-repository module covering Spring AI, Enterprise RAG Pipelines, pgvector Search & Agentic Tool Calling with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Generative AI Fundamentals for Java Engineers, Spring AI ChatClient & Typed Java Record Structured Outputs — Purpose: Integrate enterprise LLMs into Spring Boot 3 services with strongly typed JSON-to-Record response binding. • ✓ Embeddings, Semantic Chunking, PostgreSQL pgvector HNSW Indexing & Hybrid Keyword + Vector Retrieval — Purpose: Index internal policies, docs, and ticket histories so enterprise AI copilots retrieve grounded facts in milliseconds. • ✓ Spring AI Advisors, Citation-Backed RAG Pipelines, Conversational Memory & SSE Streaming Endpoints — Purpose: Stream real-time cited answers to React 19 UIs with zero unverified hallucinations and per-tenant data isolation. • ✓ Guardrailed Spring AI @Tool Function Calling, Prompt-Injection Defense, Cost Quotas & AI Quality Evals — Purpose: Let Java copilots invoke verified backend tools safely under strict RBAC, input sanitization, and token budgets.

16.1 Generative AI Fundamentals for Java Engineers, Spring AI ChatClient & Typed Java Record Structured Outputs [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Generative AI Fundam... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Generative AI Fundamentals for Java Engineers • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Generative AI Fundamentals for Java Engineers, Spring AI ChatClient & Typed Java Record Structured Outputs in real Java Full Stack Development & E...

What Topic Is Next → Next Topic 16.2: Embeddings, Semantic Chunking, PostgreSQL pgvector HNSW Indexing & Hybrid Keyword + Vector Retrieval

16.2 Embeddings, Semantic Chunking, PostgreSQL pgvector HNSW Indexing & Hybrid Keyword + Vector Retrieval

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

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Embeddings • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Embeddings, Semantic Chunking, PostgreSQL pgvector HNSW Indexing & Hybrid Keyword + Vector Retrieval in real Java Full Stack Development & Enterpr...

What Topic Is Next → Next Topic 16.3: Spring AI Advisors, Citation-Backed RAG Pipelines, Conversational Memory & SSE Streaming Endpoints

16.3 Spring AI Advisors, Citation-Backed RAG Pipelines, Conversational Memory & SSE Streaming Endpoints

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

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Spring AI Advisors • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Spring AI Advisors, Citation-Backed RAG Pipelines, Conversational Memory & SSE Streaming Endpoints in real Java Full Stack Development & Enterpris...

What Topic Is Next → Next Topic 16.4: Guardrailed Spring AI @Tool Function Calling, Prompt-Injection Defense, Cost Quotas & AI Quality Evals

16.4 Guardrailed Spring AI @Tool Function Calling, Prompt-Injection Defense, Cost Quotas & AI Quality Evals

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

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Guardrailed Spring A... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Guardrailed Spring AI @Tool Function Calling • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Guardrailed Spring AI @Tool Function Calling, Prompt-Injection Defense, Cost Quotas & AI Quality Evals in real Java Full Stack Development & Enter...

What Topic Is Next → Module 16 Capstone (fincore-m16-spring-ai-rag-copilot) → Next Topic 17.1: Containers vs Virtual Machines From Scratch, Layered Multi-Stage Java 21 Dockerfiles & Non-Root Security

Module 16 Hands-On Capstone Project (100 Marks): fincore-m16-spring-ai-rag-copilot — Architect, implement, test, and push the production GitHub repository 'fincore-m16-spring-ai-rag-copilot' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 16: Spr...

MODULE 17 OF 22 • STAGE 3 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

Docker Containerization & Kubernetes (EKS) Production Orchestration

Beginner-to-expert real-repository module covering Docker Containerization & Kubernetes (EKS) Production Orchestration with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Containers vs Virtual Machines From Scratch, Layered Multi-Stage Java 21 Dockerfiles & Non-Root Security — Purpose: Package Spring Boot and React builds into lean, reproducible, CVE-scanned container images that start in seconds. • ✓ Docker Bridge Networking, Persistent Volumes & Multi-Service Local Stacks with Docker Compose — Purpose: Spin up Spring Boot, React, PostgreSQL, Redis, and Kafka locally with a single command for rapid integration testing. • ✓ Kubernetes Architecture From Scratch: Pods, ReplicaSets, Rolling Deployments, Services & Namespaces — Purpose: Schedule self-healing microservice replicas across cluster nodes with zero-downtime rolling updates and rollbacks. • ✓ ConfigMaps, External Secrets, Startup/Readiness Probes, Ingress TLS, HPA Autoscaling & Helm Charts — Purpose: Scale Java pods automatically under traffic spikes and template multi-environment releases cleanly with Helm.

17.1 Containers vs Virtual Machines From Scratch, Layered Multi-Stage Java 21 Dockerfiles & Non-Root Security

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

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Containers vs Virtua... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Containers vs Virtual Machines From Scratch • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Containers vs Virtual Machines From Scratch, Layered Multi-Stage Java 21 Dockerfiles & Non-Root Security in real Java Full Stack Development & Ent...

What Topic Is Next → Next Topic 17.2: Docker Bridge Networking, Persistent Volumes & Multi-Service Local Stacks with Docker Compose

17.2 Docker Bridge Networking, Persistent Volumes & Multi-Service Local Stacks with Docker Compose

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

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Docker Bridge Networ... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Docker Bridge Networking • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Docker Bridge Networking, Persistent Volumes & Multi-Service Local Stacks with Docker Compose in real Java Full Stack Development & Enterprise Clo...

What Topic Is Next → Next Topic 17.3: Kubernetes Architecture From Scratch: Pods, ReplicaSets, Rolling Deployments, Services & Namespaces

17.3 Kubernetes Architecture From Scratch: Pods, ReplicaSets, Rolling Deployments, Services & Namespaces [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Kubernetes Architect... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Kubernetes Architecture From Scratch • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Kubernetes Architecture From Scratch: Pods, ReplicaSets, Rolling Deployments, Services & Namespaces in real Java Full Stack Development & Enterpri...

What Topic Is Next → Next Topic 17.4: ConfigMaps, External Secrets, Startup/Readiness Probes, Ingress TLS, HPA Autoscaling & Helm Charts

17.4 ConfigMaps, External Secrets, Startup/Readiness Probes, Ingress TLS, HPA Autoscaling & Helm Charts [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for ConfigMaps • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply ConfigMaps, External Secrets, Startup/Readiness Probes, Ingress TLS, HPA Autoscaling & Helm Charts in real Java Full Stack Development & Enterpris...

What Topic Is Next → Module 17 Capstone (fincore-m17-docker-kubernetes-eks) → Next Topic 18.1: Cloud Computing Concepts, AWS Global Regions / AZs, Billing Alerts & Zero-Trust IAM Roles / OIDC

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

MODULE 18 OF 22 • STAGE 4 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

AWS Cloud Engineering From Foundations & Terraform Infrastructure as Code

Beginner-to-expert real-repository module covering AWS Cloud Engineering From Foundations & Terraform Infrastructure as Code with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ Cloud Computing Concepts, AWS Global Regions / AZs, Billing Alerts & Zero-Trust IAM Roles / OIDC — Purpose: Configure secure AWS accounts from scratch with MFA, least-privilege IAM policies, and zero hardcoded access keys. • ✓ AWS Networking & Compute: Multi-AZ VPC, Public/Private Subnets, NAT, EC2, Security Groups & ALB — Purpose: Design isolated 3-tier enterprise cloud networks where databases and internal services have zero public IPs. • ✓ Managed AWS Data & Edge: Amazon S3, CloudFront CDN, Route 53 TLS DNS, RDS Multi-AZ & Secrets Manager — Purpose: Host global static assets at edge locations and run automated-failover PostgreSQL databases with KMS encryption. • ✓ Terraform Infrastructure as Code (IaC): HCL Syntax, Remote S3 State Locking, Modules & AWS EKS Provisioning — Purpose: Provision and version-control complete AWS VPC, RDS, ECR, and EKS infrastructure reproducibly from Git.

18.1 Cloud Computing Concepts, AWS Global Regions / AZs, Billing Alerts & Zero-Trust IAM Roles / OIDC [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Cloud Computing Conc... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Cloud Computing Concepts • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Cloud Computing Concepts, AWS Global Regions / AZs, Billing Alerts & Zero-Trust IAM Roles / OIDC in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 18.2: AWS Networking & Compute: Multi-AZ VPC, Public/Private Subnets, NAT, EC2, Security Groups & ALB

18.2 AWS Networking & Compute: Multi-AZ VPC, Public/Private Subnets, NAT, EC2, Security Groups & ALB [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for AWS Networking • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply AWS Networking & Compute: Multi-AZ VPC, Public/Private Subnets, NAT, EC2, Security Groups & ALB in real Java Full Stack Development & Enterprise C...

What Topic Is Next → Next Topic 18.3: Managed AWS Data & Edge: Amazon S3, CloudFront CDN, Route 53 TLS DNS, RDS Multi-AZ & Secrets Manager

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

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Managed AWS Data • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply Managed AWS Data & Edge: Amazon S3, CloudFront CDN, Route 53 TLS DNS, RDS Multi-AZ & Secrets Manager in real Java Full Stack Development & Enterpr...

What Topic Is Next → Next Topic 18.4: Terraform Infrastructure as Code (IaC): HCL Syntax, Remote S3 State Locking, Modules & AWS EKS Provisioning

18.4 Terraform Infrastructure as Code (IaC): HCL Syntax, Remote S3 State Locking, Modules & AWS EKS Provisioning [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Terraform Infrastruc... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Terraform Infrastructure as Code (IaC) • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Terraform Infrastructure as Code (IaC): HCL Syntax, Remote S3 State Locking, Modules & AWS EKS Provisioning in real Java Full Stack Development &...

What Topic Is Next → Module 18 Capstone (fincore-m18-aws-cloud-terraform-iac) → Next Topic 19.1: GitHub Actions DevSecOps CI/CD: Automated Maven Tests, SonarQube SAST, Trivy Image Scans & EKS Rollouts

Module 18 Hands-On Capstone Project (100 Marks): fincore-m18-aws-cloud-terraform-iac — Architect, implement, test, and push the production GitHub repository 'fincore-m18-aws-cloud-terraform-iac' (strictly zero dummy scaffolding) demonstrating end-to-end mastery of Module 18: A...

MODULE 19 OF 22 • STAGE 4 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

DevSecOps CI/CD Pipelines, OpenTelemetry Observability & System Design (LLD + HLD)

Beginner-to-expert real-repository module covering DevSecOps CI/CD Pipelines, OpenTelemetry Observability & System Design (LLD + HLD) with supervised live coding, ≥90% MockAttempt AI Viva Gates, and production GitHub PR verification.

Objectives: ✓ GitHub Actions DevSecOps CI/CD: Automated Maven Tests, SonarQube SAST, Trivy Image Scans & EKS Rollouts — Purpose: Block buggy or vulnerable commits automatically and deploy signed container images on every verified merge. • ✓ Full-Stack Observability: OpenTelemetry Distributed Tracing, Micrometer Prometheus, Grafana & JSON Logs — Purpose: Trace a single slow request across React -> Gateway -> Spring -> Kafka -> Postgres in one unified flamegraph. • ✓ Low-Level Design (LLD) & Machine Coding Rounds: Parking Lot, Splitwise, Rate Limiter & Notification Engine — Purpose: Ace 90-minute object-oriented machine coding rounds with clean interfaces, concurrency safety, and unit tests. • ✓ High-Level System Design (HLD): Caching, Database Sharding, CAP Theorem, UPI Ledger & Scalable Feeds — Purpose: Design 100k-RPS distributed architectures on interview whiteboards with clear latency, cost, and failure trade-offs.

19.1 GitHub Actions DevSecOps CI/CD: Automated Maven Tests, SonarQube SAST, Trivy Image Scans & EKS Rollouts [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for GitHub Actions DevSe... • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for GitHub Actions DevSecOps CI/CD • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply GitHub Actions DevSecOps CI/CD: Automated Maven Tests, SonarQube SAST, Trivy Image Scans & EKS Rollouts in real Java Full Stack Development & Ente...

What Topic Is Next → Next Topic 19.2: Full-Stack Observability: OpenTelemetry Distributed Tracing, Micrometer Prometheus, Grafana & JSON Logs

19.2 Full-Stack Observability: OpenTelemetry Distributed Tracing, Micrometer Prometheus, Grafana & JSON Logs [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Full • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Full-Stack Observability • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Full-Stack Observability: OpenTelemetry Distributed Tracing, Micrometer Prometheus, Grafana & JSON Logs in real Java Full Stack Development & Ente...

What Topic Is Next → Next Topic 19.3: Low-Level Design (LLD) & Machine Coding Rounds: Parking Lot, Splitwise, Rate Limiter & Notification Engine

19.3 Low-Level Design (LLD) & Machine Coding Rounds: Parking Lot, Splitwise, Rate Limiter & Notification Engine [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Low • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Low-Level Design (LLD) • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Low-Level Design (LLD) & Machine Coding Rounds: Parking Lot, Splitwise, Rate Limiter & Notification Engine in real Java Full Stack Development & E...

What Topic Is Next → Next Topic 19.4: High-Level System Design (HLD): Caching, Database Sharding, CAP Theorem, UPI Ledger & Scalable Feeds

19.4 High-Level System Design (HLD): Caching, Database Sharding, CAP Theorem, UPI Ledger & Scalable Feeds [90 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for High • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for High-Level System Design (HLD) • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply High-Level System Design (HLD): Caching, Database Sharding, CAP Theorem, UPI Ledger & Scalable Feeds in real Java Full Stack Development & Enterpr...

What Topic Is Next → Module 19 Capstone (fincore-m19-devsecops-otel-system-design) → Next Topic 20.1: End-to-End Architecture & Delivery of FinCore Enterprise Banking, Spring AI Copilot & Kafka Platform

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

MODULE 20 OF 22 • STAGE 4 • 24 LIVE + LAB HOURS • 4 TOPICS (32 SUBTOPICS) • 1 CAPSTONE PROJECT (100 MARKS)

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

Beginner-to-expert real-repository module covering FinCore 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 Architecture & Delivery of FinCore Enterprise Banking, Spring AI Copilot & Kafka Platform — Purpose: Integrate Java 21 microservices, Spring AI RAG, Kafka sagas, and React 19 consoles in a mentor-led client sprint. • ✓ OWASP Security Audit, k6 Load Testing, Chaos Failover Drill & Live AWS HTTPS Production Cutover — Purpose: Prove p95 latency SLAs, zero data loss under pod restarts, and TLS readiness before earning your Internship Letter. • ✓ ATS-Winning Java Full Stack Resume, 20-Repo GitHub Portfolio Curation & System Design Demo Pitch — Purpose: Present verifiable production repositories, commit histories, and architecture diagrams that stand out to recruiters. • ✓ Full-Length Coding Tests, Spoken MockAttempt AI Viva Loops & ≥90% Placement Guarantee Activation — Purpose: Clear all ≥90% assessment gates (with unlimited free re-attempts) to unlock 100 curated IT job openings in 6 months.

20.1 End-to-End Architecture & Delivery of FinCore Enterprise Banking, Spring AI Copilot & Kafka Platform

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

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for End • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for End-to-End Architecture • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply End-to-End Architecture & Delivery of FinCore Enterprise Banking, Spring AI Copilot & Kafka Platform in real Java Full Stack Development & Enterpr...

What Topic Is Next → Next Topic 20.2: OWASP Security Audit, k6 Load Testing, Chaos Failover Drill & Live AWS HTTPS Production Cutover

20.2 OWASP Security Audit, k6 Load Testing, Chaos Failover Drill & 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 JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for OWASP Security Audit • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10 • Diagnostics, Profiling & Telemetry: Spring Actuator, Micrometer OpenTelemetry & EXPLAIN ANALYZE

Purpose: To understand and apply OWASP Security Audit, k6 Load Testing, Chaos Failover Drill & Live AWS HTTPS Production Cutover in real Java Full Stack Development & Enterprise C...

What Topic Is Next → Next Topic 20.3: ATS-Winning Java Full Stack Resume, 20-Repo GitHub Portfolio Curation & System Design Demo Pitch

20.3 ATS-Winning Java Full Stack Resume, 20-Repo GitHub Portfolio Curation & System Design Demo Pitch

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

Subtopics Covered: Input Validation, Edge • Modular Clean • Idiomatic JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for ATS • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for ATS-Winning Java Full Stack Resume • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply ATS-Winning Java Full Stack Resume, 20-Repo GitHub Portfolio Curation & System Design Demo Pitch in real Java Full Stack Development & Enterprise...

What Topic Is Next → Next Topic 20.4: Full-Length Coding Tests, Spoken MockAttempt AI Viva Loops & ≥90% Placement Guarantee Activation

20.4 Full-Length Coding Tests, 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 JFSC Implementation • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Full • Runnable Code Implementation (JAVA) • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Full-Length Coding Tests • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Full-Length Coding Tests, Spoken MockAttempt AI Viva Loops & ≥90% Placement Guarantee Activation in real Java Full Stack Development & Enterprise...

What Topic Is Next → Module 20 Capstone (fincore-m20-live-client-production-capstone) → Next Topic 21.1: Spring AI Advisors, Structured Output, Tool Calling & Grounded pgvector Retrieval

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

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

Enterprise Spring AI, pgvector RAG & Event-Driven Saga Resilience

Self-explanatory list-and-bullet production mastery module covering Enterprise Spring AI, pgvector RAG & Event-Driven Saga Resilience in Java Full Stack Development & Enterprise Cloud Architecture, paired with hands-on labs, capstone repository, Topic Mock Tests, and Spoken MockAttempt AI Viva.

Objectives: ✓ Spring AI Advisors, Structured Output, Tool Calling & Grounded pgvector Retrieval — Purpose: Embed audited generative AI workflows directly inside type-safe Java 21 Spring Boot services. • ✓ Transactional Outbox, Idempotent Kafka Consumers & Resilience4j Bulkheads/Circuit Breakers — Purpose: Guarantee financial consistency and fault isolation across distributed Java microservices. • ✓ React 19 + TypeScript Enterprise Frontend Integration with OAuth2/OIDC BFF Pattern — Purpose: Connect React workspaces securely to Spring Security 6 resource servers without token exposure. • ✓ Build, verify, and document 'Spring AI + pgvector Enterprise Copilot & Transactional Outbox Microservice Repo' with clean Git commits and edge-case protection. • ✓ Clear 100% of Topic Timed Mock Tests and Spoken Voice + MockAttempt AI Technical Interviews.

21.1 Spring AI Advisors, Structured Output, Tool Calling & Grounded pgvector Retrieval [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Security Hygiene & Data Safety Rules for Spring AI Advisors, Structured Output, Tool Calling & Grounded pgvector Re... • Performance Optimization, Complexity & Latency Impact of Spring AI Advisors, Structured Output, Tool Calling & Grou... • Live Debugging, Stack • Runnable Code Implementation (JAVA) • Hands • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Spring AI Advisors • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Spring AI Advisors, Structured Output, Tool Calling & Grounded pgvector Retrieval in real Java Full Stack Development & Enterprise Cloud Architect...

What Topic Is Next → Next Topic 21.2: Transactional Outbox, Idempotent Kafka Consumers & Resilience4j Bulkheads/Circuit Breakers

21.2 Transactional Outbox, Idempotent Kafka Consumers & Resilience4j Bulkheads/Circuit Breakers [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Security Hygiene & Data Safety Rules for Transactional Outbox, Idempotent Kafka Consumers & Resilience4j Bulkheads/... • Performance Optimization, Complexity & Latency Impact of Transactional Outbox, Idempotent Kafka Consumers & Resilie... • Live Debugging, Stack • Runnable Code Implementation (JAVA) • Hands • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Transactional Outbox • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Transactional Outbox, Idempotent Kafka Consumers & Resilience4j Bulkheads/Circuit Breakers in real Java Full Stack Development & Enterprise Cloud...

What Topic Is Next → Next Topic 21.3: React 19 + TypeScript Enterprise Frontend Integration with OAuth2/OIDC BFF Pattern

21.3 React 19 + TypeScript Enterprise Frontend Integration with OAuth2/OIDC BFF Pattern [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Security Hygiene & Data Safety Rules for React 19 + TypeScript Enterprise Frontend Integration with OAuth2/OIDC BFF... • Performance Optimization, Complexity & Latency Impact of React 19 + TypeScript Enterprise Frontend Integration with... • Live Debugging, Stack • Runnable Code Implementation (JAVA) • Hands • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for React 19 + TypeScript Enterprise Frontend Integratio... • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply React 19 + TypeScript Enterprise Frontend Integration with OAuth2/OIDC BFF Pattern in real Java Full Stack Development & Enterprise Cloud Architec...

What Topic Is Next → Module 21 Capstone (Spring AI + pgvector Enterprise Copilot & Transactional Outbox Microservice Repo) → Next Topic 22.1: Testcontainers CI, Multi-Stage JRE Images & Zero-Downtime AWS Cloud Rollouts

Module 21 Hands-On Capstone Project (100 Marks): Spring AI + pgvector Enterprise Copilot & Transactional Outbox Microservice Repo — Architect, implement, test, and document 'Spring AI + pgvector Enterprise Copilot & Transactional Outbox Microservice Repo' applying every bullet-wise engineering concept from Module 21 (Ent...

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

AWS Cloud Deployment, JVM Observability & ≥90% Placement Gate System Design

Self-explanatory list-and-bullet production mastery module covering AWS Cloud Deployment, JVM Observability & ≥90% Placement Gate System Design in Java Full Stack Development & Enterprise Cloud Architecture, paired with hands-on labs, capstone repository, Topic Mock Tests, and Spoken MockAttempt AI Viva.

Objectives: ✓ Testcontainers CI, Multi-Stage JRE Images & Zero-Downtime AWS Cloud Rollouts — Purpose: Automate end-to-end verification and production deployment from GitHub Actions to AWS. • ✓ OpenTelemetry Distributed Tracing, Micrometer Prometheus Dashboards & JFR Profiling — Purpose: Diagnose cross-service p99 latency, connection pool exhaustion, and JVM memory pressure. • ✓ Full Stack Java Architect System Design, Live Coding & Placement Interview Defense — Purpose: Clear product-company DSA, Spring internals, React, and cloud architecture hiring rounds. • ✓ Build, verify, and document 'Docker + EKS/AWS CI/CD Deployment, OpenTelemetry Tracing & Mock Hiring Loop Repo' with clean Git commits and edge-case protection. • ✓ Clear 100% of Topic Timed Mock Tests and Spoken Voice + MockAttempt AI Technical Interviews.

22.1 Testcontainers CI, Multi-Stage JRE Images & Zero-Downtime AWS Cloud Rollouts [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Security Hygiene & Data Safety Rules for Testcontainers CI, Multi • Performance Optimization, Complexity & Latency Impact of Testcontainers CI, Multi • Live Debugging, Stack • Runnable Code Implementation (JAVA) • Hands • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Testcontainers CI • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Testcontainers CI, Multi-Stage JRE Images & Zero-Downtime AWS Cloud Rollouts in real Java Full Stack Development & Enterprise Cloud Architecture s...

What Topic Is Next → Next Topic 22.2: OpenTelemetry Distributed Tracing, Micrometer Prometheus Dashboards & JFR Profiling

22.2 OpenTelemetry Distributed Tracing, Micrometer Prometheus Dashboards & JFR Profiling [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Subtopics Covered: Security Hygiene & Data Safety Rules for OpenTelemetry Distributed Tracing, Micrometer Prometheus Dashboards & JFR.... • Performance Optimization, Complexity & Latency Impact of OpenTelemetry Distributed Tracing, Micrometer Prometheus D... • Live Debugging, Stack • Runnable Code Implementation (JAVA) • Hands • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for OpenTelemetry Distributed Tracing • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply OpenTelemetry Distributed Tracing, Micrometer Prometheus Dashboards & JFR Profiling in real Java Full Stack Development & Enterprise Cloud Archite...

What Topic Is Next → Next Topic 22.3: Full Stack Java Architect System Design, Live Coding & Placement Interview Defense

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

Subtopics Covered: Security Hygiene & Data Safety Rules for Full Stack Java Architect System Design, Live Coding & Placement Interview... • Performance Optimization, Complexity & Latency Impact of Full Stack Java Architect System Design, Live Coding & Pla... • Live Debugging, Stack • Runnable Code Implementation (JAVA) • Hands • Idiomatic JFSC Implementation: constructor DI, Record DTOs, @Transactional boundaries & Resilience4j • Java 21 + Spring Boot 3 IoC & Hibernate 6 Persistence Context Execution & Memory Lifecycle for Full Stack Java Architect System Design • Production Hardening & Security: Spring Security 6 JWT/OAuth2 RBAC & OWASP API Top-10

Purpose: To understand and apply Full Stack Java Architect System Design, Live Coding & Placement Interview Defense in real Java Full Stack Development & Enterprise Cloud Architec...

What Topic Is Next → Module 22 Capstone → End-of-Course Live Client Project: FinCore Enterprise Multi-Tenant Banking, Spring AI Copilot & Kafka Settlement Platform

Module 22 Hands-On Capstone Project (100 Marks): Docker + EKS/AWS CI/CD Deployment, OpenTelemetry Tracing & Mock Hiring Loop Repo — Architect, implement, test, and document 'Docker + EKS/AWS CI/CD Deployment, OpenTelemetry Tracing & Mock Hiring Loop Repo' applying every bullet-wise engineering concept from Module 22 (AWS...

3. LIVE CLIENT PRODUCTION PROJECT & CAREER SCOPE

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

Project: FinCore Enterprise Multi-Tenant Banking, Spring AI Copilot & Kafka Settlement Platform • **Confirmed Role:** Full Stack Java, Spring AI & AWS Cloud Engineering Resident

Domain & Production Infra: BFSI, Enterprise Cloud Banking & AI Operations Client (Core-to-Java 21 + Spring Boot 3.3 + Spring AI pgvector + React 19 TS + Kafka + AWS EKS)

- Engineered Java 21 Virtual-Thread Spring Boot 3.3 microservices with OAuth2/OIDC security and Flyway PostgreSQL ledgers.
- Built a grounded Spring AI + pgvector RAG compliance copilot with streaming React 19 TypeScript operator workspaces.
- Orchestrated idempotent Apache Kafka payment sagas, Redis 7 rate limiters, and Testcontainers CI suites.
- Deployed multi-service production workloads onto AWS VPC/EKS via Terraform and GitHub Actions CI/CD.

Target Job Roles: Full Stack Java & Cloud Developer (SDE-1 / SDE-2), Spring Boot + Spring AI Backend Engineer, Java Microservices & AWS Engineer, Enterprise Full Stack Software Engineer

Realistic Annual CTC Scope: **Placement-Guaranteed Graduate / SDE-1 (0–1 Yr):** ₹6.5 LPA – ₹14.0 LPA **Full Stack Java + Cloud SDE-2 (1–3 Yrs):** ₹15.0 LPA – ₹28.0 LPA **Principal Enterprise Cloud Architect (3–6+ Yrs):** ₹30.0 LPA – ₹60.0+ LPA

Optional Worthwhile Global Certifications (100% Exam-Success Mentorship Available): **Oracle Certified Professional: Java SE 21 Developer (1Z0-830):** ₹10.0 LPA – ₹24.0 LPA (+25–35% CTC Edge) • **AWS Certified Solutions Architect – Associate (SAA-C03):** ₹12.0 LPA – ₹28.0 LPA (+30–40% Cloud Premium) • **Certified Kubernetes Application Developer (CKAD):** ₹15.0 LPA – ₹32.0 LPA (Hands-On Lab Credential)

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

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