Admissions & Support: support@mockattempt.com
Phone / WhatsApp: +91 98765 43210
Course Code: SPBT • **Updated:** 01 Nov 2026
https://www.mockattempt.com/courses/springboot-training

SPBT • JAVA ENTERPRISE BACKEND & MICROSERVICES ENGINEERING | **BEGINNER TO PRINCIPAL SPRING BOOT 3 & JAVA 21 ARCHITECT**

100% LIVE INDUSTRY EXPERT COHORT + INTERNSHIP LETTER

Springboot Training

Master Spring Boot 3 from IoC/DI, Starters, Auto-Configuration & DispatcherServlet REST APIs to ProblemDetail Exception Handling, Jakarta Validation, Spring AOP, HikariCP + Spring Data JPA (solving N+1), Caching, Spring Security 6 JWT & Actuator — with runnable Java 21 programs, 10×50 MCQ Mock Tests per topic, and Spoken Voice + Gemini AI Interviews at ₹4,499.

Duration: 16–18 Weeks • Live + Hands-On Projects (208 Hrs) • **Modules:** 16 • **Live Topics:** 53 • **Labs & Tasks:** 106 • **Mocks / AI Vivas:** 386 / 386 • **Projects:** 16 Module + 1 Live Client Prod

Fixed Non-Negotiable Tuition Fee: Base Fee **₹4,499** + 18% GST (₹810) = **Total Incl. GST: ₹5,309**

Live Batch Tracks: Weekday Live Online Batch (Mon – Fri) (Monday to Friday (5 Live Interactive Classes / Week) • 1.5 Hours / Day (60 Mins Live Architecture & Coding + 30 Mins Hands-On Lab) • 17–19 Weeks Fast-Track) | Weekend Intensive Live Online Batch (Sat – Sun) (Saturday & Sunday (Designed for Working Professionals & College Students) • 3.0 Hours / Day (2 Hrs Deep-Dive Live Coding + 1 Hr Guided Project Build) • 19–21 Weeks Weekend Track)

Core Technologies & Engineering Stack: Spring Boot 3.4 • Java 21 Virtual Threads • Spring Web MVC & REST • Spring Data JPA & Hibernate 6 • Spring AOP & Caching • Spring Security 6 & JWT • Actuator, OpenAPI 3 & JUnit 5

1. PROGRAM MASTERY OUTCOMES & ELIGIBILITY

End-to-End Engineering Outcomes:

1. Understand Spring Boot Auto-Configuration (@ConditionalOnClass, @ConditionalOnMissingBean), Constructor DI, and Embedded Tomcat with Java 21 Virtual Threads
2. Engineer production RESTful APIs with @RestController, ResponseEntity, @Valid Jakarta Validation, RFC-9457 ProblemDetail @RestControllerAdvice, and OpenAPI 3 Swagger UI
3. Eliminate Hibernate N+1 query bottlenecks using FetchType.LAZY + @EntityGraph, guard @Transactional CGLIB proxy boundaries, and accelerate reads with @Cacheable
4. Clear 350 Timed Mock Exams (17,500 Spring Boot MCQs) and 350 Spoken Voice + Gemini AI Java Backend Technical Interviews
5. Clean Spring Boot 3.3 architecture with constructor injection, immutable DTO records, and Bean Validation (`@Valid`).
6. Spring Security 6 stateless OAuth2/JWT filter chains, CORS configuration, and method-level RBAC authorization.
7. Hibernate / Spring Data JPA performance: solving N+1 queries, optimistic locking, projections, and Flyway migrations.
8. Resilient external integrations with RestClient, Resilience4j retries/circuit breakers, and Spring Cache.

Eligibility & Prerequisites: Basic Core Java OOP & Collections familiarity — covers IoC, Maven Starters, and Spring Boot 3 fundamentals step-by-step from scratch • Any laptop/PC with JDK 21, Maven/Gradle, and IntelliJ IDEA Community / STS / VS Code • No prior Springboot industry experience required — Module 1 starts from environment setup, core mental models, and plain-English fundamentals. • Basic computer & internet familiarity on a Windows, macOS, or Linux laptop (8 GB+ RAM recommended for smooth live coding labs). • Ideal for B.Tech / BCA / MCA / B.Sc IT students, non-CS beginners, career switchers, and working developers upgrading to 2026 production standards. • Commitment to attend live Weekday or Weekend classes, complete daily hands-on lab checklists, and practice topic-wise Mock Tests & AI Interviews.

2. DETAILED MODULE & TOPIC-WISE SYLLABUS (16 MODULES • 53 LIVE TOPICS)

MODULE 1 OF 16 • 12 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 1: Spring Boot 3 Architecture, IoC/DI Foundations & Spring vs Spring Boot vs Spring MVC

Understand Inversion of Control (IoC), why Spring Boot eliminates boilerplate XML/WAR ceremony over classic Spring Framework, how the 4-layer enterprise architecture flows, and Java 21 Virtual Threads.

Objectives: ✓ Contrast Spring Core IoC, Spring Web MVC, and Spring Boot opinionated auto-configuration • ✓ Trace an HTTP request cleanly across Controller -> Service -> Repository -> Database layers • ✓ Configure SpringApplication startup hooks, banners, profiles, and Java 21 Virtual Threads

1.1 Spring Boot 3 Overview — Opinionated Defaults, Embedded Servers & Zero-XML Philosophy [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Spring Boot 3 Overview — Opinionated Defaults, Embedded Servers & Zero-XML Philosophy in real Springboot systems — specifically: How Spring Boot w...

1.2 Spring Framework vs Spring MVC vs Spring Boot — IoC Container, `DispatcherServlet` & Auto-Setup [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Spring Framework vs Spring MVC vs Spring Boot — IoC Container, `DispatcherServlet` & Auto-Setup in real Springboot systems — specifically: Clear a...

1.3 Spring Boot 4-Layer Architecture — Presentation, Business Service, Persistence & Database Flow [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Spring Boot 4-Layer Architecture — Presentation, Business Service, Persistence & Database Flow in real Springboot systems — specifically: Strict s...

1.4 Core Spring Boot 3.4 Features — `SpringApplication`, `CommandLineRunner`, Profiles & Java 21 Virtual Threads [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Core Spring Boot 3.4 Features — `SpringApplication`, `CommandLineRunner`, Profiles & Java 21 Virtual Threads in real Springboot systems — specific...

Module 1 Hands-On Capstone Project (100 Marks): Layered Course Catalog Engine with Constructor DI & Java 21 Virtual Threads — Architect a 4-layer Spring Boot 3.4 service (`CourseController` -> `CourseService` -> `CourseRepository` -> immutable `CourseRecord`) using zero `@Autowired` field injection, `spring.threads...

MODULE 2 OF 16 • 11 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 2: Project Bootstrapping (`Spring Initializr`), Maven Starters, Executable Fat JARs & DevTools

Bootstrap clean Maven/Gradle Spring Boot projects, master BOM dependency management via Starters, understand `BOOT-INF` Nested Fat JAR classloading + Docker Layered JARs, and accelerate local loops with DevTools.

Objectives: ✓ Generate multi-dependency Maven/Gradle builds via `start.spring.io` targeting Java 21 LTS • ✓ Resolve dependency version conflicts effortlessly through `spring-boot-starter-parent` BOM inheritance • ✓ Inspect `spring-boot-maven-plugin` `repackage` Fat JAR layout (`BOOT-INF/classes`, `BOOT-INF/lib`) vs Tomcat WAR deployment

2.1 `Spring Initializr` (`start.spring.io`) — Project Metadata, Maven Wrapper (`mvnw`) & Directory Layout [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply `Spring Initializr` (`start.spring.io`) — Project Metadata, Maven Wrapper (`mvnw`) & Directory Layout in real Springboot systems — specifically: G...

2.2 Spring Boot Starters & `spring-boot-starter-parent` BOM Dependency Management [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Spring Boot Starters & `spring-boot-starter-parent` BOM Dependency Management in real Springboot systems — specifically: How opinionated `spring-b...

2.3 Executable Fat JARs (`JarLauncher`, `BOOT-INF`) vs External Tomcat WAR Deployment [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Executable Fat JARs (`JarLauncher`, `BOOT-INF`) vs External Tomcat WAR Deployment in real Springboot systems — specifically: How `spring-boot-mave...

2.4 Spring Boot DevTools (`RestartClassLoader`), LiveReload & Rapid Local Iteration [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Spring Boot DevTools (`RestartClassLoader`), LiveReload & Rapid Local Iteration in real Springboot systems — specifically: How `spring-boot-devtoo...

Module 2 Hands-On Capstone Project (100 Marks): Docker-Layered Spring Boot 3 Maven Build & Custom Company Starter Blueprint — Configure a Java 21 `pom.xml` inheriting `spring-boot-starter-parent:3.4.x`, extract layered JAR segments (`dependencies`, `spring-boot-loader`, `application`) for fast 2-second Docker image...

MODULE 3 OF 16 • 13 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 3: Auto-Configuration Internals, Core Stereotype Annotations & Externalized Configuration

Demystify `@SpringBootApplication`, master Constructor Injection + Bean Scopes, inspect `AutoConfiguration.imports` + `@ConditionalOn*` evaluation reports (`--debug`), and bind type-safe immutable `@ConfigurationProperties` Java 21 `record`s.

Objectives: ✓ Deconstruct `@SpringBootApplication` into `@SpringBootConfiguration`, `@EnableAutoConfiguration`, and `@ComponentScan` • ✓ Read `META-INF/spring/org.springframework.boot.autoconfigure.AutoConfiguration.imports` and `--debug` ConditionEvaluationReport • ✓ Bind and validate `application.yml` hierarchies into immutable `@ConfigurationProperties` Java `record`s with `@Validated`

3.1 Core Stereotype & DI Annotations — `@SpringBootApplication`, `@Service`, `@Bean`, `@Qualifier` & Scopes [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Core Stereotype & DI Annotations — `@SpringBootApplication`, `@Service`, `@Bean`, `@Qualifier` & Scopes in real Springboot systems — specifically:...

3.2 Spring Boot 3 Auto-Configuration Internals — `AutoConfiguration.imports` & `@ConditionalOn*` Guards [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Spring Boot 3 Auto-Configuration Internals — `AutoConfiguration.imports` & `@ConditionalOn*` Guards in real Springboot systems — specifically: How...

3.3 Externalized Config — `application.yml`, Property Precedence, `@Profile` & `@ConfigurationProperties` Records [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Externalized Config — `application.yml`, Property Precedence, `@Profile` & `@ConfigurationProperties` Records in real Springboot systems — specifi...

3.4 Embedded Server Tuning — `server.port`, Random Test Ports (`0`), Graceful Shutdown & Compression [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Embedded Server Tuning — `server.port`, Random Test Ports (`0`), Graceful Shutdown & Compression in real Springboot systems — specifically: Config...

Module 3 Hands-On Capstone Project (100 Marks): Type-Safe Multi-Environment Payment Gateway Auto-Configuration & `@ConfigurationProperties` Engine — Create an immutable `@Validated @ConfigurationProperties(prefix = "academy.payment") record PaymentProps(...)` failing fast at startup if API timeouts or URLs are invalid, paired with `@Prof...

MODULE 4 OF 16 • 13 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 4: Spring Web MVC, `DispatcherServlet` Lifecycle & Production RESTful Web Services

Master `spring-boot-starter-web`, trace the `DispatcherServlet` Front Controller lifecycle step-by-step, and build clean HTTP REST endpoints with `@RestController`, `ResponseEntity<T>`, `201 Created` `Location` headers, and idempotent verbs.

Objectives: ✓ Trace incoming HTTP requests through `DispatcherServlet` -> `HandlerMapping` -> `HandlerAdapter` -> `HttpMessageConverter` (Jackson) • ✓ Contrast `@Controller` (View resolution) vs `@RestController` (`@Controller + @ResponseBody`) • ✓ Design semantic `GET`, `POST` (`201 Created` + `Location` URI), `PUT`, `PATCH`, and `DELETE` (`204 No Content`) endpoints

4.1 `spring-boot-starter-web` — Embedded Tomcat, Jackson `ObjectMapper` & MVC Auto-Wiring [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply `spring-boot-starter-web` — Embedded Tomcat, Jackson `ObjectMapper` & MVC Auto-Wiring in real Springboot systems — specifically: What `spring-boot...

4.2 `DispatcherServlet` Front Controller Lifecycle — `HandlerMapping`, `HandlerInterceptor` & Message Converters [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply `DispatcherServlet` Front Controller Lifecycle — `HandlerMapping`, `HandlerInterceptor` & Message Converters in real Springboot systems — specific...

4.3 Routing & Parameter Binding — `@PathVariable`, `@RequestParam`, `@RequestHeader` & `ResponseEntity<T>` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Routing & Parameter Binding — `@PathVariable`, `@RequestParam`, `@RequestHeader` & `ResponseEntity<T>` in real Springboot systems — specifically:...

4.4 Resource Mutation — `@PostMapping` (`201 Created` + `Location`), `PUT` vs `PATCH` & `@DeleteMapping` (`204`) [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Resource Mutation — `@PostMapping` (`201 Created` + `Location`), `PUT` vs `PATCH` & `@DeleteMapping` (`204`) in real Springboot systems — specific...

Module 4 Hands-On Capstone Project (100 Marks): RESTful Enrollment & Cohort Management API (`201 Created` + `Location` + Cursor Pagination) — Build an idiomatic `/api/v1/cohorts` REST controller returning `ResponseEntity<CohortDto>` with `ServletUriComponentsBuilder` `Location` headers on `POST`, `204 No Content` on `DELETE`, and...

MODULE 5 OF 16 • 13 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 5: Global Exception Handling (`ProblemDetail`), Jakarta Validation, DTO Filtering & API Versioning

Centralize error handling with `@RestControllerAdvice` + Spring 6 RFC-9457 `ProblemDetail`, validate nested DTOs with Jakarta Bean Validation (`@Valid`), prevent Mass-Assignment/data leaks with DTO projections, and version APIs cleanly.

Objectives: ✓ Standardize API error responses using `@RestControllerAdvice`, `@ExceptionHandler`, and native Spring Boot 3 `ProblemDetail` • ✓ Enforce input constraints with `spring-boot-starter-validation` (`@Valid`, `@NotBlank`, `@Positive`, custom `ConstraintValidator`) • ✓ Compare DTO Records vs `@JsonIgnore` / `@JsonView` / `MappingJacksonValue` dynamic filtering, URI/Header versioning, and HATEOAS

5.1 Global Exception Handling — `@RestControllerAdvice`, `@ExceptionHandler` & RFC-9457 `ProblemDetail` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Global Exception Handling — `@RestControllerAdvice`, `@ExceptionHandler` & RFC-9457 `ProblemDetail` in real Springboot systems — specifically: Rep...

5.2 Jakarta Bean Validation (`spring-boot-starter-validation`) — `@Valid`, Nested DTOs & Custom Validators [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Jakarta Bean Validation (`spring-boot-starter-validation`) — `@Valid`, Nested DTOs & Custom Validators in real Springboot systems — specifically:...

5.3 DTO Projections, Static (`@JsonIgnore`) & Dynamic (`MappingJacksonValue`) Filtering + Content Negotiation [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply DTO Projections, Static (`@JsonIgnore`) & Dynamic (`MappingJacksonValue`) Filtering + Content Negotiation in real Springboot systems — specificall...

5.4 REST API Versioning Strategies, `Spring HATEOAS` (`EntityModel`) & Internationalization (`i18n`) [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply REST API Versioning Strategies, `Spring HATEOAS` (`EntityModel`) & Internationalization (`i18n`) in real Springboot systems — specifically: Four i...

Module 5 Hands-On Capstone Project (100 Marks): RFC-9457 `ProblemDetail` Error Engine, Custom `@ValidCoupon` Validator & Versioned REST API — Implement a global `@RestControllerAdvice` mapping `ResourceNotFoundException` (`404`), `MethodArgumentNotValidException` (`400` with per-field violation map), and `DataIntegrityViolationExc...

MODULE 6 OF 16 • 11 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 6: Aspect-Oriented Programming (`Spring AOP`) — CGLIB Proxies, Pointcuts & Cross-Cutting Advices

Modularize cross-cutting concerns (latency telemetry, audit logging, rate-limiting, retry policies) cleanly using `Spring AOP`, understand CGLIB vs JDK Dynamic Proxies + the Self-Invocation trap, and write `@Before`, `@AfterReturning`, `@AfterThrowing`, and `@Around` aspects.

Objectives: ✓ Understand how Spring Boot defaults to **CGLIB subclass proxies** (`proxyTargetClass = true`) and why `private`/`final`/self-invoked methods bypass AOP • ✓ Write targeted custom annotation pointcuts (`@annotation(Audited)`) instead of fragile broad wildcard expressions • ✓ Build `@Around` latency & resilience interceptors safely invoking `ProceedingJoinPoint.proceed()`

6.1 Spring AOP Architecture — CGLIB Subclass Proxies, JoinPoints, Pointcuts & Self-Invocation Trap [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Spring AOP Architecture — CGLIB Subclass Proxies, JoinPoints, Pointcuts & Self-Invocation Trap in real Springboot systems — specifically: Core AOP...

6.2 `@Before`, `@After`, `@AfterReturning` & `@AfterThrowing` — Security, Audit & Failure Logging Advices [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply `@Before`, `@After`, `@AfterReturning` & `@AfterThrowing` — Security, Audit & Failure Logging Advices in real Springboot systems — specifically: C...

6.3 `@Around` Advice with `ProceedingJoinPoint` — SLA Latency Profiling & Idempotent Retry Interceptors [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply `@Around` Advice with `ProceedingJoinPoint` — SLA Latency Profiling & Idempotent Retry Interceptors in real Springboot systems — specifically: The...

Module 6 Hands-On Capstone Project (100 Marks): Custom `@TrackLatency` & `@AuditAction` Spring AOP Telemetry Suite — Create a custom `@Target(ElementType.METHOD) @Retention(RetentionPolicy.RUNTIME) @interface TrackLatency` annotation and an `@Aspect` using `@Around` (`ProceedingJoinPoint`) that logs method...

MODULE 7 OF 16 • 15 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 7: Data Persistence — `HikariCP`, `JdbcClient`, H2 Database, `Spring Data JPA` & Solving `N+1`

Connect to relational databases via ultra-fast `HikariCP`, write fluent SQL with Spring 6.1 `JdbcClient` / `JdbcTemplate`, run rapid local/test schemas on H2 + Flyway, and master `Spring Data JPA` (`JpaRepository`, `@Transactional`, `FetchType.LAZY`, `@EntityGraph`).

Objectives: ✓ Size `HikariCP` connection pools accurately and execute parameterized SQL via Spring Boot 3.2+ `JdbcClient` • ✓ Use H2 in-memory MySQL/PostgreSQL compatibility modes locally while enforcing `ddl-auto: validate` + Flyway in production • ✓ Master `Spring Data JPA` dirty checking, `@Transactional(readOnly = true)`, `FetchType.LAZY` on `@ManyToOne`, and `@EntityGraph` / DTO projections to eliminate `N+1` queries

7.1 Spring JDBC & `HikariCP` — Connection Pool Sizing & Modern `JdbcClient` / `JdbcTemplate` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Spring JDBC & `HikariCP` — Connection Pool Sizing & Modern `JdbcClient` / `JdbcTemplate` in real Springboot systems — specifically: Why Spring Boo...

7.2 Embedded H2 Database & Schema Migrations — Dev Prototyping vs Production `Flyway` / `Liquibase` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Embedded H2 Database & Schema Migrations — Dev Prototyping vs Production `Flyway` / `Liquibase` in real Springboot systems — specifically: Running...

7.3 `Spring Data JPA` & Hibernate 6 — `@Entity`, `JpaRepository`, `@Query`, Pagination & `@Transactional` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply `Spring Data JPA` & Hibernate 6 — `@Entity`, `JpaRepository`, `@Query`, Pagination & `@Transactional` in real Springboot systems — specifically: J...

7.4 JPA Entity Relationships (`@ManyToOne`, `@OneToMany`) — `FetchType.LAZY` & Crushing the `N+1` Trap [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply JPA Entity Relationships (`@ManyToOne`, `@OneToMany`) — `FetchType.LAZY` & Crushing the `N+1` Trap in real Springboot systems — specifically: Bidi...

Module 7 Hands-On Capstone Project (100 Marks): Zero-`N+1` Spring Data JPA Academy Ledger with Optimistic `@Version` Locking — Model `Course` (1) <-> (N) `CohortBatch` entities with explicit `FetchType.LAZY` on `@ManyToOne`, fetch paginated DTO summaries in 1 SQL query via JPQL Constructor/Record Projection, disable...

MODULE 8 OF 16 • 11 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 8: High-Speed Caching (`@Cacheable`), Cache Providers (`Caffeine` / `Redis`) & `Thymeleaf` SSR

Slash database read load by 95% using Spring's Cache Abstraction (`@EnableCaching`, `@Cacheable`, `@CachePut`, `@CacheEvict`), compare in-memory `Caffeine`/`EhCache` vs distributed `Redis`, and render XSS-safe HTML with `Thymeleaf`.

Objectives: ✓ Apply `@Cacheable(sync = true)` to prevent Cache Stampede (Thundering Herd) on hot catalog keys • ✓ Contrast ultra-fast L1 JVM `Caffeine` / `EhCache` caches against multi-pod shared L2 `RedisCacheManager` with TTLs • ✓ Render server-side views & transactional email templates safely via `spring-boot-starter-thymeleaf`

8.1 Spring Cache Abstraction — `@EnableCaching`, `@Cacheable(sync = true)`, `@CachePut` & `@CacheEvict` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Spring Cache Abstraction — `@EnableCaching`, `@Cacheable(sync = true)`, `@CachePut` & `@CacheEvict` in real Springboot systems — specifically: Dec...

8.2 Cache Providers (`Caffeine`, `EhCache 3`, `Redis`) — Local JVM Heap Speed vs Distributed Consistency [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Cache Providers (`Caffeine`, `EhCache 3`, `Redis`) — Local JVM Heap Speed vs Distributed Consistency in real Springboot systems — specifically: Ch...

8.3 Server-Side Rendering with `Thymeleaf` — Natural HTML5 Templates, `th:text` XSS Escaping & Emails [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Server-Side Rendering with `Thymeleaf` — Natural HTML5 Templates, `th:text` XSS Escaping & Emails in real Springboot systems — specifically: Retur...

Module 8 Hands-On Capstone Project (100 Marks): Two-Tier Course Catalog Cache (`Caffeine` + TTL Eviction) & `Thymeleaf` Certificate Previewer — Configure `@EnableCaching` with `CaffeineCacheManager` (`maximumSize=1000, expireAfterWrite=10m`), guard hot course lookups with `@Cacheable(cacheNames = "courses", key = "#slug", sync = tru...

MODULE 9 OF 16 • 13 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 9: OpenAPI 3 / Swagger UI, `Spring Security 6` Filter Chains & `Actuator` Observability

Auto-generate interactive OpenAPI 3.1 Swagger docs (`springdoc-openapi`), lock down microservices with modern `Spring Security 6` Lambda DSL (`SecurityFilterChain`, Stateless JWT, `@PreAuthorize`, `BCrypt`), and expose Kubernetes probes + Prometheus metrics via `Actuator`.

Objectives: ✓ Document REST contracts & JWT Bearer auth in Swagger UI using `springdoc-openapi-starter-webmvc-ui` (never abandoned Springfox!) • ✓ Configure `Spring Security 6` `SecurityFilterChain` beans using Lambda DSL, `SessionCreationPolicy.STATELESS`, `BCryptPasswordEncoder`, and `@EnableMethodSecurity` • ✓ Operate `Spring Boot Actuator` (`/actuator/health/readiness`, `/actuator/health/liveness`, `/actuator/prometheus`) securely

9.1 OpenAPI 3.1 & Swagger UI (`springdoc-openapi`) — `@Operation`, `@Schema` & Bearer JWT Docs [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply OpenAPI 3.1 & Swagger UI (`springdoc-openapi`) — `@Operation`, `@Schema` & Bearer JWT Docs in real Springboot systems — specifically: Replacing de...

9.2 `Spring Security 6` Architecture — `SecurityFilterChain`, Lambda DSL, Stateless JWT, CORS & `@PreAuthorize` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply `Spring Security 6` Architecture — `SecurityFilterChain`, Lambda DSL, Stateless JWT, CORS & `@PreAuthorize` in real Springboot systems — specifica...

9.3 `Spring Boot Actuator` & Micrometer — Kubernetes Liveness/Readiness Probes, Custom `HealthIndicator` & Prometheus [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply `Spring Boot Actuator` & Micrometer — Kubernetes Liveness/Readiness Probes, Custom `HealthIndicator` & Prometheus in real Springboot systems — spe...

Module 9 Hands-On Capstone Project (100 Marks): Hardened `Spring Security 6` Stateless JWT Resource Server + `Actuator` + OpenAPI 3 Gateway — Configure a `SecurityFilterChain` bean permitting `/api/v1/public/**` and `/actuator/health/**`, requiring `ROLE_ADMIN` via `@PreAuthorize("hasRole('ADMIN')")` on curriculum mutations, hashi...

MODULE 10 OF 16 • 12 LIVE + LAB HOURS • 2 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 10: Automated Testing (`@SpringBootTest`, Slice Tests) & Cloud-Native REST Best Practices

Ship bug-free releases with `spring-boot-starter-test` (`JUnit 5`, `Mockito`, `AssertJ`), fast Test Slices (`@WebMvcTest` `MockMvc`, `@DataJpaTest`, `Testcontainers`), and master Richardson Maturity Model + Cloud-Native 12-Factor production standards.

Objectives: ✓ Apply the Spring Testing Pyramid: 70% pure `JUnit 5 + Mockito` unit tests (< 10 ms), 20% Slice tests (`@WebMvcTest`, `@DataJpaTest`), and 10% `@SpringBootTest` + `Testcontainers` integration tests • ✓ Verify controller status codes, `@Valid` errors, and JSON paths using `MockMvc` / `RestTestClient` • ✓ Audit REST APIs against Richardson Maturity Model Levels 0–3, Idempotency, Rate Limiting, and 12-Factor Cloud readiness

10.1 Spring Boot Testing Pyramid — `JUnit 5`, `Mockito`, `@WebMvcTest` (`MockMvc`), `@DataJpaTest` & `@SpringBootTest` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Spring Boot Testing Pyramid — `JUnit 5`, `Mockito`, `@WebMvcTest` (`MockMvc`), `@DataJpaTest` & `@SpringBootTest` in real Springboot systems — spe...

10.2 Richardson Maturity Model (Levels 0–3) & Production Cloud-Native Spring Boot Best Practices [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Richardson Maturity Model (Levels 0–3) & Production Cloud-Native Spring Boot Best Practices in real Springboot systems — specifically: Evaluating...

Module 10 Hands-On Capstone Project (100 Marks): Full Testing Pyramid Suite (`Mockito` Unit + `@WebMvcTest` `MockMvc` + `@DataJpaTest`) & Production Readiness Gate — Write a 3-tier automated test suite for the Enrollment Service: (1) pure `@ExtendWith(MockitoExtension.class)` unit test verifying business rules in 5 ms, (2) `@WebMvcTest` `MockMvc` slice t...

MODULE 11 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Spring Security 6 FilterChains, OAuth2/JWT & Method Authorization

Production-grade Springboot Training engineering module covering Spring Security 6 FilterChains, OAuth2/JWT & Method Authorization with hands-on labs and capstone build.

Objectives: ✓ Modern Lambda SecurityFilterChain, CORS & Password Encoders — Purpose: Secure REST endpoints cleanly with Argon2/BCrypt and structured ProblemDetail 401/403s. • ✓ RSA JWT Issuance, Refresh Token Rotation & OAuth2 Resource Server — Purpose: Verify signed claims and tenant IDs across endpoints with zero session memory bloat. • ✓ @PreAuthorize Method RBAC & Security Audit Listeners — Purpose: Enforce fine-grained ownership checks directly on business service methods.

11.1 Modern Lambda SecurityFilterChain, CORS & Password Encoders [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Modern Lambda SecurityFilterChain, CORS & Password Encoders in real Springboot systems — specifically: Secure REST endpoints cleanly with Argon2/B...

11.2 RSA JWT Issuance, Refresh Token Rotation & OAuth2 Resource Server [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply RSA JWT Issuance, Refresh Token Rotation & OAuth2 Resource Server in real Springboot systems — specifically: Verify signed claims and tenant IDs a...

11.3 @PreAuthorize Method RBAC & Security Audit Listeners [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply @PreAuthorize Method RBAC & Security Audit Listeners in real Springboot systems — specifically: Enforce fine-grained ownership checks directly on...

Module 11 Hands-On Capstone Project (100 Marks): Stateless JWT + Role-Scoped Enterprise Spring Boot API — Design, implement, test, and document 'Stateless JWT + Role-Scoped Enterprise Spring Boot API' applying every production concept from Module 11 (Spring Security 6 FilterChains, OAuth2/JWT &...

MODULE 12 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Hibernate/JPA Performance, Flyway Migrations & Redis Caching

Production-grade Springboot Training engineering module covering Hibernate/JPA Performance, Flyway Migrations & Redis Caching with hands-on labs and capstone build.

Objectives: ✓ Eliminating N+1 Queries via @EntityGraph, JOIN FETCH & Record DTOs — Purpose: Slash 150 lazy-loading SQL queries down to 1 indexed projection query. • ✓ Optimistic Locking (@Version), HikariCP Tuning & Flyway SQL Migrations — Purpose: Prevent concurrent lost updates and version every schema change reproducibly. • ✓ Spring @Cacheable with Redis TTLs & Targeted Eviction — Purpose: Serve hot reference and catalog endpoints in < 4ms under

peak concurrency.

12.1 Eliminating N+1 Queries via @EntityGraph, JOIN FETCH & Record DTOs [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Eliminating N+1 Queries via @EntityGraph, JOIN FETCH & Record DTOs in real Springboot systems — specifically: Slash 150 lazy-loading SQL queries d...

12.2 Optimistic Locking (@Version), HikariCP Tuning & Flyway SQL Migrations [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Optimistic Locking (@Version), HikariCP Tuning & Flyway SQL Migrations in real Springboot systems — specifically: Prevent concurrent lost updates...

12.3 Spring @Cacheable with Redis TTLs & Targeted Eviction [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Spring @Cacheable with Redis TTLs & Targeted Eviction in real Springboot systems — specifically: Serve hot reference and catalog endpoints in < 4m...

Module 12 Hands-On Capstone Project (100 Marks): N+1-Free Order Engine with Flyway & Two-Tier Redis Cache — Design, implement, test, and document 'N+1-Free Order Engine with Flyway & Two-Tier Redis Cache' applying every production concept from Module 12 (Hibernate/JPA Performance, Flyway Migration...

MODULE 13 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Async Messaging (Kafka/RabbitMQ), Resilience4j & Scheduled Locks

Production-grade Springboot Training engineering module covering Async Messaging (Kafka/RabbitMQ), Resilience4j & Scheduled Locks with hands-on labs and capstone build.

Objectives: ✓ Spring Kafka Producers, Consumer Groups, Retries & Dead-Letter Topics — Purpose: Decouple slow invoice/notification pipelines from synchronous customer APIs. • ✓ Resilience4j CircuitBreaker, Bulkhead & Exponential Backoff Retry — Purpose: Isolate degraded third-party gateways so upstream Spring threads never hang. • ✓ Transactional Outbox & ShedLock Distributed Cron Jobs — Purpose: Guarantee reliable event publishing and single-pod execution across scaled replicas.

13.1 Spring Kafka Producers, Consumer Groups, Retries & Dead-Letter Topics [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Spring Kafka Producers, Consumer Groups, Retries & Dead-Letter Topics in real Springboot systems — specifically: Decouple slow invoice/notificatio...

13.2 Resilience4j CircuitBreaker, Bulkhead & Exponential Backoff Retry [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Resilience4j CircuitBreaker, Bulkhead & Exponential Backoff Retry in real Springboot systems — specifically: Isolate degraded third-party gateways...

13.3 Transactional Outbox & ShedLock Distributed Cron Jobs [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Transactional Outbox & ShedLock Distributed Cron Jobs in real Springboot systems — specifically: Guarantee reliable event publishing and single-po...

Module 13 Hands-On Capstone Project (100 Marks): Fault-Tolerant Event-Driven Billing Service with Circuit Breakers — Design, implement, test, and document 'Fault-Tolerant Event-Driven Billing Service with Circuit Breakers' applying every production concept from Module 13 (Async Messaging (Kafka/RabbitMQ),...

MODULE 14 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Testcontainers CI Suites, OpenTelemetry Actuator & Docker/K8s Prod

Production-grade Springboot Training engineering module covering Testcontainers CI Suites, OpenTelemetry Actuator & Docker/K8s Prod with hands-on labs and capstone build.

Objectives: ✓ @WebMvcTest Slices, Mockito & @ServiceConnection Testcontainers — Purpose: Run automated integration tests against real PostgreSQL and Redis containers in CI. • ✓ Actuator Readiness/Liveness Probes, Micrometer Prometheus & Tracing — Purpose: Track p99 latency, JVM heap, Hikari pools, and trace IDs across every request. • ✓ Layered Spring Boot Dockerfiles, Virtual Threads & Graceful Shutdown — Purpose: Deploy fast-booting, zero-downtime Java 21 containers to cloud clusters.

14.1 @WebMvcTest Slices, Mockito & @ServiceConnection Testcontainers [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply @WebMvcTest Slices, Mockito & @ServiceConnection Testcontainers in real Springboot systems — specifically: Run automated integration tests against...

14.2 Actuator Readiness/Liveness Probes, Micrometer Prometheus & Tracing [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Actuator Readiness/Liveness Probes, Micrometer Prometheus & Tracing in real Springboot systems — specifically: Track p99 latency, JVM heap, Hikari...

14.3 Layered Spring Boot Dockerfiles, Virtual Threads & Graceful Shutdown [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Layered Spring Boot Dockerfiles, Virtual Threads & Graceful Shutdown in real Springboot systems — specifically: Deploy fast-booting, zero-downtime...

Module 14 Hands-On Capstone Project (100 Marks): Production-Ready Containerized Spring Boot 3 Release Pipeline — Design, implement, test, and document 'Production-Ready Containerized Spring Boot 3 Release Pipeline' applying every production concept from Module 14 (Testcontainers CI Suites, OpenTelemetr...

MODULE 15 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Spring Boot 3 Production Resilience: Caching, Rate Limiting & Async Events

Self-explanatory list-and-bullet production mastery module covering Spring Boot 3 Production Resilience: Caching, Rate Limiting & Async Events in Springboot Training, paired with hands-on labs, capstone repository, Topic Mock Tests, and Spoken Gemini AI Viva.

Objectives: ✓ RFC 7807 ProblemDetails Global Error Responses & Jakarta Bean Validation Groups — Purpose: Return standardized, client-friendly JSON error contracts across every REST controller. • ✓ Spring Cache Abstraction with Redis TTLs, Cache Stampede Prevention & Eviction — Purpose: Accelerate read-heavy catalog/profile endpoints while keeping cached entries fresh on updates. • ✓ Resilience4j Retries, TimeLimiters, Circuit Breakers & @TransactionalEventListener — Purpose: Isolate downstream service failures and trigger side effects only after DB transactions commit. • ✓ Build, verify, and document 'Resilient Spring Boot 3 Order API with Redis Cache, Circuit Breakers & ProblemDetails' with clean Git commits and edge-case protection. • ✓ Clear 100% of Topic Timed Mock Tests and Spoken Voice + Google Gemini AI Technical Interviews.

15.1 RFC 7807 ProblemDetails Global Error Responses & Jakarta Bean Validation Groups [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply RFC 7807 ProblemDetails Global Error Responses & Jakarta Bean Validation Groups in real Springboot systems — specifically: Return standardized, cl...

15.2 Spring Cache Abstraction with Redis TTLs, Cache Stampede Prevention & Eviction [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Spring Cache Abstraction with Redis TTLs, Cache Stampede Prevention & Eviction in real Springboot systems — specifically: Accelerate read-heavy ca...

15.3 Resilience4j Retries, TimeLimiters, Circuit Breakers & @TransactionalEventListener [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Resilience4j Retries, TimeLimiters, Circuit Breakers & @TransactionalEventListener in real Springboot systems — specifically: Isolate downstream s...

Module 15 Hands-On Capstone Project (100 Marks): Resilient Spring Boot 3 Order API with Redis Cache, Circuit Breakers & ProblemDetails — Architect, implement, test, and document 'Resilient Spring Boot 3 Order API with Redis Cache, Circuit Breakers & ProblemDetails' applying every bullet-wise engineering concept from Module 15...

MODULE 16 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Testcontainers Integration Suites, Micrometer Tracing & Spring Interviews

Self-explanatory list-and-bullet production mastery module covering Testcontainers Integration Suites, Micrometer Tracing & Spring Interviews in Springboot Training, paired with hands-on labs, capstone repository, Topic Mock Tests, and Spoken Gemini AI Viva.

Objectives: ✓ Slice Tests (@WebMvcTest, @DataJpaTest) & Real-DB Integration with Testcontainers — Purpose: Verify JPA queries, Flyway migrations, and security filters against a real containerized database. • ✓ Spring Boot Actuator, Micrometer Metrics & Distributed Trace Propagation — Purpose: Expose production health/readiness endpoints and latency histograms for cloud monitoring. • ✓ Spring IoC/Proxy Pitfalls, N+1 Query Fixes & Enterprise Interview Scenarios — Purpose: Spot @Transactional self-invocation bugs and Hibernate N+1 bottlenecks instantly. • ✓ Build, verify, and document 'Testcontainers PostgreSQL Suite + Actuator Prometheus/OpenTelemetry Stack' with clean Git commits and edge-case protection. • ✓ Clear 100% of Topic Timed Mock Tests and Spoken Voice + Google Gemini AI Technical Interviews.

16.1 Slice Tests (@WebMvcTest, @DataJpaTest) & Real-DB Integration with Testcontainers [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Slice Tests (@WebMvcTest, @DataJpaTest) & Real-DB Integration with Testcontainers in real Springboot systems — specifically: Verify JPA queries, F...

16.2 Spring Boot Actuator, Micrometer Metrics & Distributed Trace Propagation [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Spring Boot Actuator, Micrometer Metrics & Distributed Trace Propagation in real Springboot systems — specifically: Expose production health/readi...

16.3 Spring IoC/Proxy Pitfalls, N+1 Query Fixes & Enterprise Interview Scenarios [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Spring IoC/Proxy Pitfalls, N+1 Query Fixes & Enterprise Interview Scenarios in real Springboot systems — specifically: Spot @Transactional self-in...

Module 16 Hands-On Capstone Project (100 Marks): Testcontainers PostgreSQL Suite + Actuator Prometheus/OpenTelemetry Stack — Architect, implement, test, and document 'Testcontainers PostgreSQL Suite + Actuator Prometheus/OpenTelemetry Stack' applying every bullet-wise engineering concept from Module 16 (Testcontai...

3. LIVE CLIENT PRODUCTION PROJECT, GLOBAL CERTIFICATIONS & CAREER SCOPE

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

Project: MediTrack Secure Enterprise REST API, RBAC Portal & Scheduled Billing Service • **Confirmed Role:** Java Spring Boot 3 Backend Engineering Intern

Domain & Production Infra: Healthcare Clinic Network & Insurance Claims Client (Java 21 + Spring Boot 3.3 + Spring Data JPA + Spring Security 6 JWT + PostgreSQL + Docker)

- Architected layered Spring Boot 3 REST APIs using DTO validation, @ControllerAdvice problem details, and OpenAPI docs.
- Secured endpoints with stateless Spring Security 6 filter chains, JWT access/refresh tokens, and method-level @PreAuthorize.
- Eliminated N+1 Hibernate JPA queries via entity graphs, Flyway migrations, and paginated projections.
- Validated APIs with JUnit 5, MockMvc, and Testcontainers before deploying Spring Actuator health-checked containers.

Aligned Global Vendor Certifications: **2V0-72.22** (Spring Certified Professional (Spring Boot & Enterprise Java)) • **1Z0-830** (Oracle Certified Professional: Java SE 21 Developer)

Target Job Roles: Spring Boot Backend Developer, Java REST API & Microservices Engineer, Enterprise Software Engineer (Java/Spring), Backend SDE-1 / SDE-2

Realistic Annual CTC Scope: **Fresher / Entry-Level (0–1 Yr):** ₹5.5 LPA – ₹10.0 LPA **Mid-Level Spring Microservices Engineer (1–3 Yrs):** ₹10.5 LPA – ₹21.0 LPA **Senior Java Microservices Architect (3–6+ Yrs):** ₹22.0 LPA – ₹42.0+ LPA

MockAttempt Academy Pvt. Ltd. • Email: support@mockattempt.com • Phone/WhatsApp: +91 98765 43210 • Web: https://www.mockattempt.com/courses/springboot-training

Try 2 Free Topic-Wise 50-MCQ Mock Tests & Spoken Gemini AI Interviews daily on every topic above before enrolling.