Curriculum Cum Pre-Requisites:

Note: Curriculum is designed in such a manner that candidate can learn in right chronology. It is not necessary for candidates to develop the projects mentioned here, they can modify the projects according to their choice, Objective is to have hands on experience (utmost necessary) along with conceptual knowledge. Springboot is one of the most preferred framework for Microservices and Enterprise application development, Enjoy learning!
Along with Springboot, day to day work may also involve frontend dev using React.Js, but as most of candidates already have hands on experience on it, they can revise that through their own.

Week 1: Spring Boot Basics, REST APIs & CRUD Operations

- **Topics**:
    - Introduction to Spring Boot and its setup using Spring Initializr
    - Application properties, structure, key annotations (@SpringBootApplication, @Controller, @RestController)
    - Basics of REST APIs and CRUD operations with HTTP methods (GET, POST, PUT, DELETE)
    - Controllers, services, repositories, and Spring Data JPA
    - Connecting to a MySQL database and setting up a basic database configuration
- **Hands-on Project**: Build a simple CRUD API for a "Student" entity with fields like ID, name, email, and department.

Week 2: JPA Deep Dive (Relationships, Transactions, Exception Handling)

- **Topics**:
    - JPA relationships (One-to-One, One-to-Many, Many-to-One, Many-to-Many)
    - Eager vs. Lazy loading and handling N+1 problems
    - Transaction management with @Transactional and its settings (isolation levels, propagation)
    - Exception handling with @ControllerAdvice and @ExceptionHandler for REST APIs
- **Hands-on Project**: Extend the CRUD API with a new "Course" entity and implement relationships with "Student" (One-to-Many, Many-to-One). Add a transaction-based feature for enrolling students in courses.

Week 3: Asynchronous Processing, Thread Pools, and Task Scheduling

- **Topics**:
    - Introduction to multithreading and @Async for async processing
    - Configuring thread pools and understanding ThreadPoolTaskScheduler
    - Basic concepts of thread safety, race conditions, and strategies for managing concurrency

- **Hands-on Project**: Add asynchronous processing to the course enrollment API to send notifications when a student enrolls in a course. Implement a scheduled task to perform periodic maintenance (e.g., clearing inactive enrollments).

Week 4: External API Calls (OpenFeign and RestTemplate)

- **Topics**:
    - Introduction to OpenFeign for external REST API communication
    - Setting up and using RestTemplate for making API calls
    - Error handling and response processing for external APIs
- **Hands-on Project**: Integrate a sample external API (e.g., a weather API), associating a student's profile with their local weather.

Week 5: Spring Security Basics

- **Topics**:
    - Introduction to Spring Security, setting up basic authentication and authorization
    - Securing REST endpoints using annotations (@PreAuthorize, @Secured)
    - Understanding roles, authorities, and JWT (JSON Web Token) basics
- **Hands-on Project**: Secure the CRUD and course enrollment APIs by implementing role-based access, requiring authentication for specific operations.

Week 6: Advanced JPA & Optimization

- **Topics**:
    - JPA query methods, @Query for custom queries
    - Pagination, sorting, and performance optimization techniques (using projections, fetch joins)
    - Basic caching with @Cacheable, caching strategies, and practical use cases
- **Hands-on Project**: Implement pagination and sorting in the course enrollment API. Add caching for frequently accessed data, such as the list of all available courses.

Week 7: Thread Safety, Security Deep Dive, and Best Practices

- **Topics**:
    - Advanced thread safety techniques, synchronizing methods, handling shared resources
    - Deep dive into JWT for securing APIs and enhancing token-based authentication
    - Implementing testing with JUnit and Mockito for unit and integration tests
    - Clean code practices and code optimization for Spring Boot projects
- **Final Project**: Build a comprehensive "Course Management System" API integrating all concepts learned:

- CRUD operations with multiple entities
- Secure, transaction-safe endpoints with role-based access
- Async notifications, external API integration, pagination, sorting, and caching
- Comprehensive testing suite for key features

Add-On Topics (For Early Finishers or Advanced Learners)

Spring Boot Microservices

- **Topics**:
  - Introduction to microservices architecture, advantages, and trade-offs
  - Setting up multiple Spring Boot applications to act as independent services
  - Service discovery with Eureka and load balancing with Ribbon
  - API Gateway setup using Spring Cloud Gateway or Zuul
  - Distributed tracing and monitoring with Spring Cloud Sleuth and Zipkin

Advanced Spring Security

- **Topics**:
  - Implementing OAuth2 for user authentication
  - Refresh tokens and token expiration management
  - Securing inter-service communication in a microservices setup

Message Queues and Event-Driven Microservices

- **Topics**:
  - Introduction to message brokers like RabbitMQ and Kafka
  - Implementing message-driven communication using Spring Boot and RabbitMQ/Kafka
  - Designing event-driven microservices for asynchronous data processing

Spring Boot Testing Best Practices

- **Topics**:
  - Deep dive into testing strategies: unit testing, integration testing, and end-to-end testing
  - Setting up test profiles and mock environments
  - Using Test containers for database testing with real database instances in containers