**Course Description:**

This is an intensive hands-on course aimed at experienced Spring developers. Learn how to quickly get started with Spring applications using Spring Boot. Learn how to arrange, configure and package Spring Boot applications. Learn all about the motivations and benefits of Microservices architecture. Learn all about Spring Cloud components like Zuul, Hysterix, Ribbon and Feign.

**Duration:** 10 Days

**Pre-requisites:**

* Java 8 basics
* This training assumes no prior knowledge of Spring Boot and Spring Cloud, but assumes strong working knowledge of the following:
* Spring Core - Dependency Injection, XML, Annotation and Java Config, ApplicationContext
* Resource Configurations , Datasources etc
* Spring MVC and RESTful Services

**Note :** If these pre-requisites are not being met, it is strongly recommended to go through Core Spring training first

**Recommended Learning Resources:**

* Spring Reference Documentation.

**High Level Course Contents**

* Spring Boot
* Microservices Architecture
* Spring Cloud
* Spring Cloud Components - Eureka, Zuul, Ribbon, Hysterix

**Spring Boot and MicroServices Course Outline**

**Day 1**

**Spring Framework – High Level Overview**

* Spring Architecture
* Spring Containers
* Spring Bean Lifecycle
* Spring DI
* Spring Autowiring
* Spring Data recap
* Spring MVC recap
* Spring Rest API recap

**Day 2**

**Spring Boot - Basics**

* Why Spring Boot
* Spring Boot Set Up
* Installing Spring Boot CLI
* Introduction to Spring Initializer
* Spring Boot Using External Tools
* Custom changes in spring Boot configuration
* Profiles
* Maven Build
* Using the Spring Initializr with UNIX cURL
* Spring Boot Using Spring Tool Suite (STS)
* Your First Spring Boot Application
* How Spring Boot Works

**Spring Boot – Annotations**

* @SpringBootApplication
* @EnableAutoConfiguration
* @ComponentScan
* @ConditionalOnClass
* @ConditionalOnMissingClass
* Quick Recap on Spring Web Annotations
* Quick Recap on Spring Data Annotations

**Day 3**

**Spring Boot – Starters**

* Spring Boot –Web Starter
* Spring Boot – Data Starter
* Spring Boot – Test Starter

**Spring Boot – Auto configuration**

* Disabling a Specifi c Auto-Configuration
* @EnableAutoConfiguration and @Enable<Technology> Annotations
* Spring Boot Features
* SpringApplication Class
* SpringApplicationBuilder
* Application Arguments
* Spring-boot-autoconfigure module
* Exploring spring.factories file
* ApplicationRunner and CommandLineRunner
* Application Configuration
* Configuration Properties Examples

**Day 3**

**Spring Boot CLI**

* Spring Boot CLI
* Spring Boot CLI
* The run Command
* The test Command
* The grab Command
* The jar Command
* The war Command
* The install Command
* The uninstall Command
* The init Command
* The shell Command
* The help Command

**Spring Boot – Actuators**

* Introduction to Actuators
* Configuration of different Endpoints
* Customization of different Endpoints
* Creation of new Endpoints

**Day 4**

**Spring Boot – Database Integration**

* Data Access Using the JDBC Template with Spring Boot
* Data Access Using JPA with Spring Boot
* Auto configuring data sources in Spring Boot
* SQL Databases
* NoSQL Databases
* HicariCP data source
* Tomcat data source
* Dbcp data source
* Dbcp2 data source
* Using Spring Data JPA in Spring Boot web development

**Day 5**

**Web Development with Spring Boot**

* Spring MVC
* Spring Boot Web Applications
* @RestController annotation
* Creating rest end points
* Playing with the HAL Browser

**Spring Boot – Testing**

* Testing Spring Boot
* Web Testing

**Day 6**

**Messaging with Spring Boot**

* What Is Messaging?
* JMS with Spring Boot
* A Simpler JMS Consumer
* Connect to Remote JMS Server
* RabbitMQ with Spring Boot
* Installing RabbitMQ
* RabbitMQ/AMQP: Exchanges, Bindings, and Queues
* Remote RabbitMQ

**Security with Spring Boot**

* Simple Security for Spring Boot
* Security Using the applicationproperties File
* In-Memory Security
* Security Using a Database
* Securing Resources
* Spring Boot with OAuth

**Spring Boot Actuator**

* Spring Boot Actuator
* /actuator
* /autoconfi g
* /beans
* /confi gprops
* /docs
* /dump
* env
* /fl yway
* /health
* /info
* /liquibase
* /logfi le
* /metrics
* /mappings
* /shutdown
* /trace

**Extending Spring Boot Apps**

* Custom Spring Boot Module
* The spring-boot-journal Project
* The journal-spring-boot-starter Project
* The journal-spring-boot-autoconfi gure Project
* Package and Install the Journal Project
* The spring-boot-calendar Project
* Custom Health Indicator

**Day 7**

**Existing Scenarios and Why MicroServices**

* System and service architecture, PaaS, and cloud-native design
* Overview of Monolithic System Architecture
* Running an entire monolithic applications functionality in a single process
* Overview of Webservices
* Service Oriented Architecture (SOA) and Microservices Architecture
* Service-Oriented Architecture (SOA)
* Benefit and Costs of SOA
* Successes and failures in implementing SOA
* Infrastructure and tools supporting SOA
* How Microservices Implement Web Service Concepts
* Continuous Deployment and Delivery (Continuous Integration, Continuous Build Process, etc.)
* Lightweight protocols
* Microservice Protocols and Standards
* HTTP, JMS, AMQP, Websockets, JSON, etc.
* Java-based frameworks (Spring Cloud)
* Decomposing a Monolith Application[+Practical]
* Developing independently deployable applications
* Organizing microservice applications around business capabilities
* Case study: Migrating a monolith application to three core microservices

**Microservices Architecture Core Concepts :- Advanced**

* Overall topology and architecture components
* Remote access protocols
* Protocol-aware heterogeneous interoperability
* Service components
* Service granularity
* Bounded context
* API layer
* Inter-service communication (choreography)
* Service orchestration
* Contract decoupling
* Accessing third-party systems
* Advantages and disadvantages
* Overall topology
* Service granularity
* Database scope
* Service deployment
* Advantages and disadvantages over Microservices
* Primary benefits
* Case studies and example architectures

**Day 8**

**Overview of Spring Cloud**

* Spring Cloud sub-projects: Config Server & Bus, Eureka, Ribbon, Feign, and Hystrix

**Centralized, versioned configuration management with Spring Cloud Config**

* Config server
* Accessing configuration file from git local repo
* Accessing configuration file from git central repo via web

**ZULL : working with API Gateway**

* Declarative REST clients with Feign

**Spring Cloud Ribbon**

* Introduction
* Service discovery integration
* Fault tolerance
* Configurable load-balancing rules
* Dependency management
* Ribbon configuration
* application.yml file
* Ribbon architecture
* Ribbon infrastructure
* Failure resiliency
* Ribbon client development
* Client configuration
* Ribbon client api
* Customizing the defaults
* Client property configuration
* Ribbon standards
* Ribbon integration with Eureka
* Caching ribbon configuration
* Configuring Hystrix thread pools
* Ribbon key management
* Load balancing best practices
* Ribbon best practices
* Fault tolerance configuration

**Day 9**

**Spring Cloud Eureka**

* Introduction
* Eureka Vs. AWS ELB
* Eureka Vs. Route 53
* Eureka integration in Netflix
* Eureka architecture
* Configuring Eureka
* Scaleability, robustness, reliability and availability factors
* Using memcached caching services
* Cache configuration
* Load-balancing caches
* Non-java services
* clients
* Configuring Regions
* Monitoring clients
* Monitoring servers
* Advanced monitoring using Servo

**Day 10**

**Spring Cloud Hystrix**

* What is Hystrix ?
* Hystrix purpose
* Features
* Latency and failure control
* Monitor cascading failures
* Rapid recovery
* Graceful degradation
* Real-time monitoring, Alerts and Operational control
* Application isolation
* Hystrix architecture
* Configuring app containers
* Dependency management
* Managing and monitoring network traffic
* Call wrapping
* Timeouts
* Thread pools
* Success measurement

**Spring Boot and Microservices – Final assessment**

* Practical lab assessment on Spring boot and microservices – Online shop application Use Case or as the participants may use their own scenario with at least 3 different use cases in the single application.
* Participants must use and implement all the concepts learned from day-1 till date.

**Sample Modules includes and responsibilities**

1. Login page with Spring security
2. Portal admin modules like adding products, managing orders, maintaining ware house etc.
3. Provide user login module to track individual orders and payments
4. Provide load balancing on product search module
5. Create each module as a separate service, and developing a web application which uses all the developed microservices.
6. Use messaging capabilities to communicate from service to service
7. Write test cases on various modules implemented so far