The perfect gift for readers and writers. **Give the gift of Medium**

Member-only story

Kafka + Springboot Bootcamp — From Zero to Production

Arvind Kumar Following 4 min read · Oct 29, 2025

110 4

> ***Duration:*** *15 Days (1 hour/day)*
>
> ***Focus:*** *Kafka Core + Spring Boot Integration + Real Production System*
>
> ***Target Audience:*** *Developers new to Kafka but familiar with Java & Spring Boot*

Apache Kafka isn't just a message broker — it's the **nervous system of modern distributed systems.**
From event-driven microservices to real-time analytics, Kafka sits at the center of systems that move data *continuously* instead of *occasionally*.

This **15-day Kafka Bootcamp** is designed to take a Java/Spring Boot developer from *zero* to *production-grade* Kafka integration — understanding the

internals, building fault-tolerant producers and consumers, handling retries, ensuring exactly-once delivery, and even understandoing the Kafka deployed to Cloud.

> *Full story for non-members* | *Grab My Microservices E-Book* | *Youtube* | *LinkedIn* | *Book a 1:1 Meeting*

Each day, we'll invest just **1 focused hour** — combining theory, guided demos, and hands-on mini assignments — to build up a solid Kafka-based system step by step.

Day 1: The World of Event Streaming

- Why Kafka? Evolution from message queues to event logs.

- Core concepts: topics, partitions, brokers, producers, consumers.
- Kafka architecture overview (ISR, leader-follower replication).

> Hands-on: *Set up Kafka locally (using Docker or other way)and create your first topic.*

Day 2: CLI Practice and Deep Dive into Kafka Internals. Kafdrop for kafka monitoring

- Using `kafka-topics.sh`, `kafka-console-producer`, and `kafka-console-consumer`.
- Exploring topic metadata, partition leaders, and offsets.
- Understanding message durability and replication.
- Install Kafdrop for Kafka monitoring

> Hands-on: *Produce and consume messages using CLI; simulate broker failure. Kafka monitoring with kafdrop*

Day 3: Spring Boot Integration — Producer Setup

- Setting up Kafka dependencies in Spring Boot.
- Producer configs: serializers, acks, retries, batching.
- Writing your first producer and publishing events programmatically.

> Hands-on: *Send* `OrderPlacedEvent` *using REST endpoint.*

Day 4: Spring Boot Integration — Consumer Setup

- Consumer groups and rebalancing explained.

- Deserialization and concurrent consumers.
- Manual vs auto offset commits.

> Hands-on: *Build an OrderConsumer that logs and persists data.*

Day 5: Error Handling + Retries + Dead Letter Topics

- Common failure patterns (serialization, transient errors, poison pills).
- Spring Kafka's `@RetryableTopic` and DLT setup.
- Logging and tracing failed messages.

> Hands-on: *Create a retry + DLT pipeline and verify with test messages.*

Day 6: Message Ordering and Keying Strategies

- Partitioning logic and message ordering guarantees.
- How keys influence message placement.
- Handling reordering in consumers.

> Hands-on: *Experiment with keyed vs keyless messages.*

Day 7: Idempotency, Transactions, and Exactly Once

- Producer idempotence (`enable.idempotence=true`).
- Kafka transactions and atomic writes to multiple topics.
- Exactly-once semantics — myth vs reality.

> Hands-on: *Build a payment processor with transactional producer.*

Day 8: Building Resilient Event-Driven Systems

- Event delivery patterns: fire-and-forget, synchronous, and async acknowledgment.
- Designing event contracts and versioning strategies.
- Real-world pitfalls: duplicate messages, network retries, and lag.

> Hands-on: *Add logging and validation layers around events.*

Day 9: Outbox Pattern

- Problem: dual writes and data consistency.
- Outbox table design, transactional publishing, and Debezium overview.
- Implementing the outbox pattern with Spring Boot + Kafka.

> Hands-on: *Implement outbox publisher for Order Service.*

Day 10: Saga Pattern and Event Choreography

- Coordinating distributed transactions with Kafka events.
- Saga orchestrator vs choreography models.
- Example: Order → Payment → Inventory → Notification.

> Hands-on: *Implement event-based Saga flow.*

Day 11: Kafka Streams Fundamentals

- Stream processing vs message consumption.
- Stateless transformations (`map`, `filter`, `branch`).

- Writing a Kafka Streams app inside Spring Boot.

> Hands-on: *Build order event transformer service.*

Day 12: Kafka Streams — Stateful Operations

- Aggregations, joins, and windowed computations.
- Understanding state stores and RocksDB.
- Reprocessing and materialized views.

> Hands-on: *Count orders per minute and expose via REST.*

Day 13: Schema Registry + Avro + Versioning

- Why schemas matter in event-driven systems.
- Using Confluent Schema Registry.
- Avro serialization and compatibility rules (backward, forward).

> Hands-on: *Produce and consume Avro messages.*

Day 14: Integration Testing with Embedded Kafka

- Unit testing producers and consumers.
- Integration testing with `@EmbeddedKafka`.
- Testcontainers for Kafka end-to-end flow.

> Hands-on: *Write integration tests for event pipeline.*

Day 15: Monitoring, Lag Metrics, and Cloud Deployment

- Understanding consumer lag and offsets.
- Kafka metrics via Micrometer and Actuator.
- Understanding Kafka available on AWS (known as MSK) and Confluent Cloud.
- Production tuning tips (acks, linger.ms, batch.size).

> Hands-on: *Deploy and verify your system's metrics dashboard.*

Final Project

Goal: Build a mini "Order Management System" consisting of:

- **Order Service** → produces events
- **Payment Service** → consumes + produces
- **Inventory Service** → consumes + updates stock
- Add retry/DLT handling, transaction safety, and test coverage.

Closing Note

By the end of this 15-day bootcamp, you won't just "know Kafka" —
You will understand *why* Kafka behaves the way it does under load, during failures, and across distributed systems.

They'll be able to:

- Build reliable producers and consumers in Spring Boot.

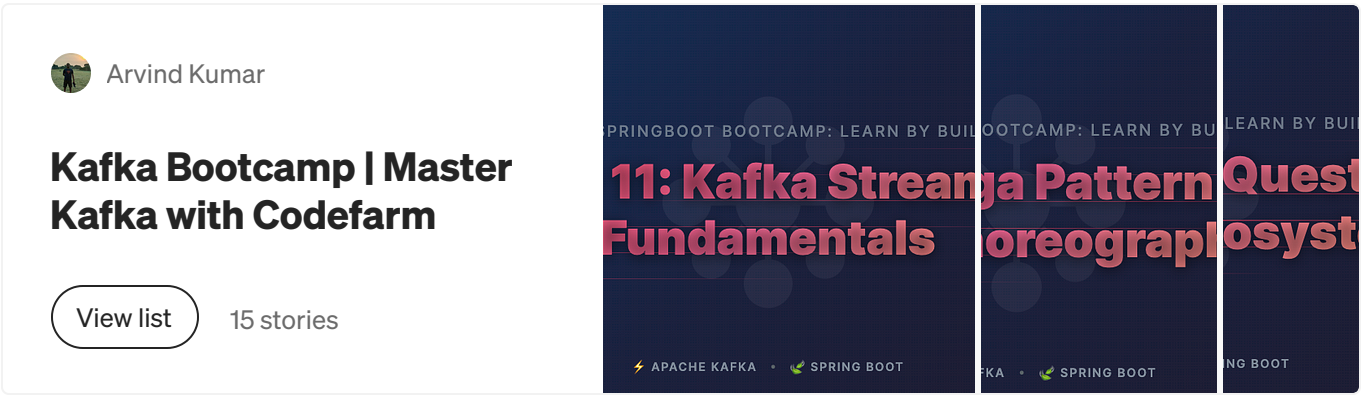
- Handle retries, DLTs, and transactional events with confidence.
- Test and monitor Kafka pipelines like a production engineer.
- Deploy and operate Kafka apps on cloud environments such as MSK or Confluent Cloud.

> *Need my help to do this bootcamp? Drop message on linkedin*

This lays the perfect foundation for **Part 2: Advanced Kafka Master class**, where we'll dive into **Kafka Connect, Schema Evolution, Performance Tuning, Multi-Cluster Replication, and Real-Time Analytics Pipelines** — the topics that turn an engineer into an architect.

======

Here is the list for all the stories around Kafka Boot Camp



Kafka | Apache Kafka | Kafka Producer | Kafka Streams | Kafka Connect

Written by Arvind Kumar

3.2K followers · 110 following

Following

Staff Engineer | System Design, Microservices, Java, SpringBoot, Kafka, DBs, AWS, GenAI | Teaching concepts via stories & characters | linkedin.com/in/codefarm0

Responses (4)

Pradeep Kasula

What are your thoughts?

Enamoradoc
Oct 29, 2025 (edited)

Awesome, glad you're doing this!

7 1 reply Reply

Engouyassa
Nov 11, 2025

Great undertaking. How is this bootcamp accessible?

4 1 reply Reply

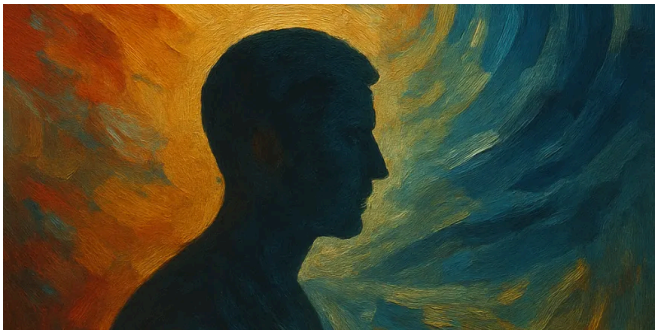
mohamad shahkhajeh
Nov 2, 2025

Have you considered how partitioning strategy might impact exactly-once delivery and consumer lag under load?

6 Reply

See all responses

More from Arvind Kumar

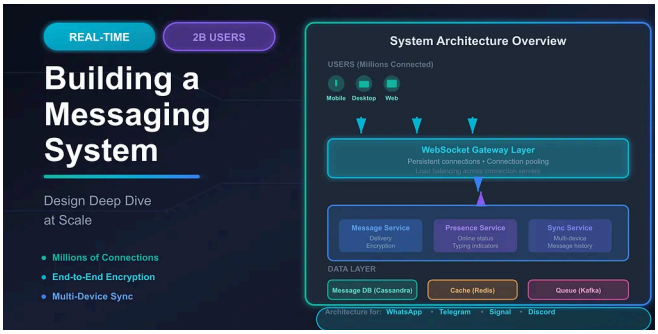


In FAUN.dev() 🐾 by Arvind Kumar

The Silent Earthquake in Tech: Why Every Engineer Must Rethink Thei...

Sharing my personal experience, thought process, and happenings around me.

Dec 13, 2025 482 20

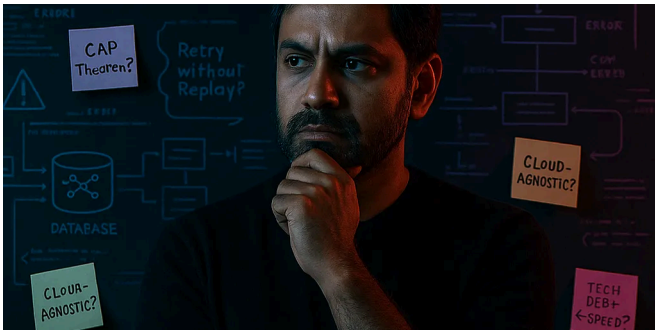


Arvind Kumar

Building a Real-Time Messaging System: Design Deep Dive

How do you deliver messages to 2 billion users in real-time? Let's design a messaging...

Dec 7, 2025 131 2



Arvind Kumar

12 Interview Questions That Separate Real Architects from...

"Staff Engineer or Just Staff Meeting Engineer?"

Jul 3, 2025 121 1

Arvind Kumar

Forward Proxy vs Reverse Proxy: The Deep Dive Every Engineer Mu...

There are few networking components as critical—and as misunderstood—as forwar...

Dec 4, 2025 188 2

See all from Arvind Kumar

Recommended from Medium

Gaddam.Naveen

If You Know These 12 Microservice Design Patterns, You're Interview-...

if you are not a medium member then Click here to read free

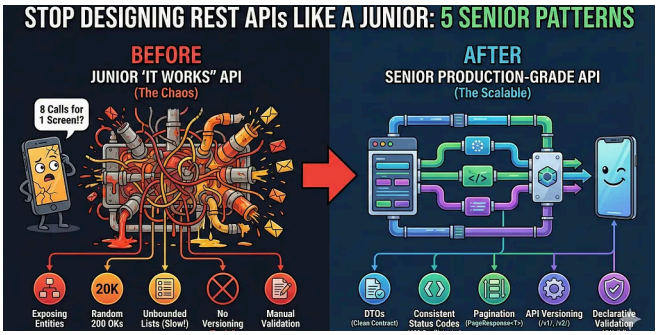
Dec 20, 2025 14 2

Arvind Kumar

Outbox Pattern- Day 9

Welcome to the Outbox Pattern!

4d ago 3

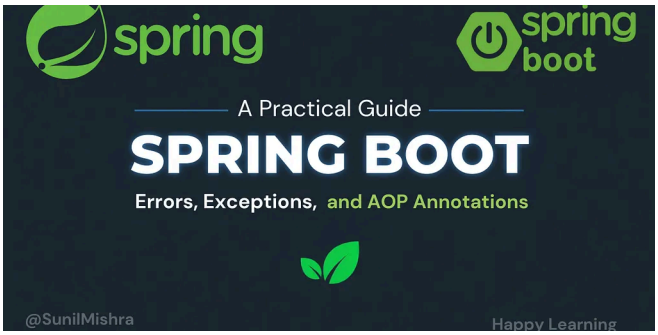


In Stackademic by Habibwahid

Stop Designing REST APIs Like a Junior: 5 Patterns Senior Engineer...

Your API works. But is it maintainable, scalable, and client-friendly? Let's understan...

Dec 10, 2025 178 1

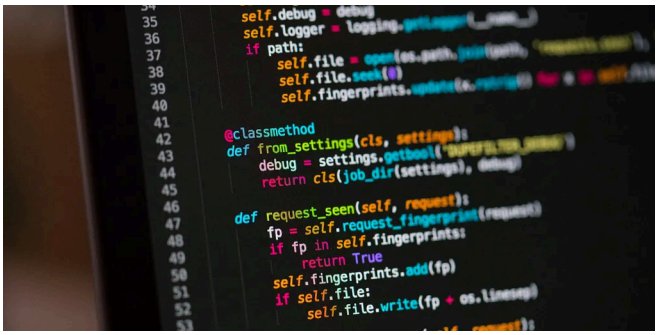


Code With Sunil | Code Smarter, not harder

Spring Boot Errors, Exceptions, and AOP Annotations—A Practical...

Learn how to centralize error handling, simplify exception management, and use AO...

4d ago 51

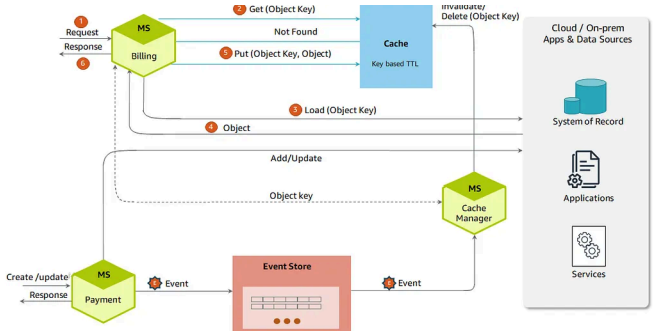


In CodeX by Riya Sharma

2 Ways to Make Spring Boot Fully Serverless in 2026

A practical guide to running Spring Boot without servers, ops headaches, or idle costs

5d ago 83



In DevOps.dev by StackStories

Caching Strategy for Microservices in Spring Boot (Production-Ready...

Caching in microservices is not optional at scale. Without a well-designed caching...

Dec 26, 2025 13 1

See more recommendations