

BEGINNER LEVEL | 40–45 HOURS

OBJECTIVE Learn microservices fundamentals, REST APIs, and containerization with Java (Spring Boot) and Python (Flask/FastAPI).

Module 1: Microservices Fundamentals (4 hrs)

- Monolith vs SOA vs Microservices
- Microservices advantages & challenges
- 12-Factor app principles
- Industry use cases (Netflix, Amazon, Uber)

Module 2: Java Microservices with Spring Boot (8 hrs)

- Spring Boot project setup (Maven/Gradle)
- REST controllers, DTOs, exception handling
- Data access with Spring Data JPA
- Testing with JUnit + Postman
- Swagger/OpenAPI docs

Module 3: Python Microservices with Flask & FastAPI (8 hrs)

- Flask basics: routes, request handling
- FastAPI async APIs with Pydantic models
- CRUD APIs with SQLAlchemy
- Dependency injection in FastAPI
- Auto-generated API docs (Swagger, Redoc)

Module 4: Database Integration (6 hrs)

- SQL databases: PostgreSQL/MySQL
- NoSQL databases: MongoDB basics
- ORM: Hibernate (Java), SQLAlchemy (Python)
- Database migrations (Flyway, Alembic)

Module 5: Containerization with Docker (7 hrs)

- Docker concepts: images, containers, volumes
- Dockerizing Java Spring Boot service
- Dockerizing Python FastAPI service
- Networking in Docker
- Multi-service setup with Docker Compose

BEGINNER PROJECTS (5)

1. Java Spring Boot CRUD service for Students.
2. Python FastAPI CRUD service for Inventory.
3. Dockerize both services & connect with Compose.
4. Build an API with Postgres backend + REST interface.
5. Create API documentation with Swagger/OpenAPI.



INTERMEDIATE LEVEL | 50 HOURS

OBJECTIVE Learn inter-service communication, messaging, discovery, monitoring, and API security.

Module 6: Inter-Service Communication (7 hrs)

- REST communication
- gRPC for microservices
- API Gateway: Kong, Nginx, Spring Cloud Gateway

Module 7: Messaging & Event-Driven Architecture (7 hrs)

- Synchronous vs asynchronous communication
- RabbitMQ producers/consumers in Java + Python
- Kafka basics: topics, partitions, producers, consumers
- Event sourcing vs CQRS

Module 8: Service Discovery & Configuration (7 hrs)

- Eureka service registry (Spring Cloud)
- Consul for Python microservices
- Centralized config with Spring Cloud Config Server
- Environment variables, secrets management

Module 9: Security in Microservices (7 hrs)

- Authentication vs Authorization
- JWT authentication flow
- Spring Security with OAuth2
- FastAPI JWT + OAuth2 integration
- Role-based access control (RBAC)

Module 10: Observability (7 hrs)

- Centralized logging with ELK stack
- Distributed tracing with Zipkin/Jaeger
- Metrics collection with Prometheus
- Visualization with Grafana

INTERMEDIATE PROJECTS (5)

6. Order + Payment microservices with API Gateway.
7. Event-driven email notification system (RabbitMQ/Kafka).
8. Secure microservices with JWT + OAuth2.
9. Centralized logging & monitoring dashboard (Grafana).
10. Service registry + config server for multi-service setup.



ADVANCED LEVEL | 60 HOURS

OBJECTIVE Master design patterns, CI/CD, Kubernetes, service mesh, and production best practices.

Module 11: Microservices Design Patterns (8 hrs)

- Saga pattern for distributed transactions
- Circuit breaker (Resilience4j, Hystrix)
- Strangler Fig & API Composition
- Bulkhead & Retry patterns

Module 12: CI/CD Pipelines for Microservices (8 hrs)

- CI/CD with Jenkins pipelines
- GitLab CI/CD for multi-service apps
- Docker build & push to registry
- Automated integration tests

Module 13: Cloud-Native Microservices (10 hrs)

- Kubernetes fundamentals (pods, services, deployments)
- Ingress controller setup
- ConfigMaps & Secrets
- Horizontal Pod Autoscaling (HPA)
- Deploying microservices on EKS/GKE/AKS

Module 14: Service Mesh & Advanced Security (8 hrs)

- Istio basics: sidecar proxy, ingress/egress
- Mutual TLS between microservices
- Policy enforcement with OPA Gatekeeper
- Zero-trust networking in microservices

Module 15: Resiliency & Production Best Practices (8 hrs)

- Canary & blue-green deployments
- API versioning strategies
- Observability with OpenTelemetry
- Handling failures gracefully
- Cost optimization & scaling strategies

ADVANCED PROJECTS (5)

11. E-commerce platform with Java + Python microservices (Products, Orders, Payments, Users).
12. Deploy complete microservices architecture on Kubernetes.
13. Implement Saga pattern with Java + Kafka for Order → Payment workflow.
14. Secure microservices with Istio mTLS + OPA policies.
15. CI/CD pipeline for microservices with Jenkins/GitLab.



CAPSTONE PROJECTS

Choose 2–3 projects | Approximately 20–30 hours

Banking	Multi-service system (Accounts, Transactions, Fraud Detection).
Healthcare	Patient record & appointment booking with event-driven architecture.
Retail	Inventory + order management with Kafka + real-time updates.
Education	Online course platform with Python recommendation engine + Java backend.
Streaming/Media	Netflix-style video platform with microservices + monitoring.

TOTAL PROJECTS | 15+

OBJECTIVE Beginner: 5 • Intermediate: 5 • Advanced: 5 • Capstone projects: 2–3

TOOLS & TECHNOLOGIES

◆ Java (Spring Boot)	◆ Python (Flask/FastAPI)
◆ Docker	◆ Kubernetes
◆ PostgreSQL / MySQL	◆ MongoDB
◆ RabbitMQ	◆ Kafka
◆ Jenkins	◆ GitLab CI/CD
◆ Prometheus	◆ Grafana
◆ ELK Stack	◆ Zipkin / Jaeger
◆ Eureka	◆ Consul
◆ Spring Cloud	◆ OAuth2 / JWT
◆ Istio	◆ OpenTelemetry
◆ OPA Gatekeeper	◆ AWS EKS / GKE / AKS