



สถาบันไอทีจีเนียส  
IT GENIUS INSTITUTE

COURSE ID

WDC-47

COURSE OUTLINE · รายละเอียดหลักสูตรอบรม

WEB DEVELOPMENT / JAVA

# Microservices with Spring Boot and Spring Cloud

สร้างระบบหลาย Service ที่ทนทานและติดตาม  
ได้ด้วย Spring Boot 4 และ Spring Cloud

● 18 ชั่วโมง

● 3 วัน

● พื้นฐาน Java และ Spring Boot

● ระดับ Intermediate



ระยะเวลา: 3 วัน

18 ชั่วโมง (3 วัน วันละ 6 ชม.) · 09:00-16:00 น.



รูปแบบ: Classroom / Hybrid

Onsite หรือ Online ผ่าน Microsoft Teams · Public / In-house



รายละเอียดหลักสูตร

[www.itgenius.co.th](http://www.itgenius.co.th)

เมื่อระบบโตขึ้นและมีหลายทีมช่วยกันพัฒนา แอปพลิเคชันค่อนข้างยาก Deploy ง่าย และฟังทั้งระบบเมื่อส่วนใดส่วนหนึ่งมีปัญหา สถาปัตยกรรม Microservices จึงถูกนำมาใช้เพื่อแยกระบบเป็นบริการย่อยที่พัฒนาและขยายได้อิสระ แต่ก็มาพร้อมโจทย์ใหม่ทั้งการตั้งค่า การค้นหา การรับมือเมื่อบริการปลายทางล่ม และการติดตามปัญหาที่วิ่งข้ามหลาย Service ซึ่ง Spring Boot และ Spring Cloud มีเครื่องมือรองรับไว้ครบ

หลักสูตรนี้ต่อยอดจากพื้นฐาน Spring Boot ไปสู่การสร้างระบบ Microservices ด้วย Spring Boot 4.1, Spring Cloud 2023.1 และ Java 21 LTS เริ่มจากออกแบบขอบเขตของแต่ละ Service สร้าง RESTful Service พร้อมเอกสาร API และ Health Check แล้วแยกเป็นหลาย Service ที่เรียกกันด้วย HTTP Service Client จากนั้นรวมศูนย์ค่าตั้งด้วย Config Server ค้นหาด้วย Eureka และ LoadBalancer ทั่ว Spring Cloud Gateway ทำ Fault Tolerance ด้วย Resilience4j และใช้ Spring Cloud Bus กับ RabbitMQ ปิดท้ายด้วย Distributed Tracing, Metrics, OAuth2 และการรันทั้งระบบด้วย Docker Compose (อบรม 3 วัน วันละ 6 ชั่วโมง รวม 18 ชั่วโมง ระดับ Intermediate)

เครื่องมือ

SB

Spring Boot 4.1

SC

Spring Cloud 2023.1

J

Java 21 LTS

R4j

Resilience4j

Dk

Docker

ราคาค่าอบรม · PUBLIC / IN-HOUSE

**9,500** บาท / ท่าน

ราคายังไม่รวมภาษีมูลค่าเพิ่ม (VAT 7%) · รวมเอกสารประกอบการอบรม Lab Guide ซอร์สโค้ดทุก Service ควบทุก Lab ไฟล์ docker-compose พร้อมใช้ และ Certificate · รับผิดชอบ In-house ขอใบเสนอราคาได้

สมัครผ่านเว็บไซต์ ลด 10%



เพิ่มเพื่อน  
LINE

☎ 02-570-8449

☎ 088-807-9770

LINE @itgenius

🌐 [www.itgenius.co.th](http://www.itgenius.co.th)



## 01 วัตถุประสงค์ Objectives

- เข้าใจข้อดี ข้อจำกัด และการแบ่งขอบเขตของระบบแบบ Microservices
- สร้าง RESTful Service ด้วย Spring Boot 4.1 พร้อม Validation, ProblemDetail, Versioning และเอกสาร OpenAPI
- ให้ Service เรียกกันด้วย HTTP Service Client และกระจายโหลดด้วย Spring Cloud LoadBalancer
- รวมศูนย์ค่าตั้งค่าด้วย Spring Cloud Config Server และกระจายการเปลี่ยนแปลงด้วย Spring Cloud Bus
- ค้นหา Service ด้วย Eureka และวาง API Gateway ด้วย Spring Cloud Gateway
- ทำ Fault Tolerance ด้วย Resilience4j ทั้ง Circuit Breaker, Retry และ Rate Limiter
- ติดตาม Request ข้าม Service ด้วย Micrometer Tracing และ Zipkin และเก็บ Metrics ด้วย Actuator
- ป้องกันระบบด้วย OAuth2 กับ JWT และรันทั้งระบบด้วย Docker Compose

## 02 กลุ่มเป้าหมาย Target Audience

- Java Developer ที่ใช้ Spring Boot อยู่แล้วและต้องการก้าวสู่ระบบ Microservices
- Backend Developer ที่ต้องแยกระบบเดิมออกเป็นหลาย Service
- Software Architect และ Tech Lead ที่ต้องวางโครงสร้างระบบแบบกระจาย
- ทีม DevOps และผู้ดูแลระบบที่ต้องดูแลและติดตามระบบ Microservices
- อาจารย์และนักศึกษาที่ต้องการเห็นการสร้างระบบ Microservices ด้วย Java ครบวงจร

## 03 จุดเด่นของหลักสูตร Highlights

- **ปรับปรุงใหม่ปี 2026** Spring Boot 4.1, Spring Cloud 2025.1 และ Java 25
- **ต่อยอดจาก Spring Boot** เน้นระบบหลาย Service ไม่ใช่แอปเดียว
- **รับมือเมื่อระบบล้ม** Circuit Breaker, Retry และ Fallback
- **มองเห็นทั้งระบบ** Distributed Tracing, Metrics และ Log
- **รันได้ทั้งระบบ** Docker Compose พร้อมเก็บ Kubernetes

#### 04 ความรู้พื้นฐาน Prerequisites

- เขียนภาษา Java แบบ OOP ได้ และเคยสร้างแอปด้วย Spring Boot มาบ้าง
- เข้าใจ RESTful API, HTTP Method และรูปแบบข้อมูล JSON
- ใช้ฐานข้อมูลเชิงสัมพันธ์ เช่น MySQL, PostgreSQL หรือ SQL Server ขั้นพื้นฐานได้
- โน้ตบุ๊กที่ติดตั้ง JDK 25, IDE และ Docker Desktop ได้ พร้อมสิทธิ์ติดตั้งโปรแกรม

#### 05 สิ่งที่ต้องเตรียมและเครื่องมือ Setup & Tools

| รายการ            | รายละเอียด                                                                          |
|-------------------|-------------------------------------------------------------------------------------|
| อุปกรณ์           | Notebook RAM 16 GB ขึ้นไป เชื่อมต่ออินเทอร์เน็ต ติดตั้ง Docker Desktop ได้          |
| ภาษาและ Framework | Java 25 LTS, Spring Boot 4.1, Spring Cloud 2025.1 และ Maven หรือ Gradle             |
| เครื่องมือพัฒนา   | IntelliJ IDEA Community หรือ VS Code, Spring Initializr, Git และ Postman หรือ Bruno |
| Infrastructure    | PostgreSQL, RabbitMQ, Zipkin, Prometheus, Grafana และ Keycloak uu Docker            |
| โปรเจกต์ตัวอย่าง  | ระบบแลกเปลี่ยนและแปลงสกุลเงินแบบหลาย Service สำหรับทุก Lab                          |

**หมายเหตุ:** ปรับปรุงใหม่ปี 2026 เทียบกับรุ่นเดิม: อัปเดตเป็น Spring Boot 4.1, Spring Cloud 2025.1 และ Java 25 LTS ซึ่งใช้ Jakarta EE แทน javax ย่อเนื้อหา RESTful พื้นฐานเหลือบทเดียวเพราะมีในหลักสูตร Spring Framework 7 and Spring Boot 4 Basic to Advanced แล้ว เปลี่ยนการเรียก Service จาก OpenFeign ซึ่งอยู่ในสถานะ Maintenance มาใช้ HTTP Service Client ใช้ Spring Cloud Gateway แบบ Server WebMVC เพิ่ม Resilience4j ครอบคลุม Retry และ Rate Limiter, ความสามารถ @Retryable ของ Spring Framework 7, OAuth2 กับ Keycloak, Metrics และ Docker Compose และจัดการรับ Zipkin บน Windows แบบเดิมมาใช้ Container เครื่องมือทั้งหมดเป็นโอเพนซอร์สหรือใช้ฟรี

#### 06 ภาพรวมการเรียนรู้ 3 วัน Big Picture

| วัน                           | หัวข้อหลัก                                                                                                                                                                           |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Day 1</b><br>Section 1-5   | ออกแบบระบบแบบ Microservices, สร้าง RESTful Service ด้วย Spring Boot 4.1, ทำเอกสาร API และ Health Check, แยกระบบเป็นหลาย Service และเรียกกันด้วย HTTP Service Client                  |
| <b>Day 2</b><br>Section 6-10  | รวมศูนย์ค่าตั้งค่าด้วย Config Server, ค้นหา Service ด้วย Eureka และ LoadBalancer, วาง API Gateway, ทำ Fault Tolerance ด้วย Resilience4j และกระจายการเปลี่ยนแปลงด้วย Spring Cloud Bus |
| <b>Day 3</b><br>Section 11-15 | ติดตาม Request ข้าม Service ด้วย Micrometer Tracing และ Zipkin, เก็บ Metrics, ป้องกับระบบด้วย OAuth2 และ JWT, รันทั้งระบบด้วย Docker Compose และปิดท้ายด้วย Capstone                 |



DAY  
1

## จาก RESTful Service สู่หลาย Service

เนื้อหาการอบรม · Section 1-5

รวม  
6 ชั่วโมง

### 1 Lab: รู้จัก Microservices Architecture

- Monolith เทียบกับ Microservices และกรณีที่ยังไม่ควรแยกระบบ
- แบ่งขอบเขต Service ตามโดเมนธุรกิจ และหลัก Database per Service
- ความท้าทายของระบบกระจาย และเครื่องมือที่ Spring Cloud มีให้
- Spring Boot 4.1, Spring Cloud 2025.1 และ Java 25 LTS ทำงานร่วมกันอย่างไร
- Lab: เตรียมโปรเจกต์ด้วย Spring Initializr และ Docker Compose สำหรับฐานข้อมูล

### 2 Lab: RESTful Service ด้วย Spring Boot 4.1

- กฎของ Controller, Request Mapping และ HTTP Status Code ที่ถูกต้อง
- Validation และการตอบ Error แบบมาตรฐานด้วย ProblemDetail
- เชื่อมฐานข้อมูลด้วย Spring Data JPA และความสัมพันธ์ระหว่าง Entity
- API Versioning ที่ Spring Framework 7 รองรับในตัว
- Lab: Service จัดการข้อมูลผู้ใช้ที่พร้อมให้ Service อื่นเรียกใช้

### 3 Lab: เอกสาร API และ Actuator

- สร้างเอกสาร OpenAPI และหน้า Swagger UI ด้วย springdoc-openapi
- Richardson Maturity Model และแนวปฏิบัติที่ดีในการออกแบบ API
- Spring Boot Actuator: Health, Info และ Metrics Endpoint
- Liveness และ Readiness Probe สำหรับระบบที่รันใน Container
- Lab: เปิดเอกสาร API และ Health Check ให้ทุก Service

### 4 Lab: แยกระบบเป็น Currency Exchange และ Conversion

- ออกแบบ Service แลกเปลี่ยนสกุลเงินและ Service แปลงสกุลเงิน
- กำหนด Port, URL และชื่อ Service ให้เป็นมาตรฐานเดียวกัน
- แยกฐานข้อมูลของแต่ละ Service และเตรียมข้อมูลตั้งต้น
- แสดง Instance และ Port ที่ตอบกลับเพื่อใช้ทดสอบการกระจายโหลด
- Lab: รันหลาย Instance ของ Service เดียวกันพร้อมกัน

### 5 Lab: เรียก Service อื่นด้วย HTTP Service Client

- RestClient และการเรียก Service ปลายทางแบบ Synchronous
- ประกาศ Client เป็น Interface ด้วย @HttpExchange และจัดกลุ่มด้วย @ImportHttpServices
- ตั้งค่า Base URL และ Timeout แยกตามกลุ่มของ Client
- เทียบกับ Spring Cloud OpenFeign ที่อยู่ในสถานะ Maintenance
- Lab: Service แปลงสกุลเงินเรียกอัตราแลกเปลี่ยนจากอีก Service



DAY  
2

## Spring Cloud สำหรับระบบกระจาย

เนื้อหาการอบรม · Section 6-10

รวม  
6 ชั่วโมง

### 6 Lab: Spring Cloud Config Server

- ปัญหาของค่าตั้งค่าที่กระจายอยู่ในแต่ละ Service
- ตั้ง Config Server ที่อ่านค่าจาก Git Repository
- แยกค่าตามชื่อ Service และ Profile เช่น dev และ prod
- เชื่อม Service ด้วย spring.config.import และเข้ารหัสค่าที่เป็นความลับ
- Lab: ย้ายค่าตั้งค่าของ Limits Service และ Currency Service ไปไว้ที่ Config Server

### 7 Lab: Eureka และ Spring Cloud LoadBalancer

- ทำไมต้องมี Naming Server และหลักการของ Service Registry
- ตั้ง Eureka Server และให้ทุก Service ลงทะเบียนเมื่อเริ่มทำงาน
- เรียก Service ด้วยชื่อแทน URL ผ่าน RestClient ที่ใช้ LoadBalancer
- กระจายค่าขอไปหลาย Instance และสังเกตเมื่อ Instance หายไป
- Lab: เพิ่มและลด Instance โดยไม่ต้องแก้โค้ด

### 8 Lab: API Gateway ด้วย Spring Cloud Gateway

- บทบาทของ API Gateway ในระบบ Microservices
- Spring Cloud Gateway แบบ Server WebMVC และ Server WebFlux ต่างกันอย่างไร
- กำหนด Route, Predicate และ Filter ไปยังแต่ละ Service ผ่าน Eureka
- Global Filter สำหรับ Logging และการจำกัดจำนวนคำขอ
- Lab: ให้ทุกคำขอจากภายนอกผ่าน Gateway จุดเดียว

### 9 Lab: Fault Tolerance ด้วย Resilience4j

- Cascading Failure และเหตุที่ระบบกระจายต้องออกแบบรับความล้มเหลว
- Circuit Breaker ผ่าน Spring Cloud CircuitBreaker พร้อม Fallback
- Retry, Rate Limiter, Bulkhead และ Time Limiter ของ Resilience4j
- @Retryable และ @ConcurrencyLimit ที่มาใหม่ใน Spring Framework 7
- Lab: ปิด Service ปลายทางแล้วดูระบบตอบด้วย Fallback แทนการล่ม

### 10 Lab: Spring Cloud Bus และ RabbitMQ

- ปัญหาการรีเฟรชค่าตั้งค่าในทุก Instance พร้อมกัน
- รับ RabbitMQ ด้วย Docker และเชื่อมทุก Service ผ่าน Spring Cloud Bus
- ส่งรีเฟรชค่าทั้งระบบผ่าน Endpoint busrefresh ครั้งเดียว
- แนวคิด Event-driven และการสื่อสารแบบ Asynchronous เบื้องต้น
- Lab: เปลี่ยนค่าใน Git แล้วให้ทุก Instance ใช้ค่าใหม่โดยไม่ Restart



DAY  
3

## Observability, ความปลอดภัย และการ Deploy

เนื้อหาการอบรม · Section 11-15

รวม  
6 ชั่วโมง

### 11 Lab: Distributed Tracing

- Trace และ Span ช่วยหาจุดที่ช้าเมื่อคำขอวิ่งข้ามหลาย Service
- Micrometer Tracing และ Observation API ใน Spring Boot 4
- ส่ง Trace ไป Zipkin และทางเลือก OpenTelemetry ผ่าน OTLP
- ใส่ Trace ID ใน Log เพื่อเชื่อมโยงปัญหาข้าม Service
- Lab: ติดตามคำขอตั้งแต่ Gateway จนถึงฐานข้อมูลบนหน้า Zipkin

### 12 Lab: Metrics และ Dashboard

- เก็บ Metrics ด้วย Micrometer และเปิด Endpoint สำหรับ Prometheus
- Metrics สำคัญของ Microservices: Latency, Error Rate และ Throughput
- สร้าง Dashboard ด้วย Grafana และตั้งการแจ้งเตือนเบื้องต้น
- ดูสถานะ Circuit Breaker ผ่าน Actuator และ Metrics
- Lab: Dashboard ภาพรวมสุขภาพของทุก Service

### 13 Lab: ความปลอดภัยของ Microservices

- OAuth2 และ OpenID Connect สำหรับระบบหลาย Service
- ใช้ Keycloak เป็น Authorization Server และออก JWT
- ตั้งแต่ละ Service เป็น Resource Server ด้วย Spring Security
- ตรวจสอบสิทธิ์ที่ Gateway และส่งต่อ Token ไปยัง Service ปลายทาง
- Lab: ปกป้อง API ทั้งระบบด้วย JWT และกำหนดสิทธิ์ตามบทบาท

### 14 Lab: Container และการ Deploy

- สร้าง Image ด้วย Dockerfile หรือ Cloud Native Buildpacks
- รับทั้งระบบด้วย Docker Compose พร้อมลำดับการเริ่มและ Health Check
- Virtual Threads ของ Java 25 กับการรองรับคำขอพร้อมกันจำนวนมาก
- ภาพรวมการย้ายขึ้น Kubernetes และส่วนที่ทดแทน Eureka กับ Config Server ได้

### 15 Workshop: Capstone ระบบ Microservices

- เพิ่ม Service ใหม่เข้าระบบพร้อม Config, Discovery และ Gateway Route
- ทดสอบความทนทานด้วยการปิด Service และเพิ่มความหน่วงโดยตั้งใจ
- ไล่หาสาเหตุปัญหาจาก Trace, Metrics และ Log
- นำเสนอผลงาน รีวิวร่วมกัน และสรุป Checklist ก่อนขึ้น Production

## 07 เรียนจบแล้วได้อะไร Outcomes

- ออกแบบ **Microservices** ได้ ขอบเขต Service และฐานข้อมูลแยกกัน
- ใช้ **Spring Cloud** คล่อง Config Server, Eureka และ Gateway
- สื่อสารข้าม **Service** ได้ HTTP Service Client และ LoadBalancer
- ระบบทนต่อความล้มเหลว Resilience4j, Retry และ Fallback
- มองเห็นปัญหาทั้งระบบ Tracing, Metrics และ Dashboard
- ปลดล็อกและพร้อม **Deploy** OAuth2, JWT และ Docker Compose

## 08 สิ่งที่คุณเรียนจะได้รับ Deliverables

- เอกสารประกอบการอบรมฉบับสมบูรณ์ครบทั้ง 3 วัน และ Lab Guide ที่ละเอียด
- ซอร์สโค้ดทุก Service ครบทุก Lab รวมถึง Config Repository ตัวอย่าง
- ไฟล์ docker-compose สำหรับ PostgreSQL, RabbitMQ, Zipkin, Prometheus, Grafana และ Keycloak
- ประกาศนียบัตรจบหลักสูตร (Certificate of Completion) และช่องทางสอบถามวิทยากรหลังจบหลักสูตร

## 09 ข้อมูลผู้สอน Instructor

**อ.สามิตร โทยม** ผู้สอนและนักพัฒนาที่ใช้ Java และ Spring Boot พัฒนาระบบ Backend และระบบที่แยกเป็นหลาย Service ในงานจริง จุดเด่นคือการสอนจากระบบตัวอย่างชุดเดียวที่ค่อย ๆ ขยายจาก Service แรกจนเป็นระบบกระจายเต็มรูปแบบ อธิบายให้เห็นปัญหาก่อนแนะนำเครื่องมือที่แก้ปัญหานั้น และพาผู้เรียนทดลองทำให้ระบบล้มเพื่อดูการรับมือจริง เพื่อให้กลับไปออกแบบและดูแลระบบ Microservices ของทีมได้อย่างมั่นใจ

ราคาค่าอบรม 9,500 บาท/ท่าน · สนใจ IN-HOUSE ขอใบเสนอราคา

**9,500 บาท** / ท่าน (ยังไม่รวม VAT 7%)

สมัครผ่านเว็บไซต์ ลด 10%

☎ 02-570-8449

📞 @itgenius

☎ 088-807-9770

✉ contact@itgenius.co.th



เพิ่มเพื่อน  
LINE