



สถาบันไอทีจีเนียส
IT GENIUS INSTITUTE

COURSE ID

WDC-92

COURSE OUTLINE · รายละเอียดหลักสูตรอบรม

WEB DEVELOPMENT / NODE.JS

Building Microservices with NestJS

แบ่งระบบเป็นบริการย่อยด้วย NestJS 12 พร้อม Broker, gRPC และ Docker

18 ชั่วโมง

3 วัน

พื้นฐาน NestJS และ TypeScript

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

Microservices คือการแบ่งระบบใหญ่ออกเป็นบริการเล็กตามหน้าที่ทางธุรกิจ เช่น ผู้ใช้ สินค้า คำสั่งซื้อ และการชำระเงิน แต่ละบริการพัฒนา Deploy และขยายแยกกันได้ แลกกับงานที่ซับซ้อนขึ้นทั้งการสื่อสารระหว่างบริการ ความสอดคล้องของข้อมูล และการติดตามปัญหา NestJS มีแพ็คเกจ @nestjs/microservices ที่รองรับ Transport หลายแบบด้วยรูปแบบการเขียนเดียวกัน จึงเหมาะกับทีมที่ต้องการวางระบบแบบนี้บน Node.js

หลักสูตรนี้ต่อยอดจาก Basic NestJS และ Advanced NestJS โดยใช้ NestJS 12 รุ่นปัจจุบัน เริ่มจากออกแบบขอบเขตของแต่ละบริการ สร้าง Microservice และ API Gateway สื่อสารผ่าน TCP, RabbitMQ, NATS และ gRPC แยกฐานข้อมูลของแต่ละบริการ จัดการ Transaction ข้ามระบบด้วย Saga และ Outbox จากนั้นทำให้ระบบทนต่อความล้มเหลว ติดตามได้ด้วย Log และ Tracing แล้วปิดท้ายด้วยการ Deploy ระบบร้านค้าทั้งชุดด้วย Docker Compose (อบรม 3 วัน วันละ 6 ชั่วโมง รวม 18 ชั่วโมง ระดับ Intermediate)

เครื่องมือ

N NestJS 12

Rb RabbitMQ

NT NATS

gR gRPC

Dk Docker Compose

ราคาค่าอบรม · PUBLIC / IN-HOUSE

10,000 บาท / ท่าน

ราคายังไม่รวมภาษีมูลค่าเพิ่ม (VAT 7%) · รวมเอกสารประกอบการอบรม Lab Guide ซอร์สโค้ดตัวอย่างครบทุก 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

01 วัตถุประสงค์ Objectives

- อธิบายข้อดีและข้อแลกเปลี่ยนของ Microservices และแบ่งบริการตามขอบเขตธุรกิจได้
- สร้าง Microservice และ Hybrid Application ด้วย NestJS 12 ได้
- เลือกใช้รูปแบบ Request-Response และ Event ได้เหมาะสมกับงาน
- สื่อสารระหว่างบริการผ่าน TCP, RabbitMQ, NATS และ gRPC ได้
- ออกแบบฐานข้อมูลแยกตามบริการและจัดการ Transaction ข้ามระบบด้วย Saga ได้
- ทำให้ระบบทนต่อความล้มเหลว และติดตามปัญหาด้วย Health Check, Log และ Tracing ได้
- Deploy ระบบหลายบริการด้วย Docker Compose ได้

02 กลุ่มเป้าหมาย Target Audience

- Backend Developer ที่ใช้ NestJS อยู่แล้วและต้องการแบ่งระบบเป็นบริการย่อย
- ผู้ที่เรียน Basic NestJS หรือ Advanced NestJS แล้วต้องการต่อยอด
- Software Architect และ Tech Lead ที่วางโครงสร้างระบบขนาดกลางถึงใหญ่
- DevOps และผู้ดูแลระบบที่ต้อง Deploy และดูแลระบบหลายบริการ

03 จุดเด่นของหลักสูตร Highlights

- **NestJS 12 รุ่นปัจจุบัน** Monorepo และ Hybrid App
- **ครบทุก Transport หลัก** TCP, RabbitMQ, NATS และ gRPC
- **ข้อมูลข้ามบริการ** Saga และ Outbox Pattern
- **ระบบที่ทนต่อปัญหา** Retry, Circuit Breaker และ Tracing
- **จบด้วย Workshop** ร้านค้าหลายบริการบน Docker

04 ความรู้พื้นฐาน Prerequisites

- เขียน NestJS ได้ ทั้ง Module, Controller, Provider และ Dependency Injection
- เขียน TypeScript และใช้ async กับ await ได้
- เคยสร้าง REST API และใช้ฐานข้อมูลมาบ้าง
- ใช้คำสั่ง Docker เบื้องต้นได้ และมีโน้ตบุ๊กที่รัน Docker Desktop ได้ (แนะนำ RAM 16 GB)

05 สิ่งที่ต้องเตรียมและเครื่องมือ Setup & Tools

รายการ	รายละเอียด
อุปกรณ์	Notebook เชื่อมต่ออินเทอร์เน็ต พร้อมสเปกติดตั้งโปรแกรม แนะนำ RAM 16 GB
Runtime และเครื่องมือ	Node.js รุ่น LTS, Nest CLI, VS Code และ Docker Desktop
Framework และไลบรารี	NestJS 12, @nestjs/microservices, @nestjs/terminus และ @grpc/grpc-js
Broker และฐานข้อมูล	RabbitMQ, NATS, Redis และ PostgreSQL uu Docker
ติดตามระบบ	OpenTelemetry และ Jaeger uu Docker

หมายเหตุ: เครื่องมือทั้งหมดเป็นโอเพนซอร์สหรือใช้ฟรีและรันบนเครื่องของผู้เรียน หลักสูตรนี้ปรับปรุงใหม่ปี 2026 จากฉบับเดิมที่สอน TCP Transport และ Deploy ด้วย PM2 ฉบับนี้ย้ายมาใช้ NestJS 12 ที่เป็น ESM บน Node.js รุ่น LTS วางโปรเจกต์แบบ Monorepo เพิ่ม API Gateway, RabbitMQ, NATS, gRPC, ฐานข้อมูลแยกตามบริการ, Saga และ Outbox, Resilience, Health Check และ Distributed Tracing และเปลี่ยนการ Deploy หลักเป็น Docker Compose โดยกล่าวถึง PM2 เป็นทางเลือกสำหรับเซิร์ฟเวอร์เดี่ยว Kafka สรุปเป็นภาพรวมเพราะใช้ทรัพยากรเครื่องสูง หลักสูตรนี้ต่อจาก Basic NestJS และ Advanced NestJS ซึ่งสอนเรื่อง ORM, JWT และ OpenAPI ไว้แล้ว เนื้อหาอิงเวอร์ชันล่าสุด ณ วันอบรม

06 ภาพรวมการเรียนรู้ 3 วัน Big Picture

วัน	หัวข้อหลัก
Day 1 Section 1-5	เข้าใจแนวคิด Microservices และการแบ่ง Service ตามขอบเขตธุรกิจ, เตรียม Monorepo ของ NestJS 12, สร้าง Microservice แรกด้วย TCP, ใช้ Message Pattern กับ Event Pattern และทำ API Gateway
Day 2 Section 6-10	สื่อสารผ่าน Message Broker ด้วย RabbitMQ และ NATS, เรียกกันแบบมี Contract ด้วย gRPC, แยกฐานข้อมูลของแต่ละ Service, จัดการ Transaction ข้ามระบบด้วย Saga และ Outbox
Day 3 Section 11-15	รับมือความล้มเหลวด้วย Timeout, Retry และ Circuit Breaker, ยืนยันตัวตนข้าม Service, ทำ Health Check, Log และ Tracing แล้ว Deploy ทั้งระบบด้วย Docker Compose ในโปรเจกต์ร้านค้า



DAY
1

แนวคิด Microservices, Transport แรก และ API Gateway

เนื้อหาการอบรม · Section 1-5

รวม
6 ชั่วโมง

1 Microservices คืออะไร และเมื่อไรควรใช้

- เปรียบเทียบ Monolith, Modular Monolith และ Microservices
- ข้อดีและข้อแลกเปลี่ยนที่ทีมต้องรับ
- แบ่งบริการตามขอบเขตธุรกิจด้วยแนวคิด Bounded Context
- สื่อสารแบบ Synchronous และ Asynchronous ต่างกันอย่างไร
- Workshop: ออกแบบบริการของระบบร้านค้าออนไลน์

2 Lab: เตรียมโปรเจกต์แบบ Monorepo

- ทบทวน NestJS 12 ที่ เป็น ESM และต้องใช้ Node.js 20 ขึ้นไป
- สร้าง Monorepo ด้วย Nest CLI และเพิ่ม Application หลายตัว
- แยกโค้ดที่ใช้ร่วมกันเป็น Library เช่น DTO และค่าคงที่
- ตั้งค่าแต่ละบริการด้วย @nestjs/config และไฟล์ .env
- Lab: โครงโปรเจกต์ร้านค้าที่มีบริการสินค้าและคำสั่งซื้อ

3 Lab: Microservice แรกด้วย TCP

- ติดตั้ง @nestjs/microservices และสร้างด้วย createMicroservice
- รับคำขอด้วย @MessagePattern และส่งค่ากลับ
- เรียกบริการอื่นด้วย ClientProxy และ send
- ลงทะเบียน Client ด้วย ClientsModule.registerAsync
- Lab: บริการสินค้าที่ตอบข้อมูลให้บริการคำสั่งซื้อ

4 Message Pattern และ Event Pattern

- Request-Response กับ Event-based ต่างกันอย่างไร
- ส่ง Event ด้วย emit และรับด้วย @EventPattern
- อ่าน Context และ Payload ด้วย @Ctx และ @Payload
- จัดการ Error ด้วย RpcException และ Exception Filter
- Lab: แจ้งเหตุการณ์สร้างคำสั่งซื้อให้หลายบริการ

5 Lab: API Gateway และ Hybrid Application

- บทบาทของ API Gateway ในระบบหลายบริการ
- สร้าง Gateway ที่รับ HTTP แล้วส่งต่อไปยังบริการภายใน
- Hybrid Application ด้วย connectMicroservice
- ตรวจสอบข้อมูลที่ Gateway และแปลง Error ให้เป็น HTTP Status
- Lab: REST API ของร้านค้าที่เรียกบริการภายในทั้งหมด



DAY
2

Message Broker, gRPC และข้อมูลข้ามบริการ

เนื้อหาการอบรม · Section 6-10

รวม
6 ชั่วโมง

6 Lab: สื่อสารผ่าน RabbitMQ

- Message Broker ช่วยลดการผูกกันระหว่างบริการอย่างไร
- รับ RabbitMQ บน Docker และใช้ Management UI
- ตั้งค่า Transport.RMQ ทั้งฝั่ง Server และ Client
- Queue แบบ Durable, การ Ack และการส่งซ้ำเมื่อประมวลผลไม่สำเร็จ
- Lab: ส่งงานออกใบเสร็จผ่าน Queue

7 NATS, Redis และภาพรวม Kafka

- ใช้ NATS สำหรับข้อความความเร็วสูงและ Queue Group
- ใช้ Redis เป็น Transport แบบ Pub/Sub
- ภาพรวม Kafka สำหรับ Event Stream ปริมาณมาก
- เปรียบเทียบ Transport ให้เหมาะกับงาน
- Lab: เปลี่ยน Transport ของบริการแจ้งเตือนเป็น NATS

8 Lab: gRPC และ Protocol Buffers

- gRPC คืออะไร และเหมาะกับการเรียกภายในแบบใด
- เขียนไฟล์ .proto เพื่อกำหนด Service และ Message
- ตั้งค่า Transport.GRPC และใช้ @GrpcMethod
- เรียก gRPC จาก Gateway และจัดการ Status Code ของ gRPC
- Lab: บริการตรวจสอบสต็อกสินค้าแบบ gRPC

9 ฐานข้อมูลแยกตามบริการ

- หลัก Database per Service และข้อควรระวัง
- รับ PostgreSQL แยกของแต่ละบริการด้วย Docker
- ทำสำเนาข้อมูลที่ต้องใช้ร่วมกันผ่าน Event
- Idempotency เพื่อรับข้อความซ้ำได้อย่างปลอดภัย
- Lab: บริการสินค้าและคำสั่งซื้อที่มีฐานข้อมูลของตัวเอง

10 Workshop: Saga และ Outbox Pattern

- ทำไม Transaction ข้ามบริการจึงทำแบบเดิมไม่ได้
- Saga แบบ Choreography และแบบ Orchestration
- เขียน Compensating Action เมื่อขั้นตอนใดล้มเหลว
- Outbox Pattern เพื่อบันทึกข้อมูลและส่ง Event ให้ตรงกัน
- Workshop: ขั้นตอนสั่งซื้อ ตัดสต็อก และชำระเงินแบบ Saga



DAY
3

ความทนทาน ความปลอดภัย การติดตาม และ Deploy

เนื้อหาการอบรม · Section 11-15

รวม
6 ชั่วโมง

11 Lab: Resilience ของระบบหลายบริการ

- ตั้ง Timeout ให้การเรียกบริการด้วย RxJS
- Retry แบบเว้นระยะและเมื่อไรไม่ควร Retry
- แนวคิด Circuit Breaker และ Fallback
- Graceful Shutdown ด้วย enableShutdownHooks
- Lab: Gateway ที่ยังตอบได้เมื่อบริการปลายทางล่ม

12 ความปลอดภัยระหว่างบริการ

- ยืนยันตัวตนที่ Gateway ด้วย JWT แล้วส่งข้อมูลผู้ใช้ต่อ
- ตรวจสอบสิทธิ์ในบริการภายในด้วย Guard
- แยกเครือข่ายภายในและไม่เปิดพอร์ตบริการสู่ภายนอก
- เก็บ Secret อย่างปลอดภัยและแนวคิด TLS ระหว่างบริการ
- Lab: ป้องกันบริการคำสั่งซื้อให้รับเฉพาะคำขอที่ผ่าน Gateway

13 Lab: Health Check, Log และ Tracing

- Health Check ของบริการ ฐานข้อมูล และ Broker ด้วย @nestjs/terminus
- Structured Log พร้อม Correlation ID ข้ามบริการ
- Distributed Tracing ด้วย OpenTelemetry
- ดูเส้นทางของคำขอใน Jaeger
- Lab: ตามหาบริการที่ทำให้คำสั่งซื้อช้า

14 Lab: Test และ Deploy ด้วย Docker Compose

- Unit Test ของ Handler ด้วย Vitest และการจำลอง ClientProxy
- เขียน Dockerfile แบบ Multi-stage ให้แต่ละบริการ
- รวมทุกบริการ Broker และฐานข้อมูลใน docker-compose.yml
- ขยายจำนวน Instance และกระจายงานผ่าน Queue
- PM2 สำหรับเซิร์ฟเวอร์เดี่ยว และภาพรวมการขยายสู่ Kubernetes

15 Workshop: Capstone ร้านค้าแบบ Microservices

- รวม Gateway, บริการผู้ใช้ สินค้า คำสั่งซื้อ และแจ้งเตือน
- เชื่อมบริการด้วย RabbitMQ, NATS และ gRPC ตามที่ออกแบบ
- ทดสอบกรณีบริการล่มและข้อความซ้ำ
- ติดตามการทำงานทั้งระบบผ่าน Health Check และ Tracing
- Workshop: นำเสนอสถาปัตยกรรมและแนวทางต่อยอดในองค์กร

07 เรียนจบแล้วได้อะไร Outcomes

- ออกแบบบริการได้ Bounded Context และ Gateway
- เลือก Transport เป็น TCP, RabbitMQ, NATS และ gRPC
- ดูแลข้อมูลข้ามบริการ Saga, Outbox และ Idempotency
- ทำระบบให้ทนทาน Timeout, Retry และ Circuit Breaker
- ติดตามปัญหาได้ Health Check, Log และ Tracing
- Deploy ทั้งระบบได้ Docker Compose หลายบริการ

08 สิ่งที่คุณเรียนจะได้รับ Deliverables

- เอกสารประกอบการอบรมครบทั้ง 3 วัน และ Lab Guide ที่ละเอียด
- ซอร์สโค้ด Monorepo ของระบบร้านค้าแบบ Microservices ครบทุก Lab
- ไฟล์ docker-compose สำหรับบริการ Broker ฐานข้อมูล และเครื่องมือติดตามระบบ
- ประกาศนียบัตรจบหลักสูตร (Certificate of Completion) และช่องทางสอบถามวิทยากรหลังจบหลักสูตร

09 ข้อมูลผู้สอน Instructor

อ.สนิทวงศ์ กมลภากรณ์ ผู้สอนและนักพัฒนาที่ใช้ NestJS ออกแบบและดูแลระบบหลังบ้านหลายบริการในงานจริง สอนจากระบบร้านค้าชุดเดียวที่ค่อย ๆ แยกเป็นบริการย่อย อธิบายเหตุผลของการเลือก Transport และรูปแบบการสื่อสารแต่ละแบบ พร้อมจำลองปัญหาที่เกิดขึ้นในระบบจริง เช่น บริการล่มหรือข้อความซ้ำ เพื่อให้ผู้เรียนนำแนวทางไปปรับใช้กับระบบของตัวเองได้อย่างมั่นใจ

ราคาค่าอบรม 10,000 บาท/ท่าน · สนใจ IN-HOUSE ขอใบเสนอราคา

10,000 บาท / ท่าน (ยังไม่รวม VAT 7%)

สมัครผ่านเว็บไซต์ ลด 10%

☎ 02-570-8449

📞 @itgenius

☎ 088-807-9770

✉ contact@itgenius.co.th



เพิ่มเพื่อน
LINE