

หลักสูตรผู้นำด้านคลาวด์ (The Cloud Leader Course)

จากความรู้ศูนย์สู่การนำทีมวิศวกรคลาวด์ — จัดทำเพื่อ Fedly Bonneau

13 สิงหาคม 2569

หลักสูตรผู้นำด้านคลาวด์ (The Cloud Leader Course)

จากความรู้ศูนย์ สู่การนำทีมวิศวกรคลาวด์ได้อย่างคล่องแคล่ว

หลักสูตรนี้ทำงานอย่างไร

คุณไม่ได้กำลังฝึกเพื่อเป็นวิศวกรคลาวด์ คุณกำลังฝึกเพื่อนำวิศวกรคลาวด์ต่างหาก นั่นคือ ตามทันทุกบทสนทนาในห้องประชุม ถามคำถามที่สำคัญจริง ๆ ตัดสินใจได้ดีในเรื่องเงิน ความเสี่ยง และคน และได้รับความเคารพจากผู้เชี่ยวชาญโดยไม่ต้องสร้างทำเป็นผู้เชี่ยวชาญเสียเอง นี่คือทักษะอีกแบบหนึ่งที่เรียนรู้ได้ทั้งหมด — และเป็นทักษะที่หลักสูตรนี้จะสอนคุณ

หลักสูตรแบ่งเป็นสามช่วง **ช่วงที่ 1 (สัปดาห์ 1–8): พูดภาษาเดียวกับทีม** คุณจะได้เรียนรู้ว่าคลาวด์คืออะไรกันแน่ และคำศัพท์ของทุกองค์ประกอบหลัก เพื่อให้การประชุมเล็กฟังดูเหมือนเสียงรบกวน **ช่วงที่ 2 (สัปดาห์ 9–16): คิดแบบคนในห้อง** สถาปัตยกรรม ความปลอดภัย และเงิน — สามบทสนทนาที่ทุกทีมคลาวด์พูดคุยกันทุกสัปดาห์ และเป็นสามเวทีที่ผู้นำจะสร้างคุณค่าได้จริง หรือไม่ก็ถูกมองข้ามไปเลย **ช่วงที่ 3 (เดือน 5–24): นำทีม** ใบรับรอง (certification) การจ้างงาน การนำประชุมตัดสินใจ และเส้นทางสองปีอย่างตรงไปตรงมาสู่การขึ้นห้อยด้ายกับวิศวกรอาวุโสได้อย่างเต็มภาคภูมิ

กติกาตลอดหลักสูตร: เรียนวันละหนึ่งชั่วโมง หกวันต่อสัปดาห์ — ความสม่ำเสมอชนะความหึกโหม ทุกโมดูลจบด้วย **Fluency Drill** (แบบฝึกความคล่อง — ประโยคที่คุณต้องพูดออกเสียงจนเป็นธรรมชาติ) และ **Milestone** (หมุดหมาย — หลักฐานว่าคุณพร้อมไปต่อ) อย่าข้ามสองอย่างนี้ เพราะการอ่านอย่างเดียวสร้างได้แค่ความคุ้นเคย ไม่ใช่ความคล่อง และตั้งแต่สัปดาห์ที่ 1 ให้เริ่มจด **Decision Journal**

(สมุดบันทึกการตัดสินใจ): ทุกครั้งที่เรียนรู้แนวคิดใหม่ ให้เขียนหนึ่งประโยคว่ามันส่งผลต่อเงิน ความเสี่ยง หรือคนอย่างไร — เพราะการแปลความแบบนี้แหละคืองานทั้งหมดของคุณ

ช่วงที่ 0 — ชั้นล่างสุด (เริ่มที่นี่หากเทคโนโลยีเป็นเรื่องใหม่สำหรับคุณ)

สองโมดูลนี้เขียนขึ้นสำหรับผู้อ่านที่ไม่เคยทำงานด้านเทคโนโลยีมาก่อนเลย เนื้อหาเหมือนกับโมดูล 1 และ 2 ของหลักสูตรแบบอินเทอร์แอคทีฟที่ b4lcilc.org หากคุณรู้อยู่แล้วว่า web server และ database คืออะไร จะอ่านผ่าน ๆ ก็ได้ แต่ถ้ายังไม่รู้ นี่คือหน้าที่สำคัญที่สุดของหนังสือเล่มนี้ — เพราะทุกสิ่งหลังจากนี้ตั้งอยู่บนพื้นฐานสองโมดูลนี้ทั้งหมด

โมดูล A: เว็บทำงานอย่างไร — เกิดอะไรขึ้นเมื่อคุณพิมพ์ที่อยู่เว็บไซต์

แนวคิดใหญ่: หน้าเว็บไม่ใช่รูปภาพที่ถูกยิงส่งมายังโทรศัพท์ของคุณ แต่คือชุดคำสั่งที่เขียนเป็นลายลักษณ์อักษร ส่งมาจากคอมพิวเตอร์เครื่องหนึ่งที่อยู่แสนไกล แล้ว browser (เบราว์เซอร์) ของคุณเองเป็นผู้ทำตามคำสั่งเหล่านั้นเพื่อประกอบหน้าขึ้นบนจอ เรียนรู้ทุกขั้นตอนระหว่าง “พิมพ์ที่อยู่” กับ “เห็นหน้าเว็บ” ให้ขึ้นใจ แล้วการประชุมเรื่องเทคโนโลยีทุกครั้งตลอดชีวิตการทำงานที่เหลือของคุณจะตามได้ง่ายขึ้นทันที

การเดินทางในหกขั้นตอน ที่คุณควรพูดออกเสียงได้เอง:

1. คุณพิมพ์ชื่อ — b4lcilc.org — ลงใน **browser** (เบราว์เซอร์: Chrome, Safari, Edge, Firefox)
2. **DNS** สมุดโทรศัพท์ของอินเทอร์เน็ต แปลงชื่อนั้นให้เป็น **IP address** — ที่อยู่แบบตัวเลขของคอมพิวเตอร์เครื่องหนึ่ง เช่น 142.250.66.14
3. browser ของคุณ — ฝ่าย **client** (ไคลเอนต์ — ผู้ที่ถาม) — ส่ง **HTTP request** (คำขอ) ไปยังคอมพิวเตอร์ที่อยู่นั้น: “GET — ขอหน้านี้หน่อย”
4. คำขอนั้นเดินทางเป็น **packet** (แพ็กเก็ต — ชิ้นข้อมูลเล็ก ๆ) ตามกติกาของ **TCP/IP** ระบบท่อประปาของอินเทอร์เน็ต ซึ่งใส่หมายเลขกำกับทุก packet ยืนยันว่าส่งถึง และส่งซ้ำทุกชิ้นที่หายระหว่างทาง
5. คอมพิวเตอร์เครื่องไกลนั้น — ฝ่าย **server** (เซิร์ฟเวอร์ — ผู้ที่ตอบ) — ส่ง **response** (คำตอบ) กลับมา: คำสั่งสำหรับสร้างหน้าเว็บ หรือไม่ก็ข้อแก้ตัวเป็นตัวเลขอย่าง **404** (“ไม่มีหน้านี้”)
6. browser ของคุณทำตามคำสั่งเหล่านั้นแล้วประกอบหน้าขึ้นมา คำสั่งเหล่านี้เขียนด้วยสามภาษา: **HTML** (โครงสร้าง — หัวข้อ ย่อหน้า ปุ่ม), **CSS** (หน้าตา — สี ขนาด การจัดวาง) และ

JavaScript (พฤติกรรม — จะเกิดอะไรขึ้นเมื่อคุณคลิก)

Client-side กับ server-side — กรอบความคิดที่ใช้จัดระเบียบทุกสิ่ง: งานที่ทำบนอุปกรณ์ของคุณภายใน browser คือ *client-side* (ฝั่ง **front end**) ส่วนงานที่ทำบนคอมพิวเตอร์เครื่องไกลคือ *server-side* (ฝั่ง **back end**) หลักง่ายๆ ๆ ที่ควรท่องให้ขึ้นใจ: **อะไรก็ตามที่เกี่ยวข้องกับความไว้วางใจหรือความจำ อยู่ฝั่ง server; อะไรก็ตามที่เกี่ยวข้องความรู้สึกสั้นไหลรวดเร็ว อยู่ฝั่ง client** ตราจรรยาบรรณ บันทึกรายการ ตัดเงินจากบัตร: server วาดหน้าจอ ตอบสนองทันทีที่คุณพิมพ์: client และระหว่างสองฝั่งนี้มักมี **proxy** (หรือกซ) คั่นกลาง — server ตัวกลางที่คอยกรอง คัดกรอง หรือกระจายการจราจรข้อมูล คำว่า “load balancer” และ “CDN” ในโมดูลถัด ๆ ไปก็คือ proxy ที่มีความเชี่ยวชาญเฉพาะทางนั่นเอง

ทำไม DNS จึงสำคัญกับผู้นำ: DNS ล่มในแบบที่พิเศษไม่เหมือนใคร — เมื่อมันพัง ทุกเว็บไซต์จะ “หายไป” พร้อมกันทั้งที่ server ทุกเครื่องยังทำงานอยู่ครบ สมุดโทรศัพท์ต่างหากที่ล่ม ไม่ใช่โทรศัพท์ เมื่อวิศวกรพูดว่า “อาจจะเป็นที่ DNS” นี่แหละคือสิ่งที่เขาหมายถึง และเหตุการณ์อินเทอร์เน็ตล่มครั้งโด่งดังที่สุดหลายครั้งก็เกิดจากเรื่องนี้เป๊ะ ๆ

และเจ้ารูปแม่กุญแจ: HTTPS ก็คือ HTTP ที่เพิ่มการเข้ารหัส — บทสนทนาทั้งหมดถูกทำให้อ่านไม่ออกสำหรับใครก็ตามที่ดักฟังอยู่ระหว่างทาง และนั่นคือสิ่งที่รูปแม่กุญแจในแถบที่อยู่รับรองให้ กติกายุคปัจจุบันคือ: ทุกเว็บไซต์ ตลอดเวลา

คำศัพท์:

คำศัพท์	ความหมาย
Web browser	โปรแกรมที่ไปดึงหน้าเว็บมาแล้ววาดขึ้นจอ: Chrome, Safari, Edge, Firefox
URL	ที่อยู่เต็มที่คุณพิมพ์: โพรโตคอล + ชื่อคอมพิวเตอร์ + เอกสารที่คุณต้องการ
IP address	ที่อยู่แบบตัวเลขของคอมพิวเตอร์บนเครือข่าย เช่น 142.250.66.14
DNS	สมุดโทรศัพท์ของอินเทอร์เน็ต: แปลงชื่อ (google.com) ให้เป็น IP address
HTTP / HTTPS	กติกาถาม-ตอบที่ browser กับ server ใช้คุยกัน ตัว S เพิ่มการเข้ารหัส (รูปแม่กุญแจ)
TCP/IP	ระบบท่อประปาของอินเทอร์เน็ต: ลำเลียง packet ที่มีหมายเลขกำกับระหว่างที่อยู่ต่าง ๆ อย่างเชื่อถือได้

คำศัพท์	ความหมาย
Client / server	ผู้ถามและผู้ตอบ — เป็นความสัมพันธ์ ไม่ใช่ประเภทของเครื่อง
Client-side (front end)	งานที่ทำใน browser ของผู้ใช้: การวาดหน้าจอ การตอบสนองทันที
Server-side (back end)	งานที่ทำบนเครื่องไกล: ความไว้วางใจ กฎกติกา บันทึกถาวร
Proxy	server คนกลางที่จัดการการจราจรข้อมูลแทนฝ่ายใดฝ่ายหนึ่ง
HTML / CSS / JavaScript	สามภาษาของหน้าเว็บ: โครงสร้าง / หน้าตา / พฤติกรรม

Fluency drill: พุดการเดินทางหกชั้นตอนออกเสียงคนเดียว ซ้ำจนใช้เวลาไม่ถึงสามสิบวินาทีและไม่มีจังหวะนึกคำติดขัดเลย จากนั้นเล่าให้อีกคนหนึ่งฟัง

Milestone: คุณตอบคำถามนี้ได้บนโต๊ะอาหารเย็นโดยไม่ต้องเตรียมตัว: “จริง ๆ แล้วเกิดอะไรขึ้นตอนที่ฉันพิมพ์ชื่อเว็บไซต์ลงในโทรศัพท์?” — ตอบเป็นเรื่องเล่า ไม่ใช่รายการศัพท์เทคนิค

โมดูล B: เว็บไซต์ถูกสร้างและโฮสต์อย่างไร — จากคอมพิวเตอร์หนึ่งเครื่องสู่คลาวด์

แนวคิดใหญ่: เว็บไซต์จริงทุกแห่งสร้างขึ้นเป็นสามชั้น — front end ที่ผู้คนสัมผัส ชั้นกลางที่กฎกติกาขององค์กรทำงาน และชั้นข้อมูลที่ไม่มีวันหลงลืม — และตลอดยี่สิบห้าปีที่ผ่านมา เกือบทั้งหมดถูกประกอบขึ้นจากเครื่องมือหนึ่งในสองตระกูลอันโด่งดัง เรียนรู้สามชั้นและสองตระกูลนี้ แล้วการประชุมกับ vendor จะเปลี่ยนจากภาษาต่างดาวกลายเป็นเมนูอาหารให้เลือกร

ร้านอาหาร ซึ่งคุณจะใช้เปรียบเทียบไปตลอดหลักสูตรนี้: front end (UI) คือห้องอาหาร — ทุกสิ่งที่ลูกค้ามองเห็น สร้างด้วย HTML, CSS และ JavaScript **ชั้นกลาง (business logic)** คือครัว — โค้ดบน server ที่บังคับใช้กฎของร้าน: “นักเรียนห้ามลงทะเลเป็นซ้ำสองครั้ง” “คำสั่งซื้อเกินวงเงินต้องขออนุมัติ” ส่วน **ชั้นข้อมูล (data layer)** คือห้องเก็บวัตถุดิบและสมุดสูตรอาหาร — นั่นคือ **database** (ฐานข้อมูล) ที่เก็บทุกสิ่งที่ห้ามลืมเด็ดขาด วงการเรียกกรุปทรงนี้ว่า **N-tier architecture** ทำไมต้องแบ่งชั้น? ก็เพื่อให้ปรับปรุงแต่ละชั้นได้โดยไม่ต้องรื้อสร้างชั้นอื่นใหม่ — และเมื่อวิศวกรพูดว่า “ขอขवादอยู่ที่ data layer” เขากำลังยื่นแผนที่ฉบับนี้ให้คุณ พร้อมวงกลมชี้จุดไว้แล้ว

เครื่องกับโปรแกรม: server คือคอมพิวเตอร์ของจริง — กล่องโลหะไร้จอที่วางเรียงในแร็คร่วมกับอีกหลายร้อยเครื่องใน **data center** (ศูนย์ข้อมูล) ถูกสร้างมาให้ทำงานทุกวินาทีตลอดทั้งปี บนเครื่องนั้นมี **โปรแกรม**ที่เรียกว่า **web server** เช่นกัน ทำหน้าที่ดำเนินบทสนทนา HTTP รู้จักสองตัวที่ดังที่สุดไว้ให้ขึ้นใจ: **IIS** (ของ Microsoft รันบน Windows) และ **Apache** (ฟรี รันบน Linux เป็นส่วนใหญ่) แยกคำให้ชัด: *browser เป็นฝ่ายถาม web server เป็นฝ่ายตอบ* Apache คือ **web server** — ฝ่ายผู้ตอบ — ไม่ใช่ browser ส่วน **web hosting** (เว็บโฮสติ้ง) คือการเช่าพื้นที่บน server ของคนอื่น เพื่อให้เครื่องของเขาตอบคำขอในนามของคุณ

database และ SQL แบบค่อยเป็นค่อยไป: database คือโปรแกรมที่เก็บบันทึกข้อมูลและตอบคำถามเกี่ยวกับข้อมูลเหล่านั้น — เปรียบเหมือนเสมียนจัดแฟ้มผู้คลังความเป็นระเบียบและไม่เคยหลับ **SQL** (อ่านว่า “ซีเควเอล”) ก็เพียงแค่**ภาษาสำหรับถามคำถามกับ database** เท่านั้นเอง *query* (ควอรี) คือคำถามหนึ่งข้อ ลองอ่านบรรทัดนี้ — คุณอ่านออกอยู่แล้ว: `SELECT name FROM students WHERE village = 'Mae La'` — จากตาราง *students* ขอชื่อของทุกคนที่มาจากแม่หละ มีสองยี่ห้อครองตลาด: **SQL Server** (ของ Microsoft ต้องซื้อ license) และ **MySQL** (ฟรี เป็น open source) ทั้งคู่พูดภาษา SQL เหมือนกัน ประเด็นสำหรับผู้นำคือ: **database คือที่ที่องค์กรดำรงอยู่จริง ๆ** — หน้าเว็บพังก็สร้างใหม่ได้ แต่ถ้าข้อมูลทะเบียนนักเรียนหายไป โรงเรียนก็ไม่เหลืออะไรเลย นี่คือเหตุผลที่ database ได้รับการปกป้องทางวิศวกรรมเข้มงวดที่สุดในทั้งอาคาร

สองตระกูล:

หน้าที่ในระบบ	ตระกูล Microsoft	ตระกูล open source LAMP
ระบบปฏิบัติการ	Windows Server	Linux
Web server (ตอบ HTTP)	IIS	Apache
ภาษาสำหรับ business logic	C# บน .NET	PHP
Database	SQL Server	MySQL
โมเดลค่าใช้จ่าย	ค่า license ผูกกับ vendor รายเดียว	ฟรีและ open source; คุณจ่ายค่าคนแทน
ถิ่นดั้งเดิม	ธนาคาร หน่วยงานรัฐ องค์กรใหญ่	เว็บเปิดกว้าง — WordPress อย่างเดียวก็รันหนึ่งในสามของเว็บไซต์ทั่วโลก

Open source (โอเพนซอร์ส) หมายถึงสูตรของซอฟต์แวร์นั้นเปิดเผยต่อสาธารณะ — ใครก็ตรวจสอบปรับปรุง และนำไปใช้ได้โดยไม่ต้องจ่ายค่า license นี่คือเหตุผลที่องค์กรเล็ก ๆ ทุนน้อยสามารถสร้างเว็บไซต์ระดับโลกได้เสมอ ด้วยราคาแค่ค่าเช่าเครื่องหนึ่งเครื่อง

ถนนสู่คลาวด์ — ประโยคที่หนังสือทั้งเล่มนี้ตั้งอยู่บนนั้น: เป็นเวลาหลายทศวรรษ ทั้งหมดนี้รันแบบ **on-premises** (ออนเพรมิส): ในห้อง server ที่องค์กรเป็นเจ้าของ ซื้อมาเพื่อวันทำงานหนักที่สุดเท่าที่จินตนาการได้ แล้วปล่อยว่างเปล่าในวันอื่น ๆ ทั้งปี พร้อมจ่ายค่าแอร์เย็น ๆ ให้มันเองตลอดเวลา จากนั้น Amazon และ Microsoft ก็เสนอให้เช่ากำลังประมวลผลจากโกดังของพวกเขาเองเป็นรายชั่วโมง ตัว stack ไม่ได้เปลี่ยนเลย — Linux, Apache, IIS, SQL Server ตัวเดิมนั้นแหละที่รันอยู่บนคลาวด์ทุกวันนี้ **สิ่งที่เปลี่ยนคือเครื่องไปตั้งอยู่ที่ไหน และใครเป็นเจ้าของ** จำประโยคนี้ไว้ให้มัน: โมดูล 1 ของช่วงที่ 1 เปิดฉากตรงจุดนี้พอดี

คำศัพท์:

คำศัพท์	ความหมาย
Server / data center	เครื่องที่เปิดทำงานตลอดเวลา และอาคารที่เต็มไปด้วยแร็คของเครื่องเหล่านั้น
Web server (โปรแกรม)	ซอฟต์แวร์ที่ตอบ HTTP request: IIS (Microsoft), Apache (open source)
Web hosting	การเช่าพื้นที่บน server ของคนอื่นสำหรับเว็บไซต์ของคุณ
N-tier architecture	การสร้างเป็นชั้น: front end (UI), ชั้นกลาง (logic), ชั้นข้อมูล (data layer)
Database	โปรแกรมที่เก็บบันทึกข้อมูลและตอบคำถาม — ที่ที่องค์กรดำรงอยู่จริง
SQL / query	ภาษาที่ใกล้เคียงภาษาอังกฤษสำหรับถามคำถามกับ database; query คือคำถามหนึ่งข้อ
SQL Server / MySQL	database แบบมี license ของ Microsoft / ตัวโปรดของชาว open source ทั้งคู่พูดภาษา SQL
Microsoft stack	Windows Server + IIS + C#/.NET + SQL Server
LAMP stack	Linux + Apache + MySQL + PHP — เป็น open source ทั้งหมด
Open source	สูตรที่เปิดเผยต่อสาธารณะ: ตรวจสอบ ปรับปรุง และใช้งานได้ฟรี

คำศัพท์	ความหมาย
On-premises	เครื่องที่คุณเป็นเจ้าของ ในห้องที่คุณเป็นเจ้าของ — โลกยุคก่อนคลาวด์

Fluency drill: แจกแจง LAMP บนนิ้วมือสี่นิ้ว พูดออกเสียง ติดต่อกันสามวัน จากนั้นอธิบายเรื่องราวอาหาร — ห้องอาหาร ครั้ว ห้องเก็บวัตถุดิบ — ให้คนที่ไม่เคยได้ยินคำว่า server ฟัง

Milestone: ผู้รับเหมาคนหนึ่งพูดว่า: “ระบบของคุณเป็น Microsoft on-prem รุ่นเก่า — IIS กับ SQL Server ส่วนบริษัทเราเป็นสาย LAMP และเราจะย้ายคุณขึ้นคลาวด์” คุณแปลได้ทุกคำให้บอร์ดบริหารฟัง — และคุณรู้ว่าจุดเสี่ยงที่ต้องชักให้ลึกคือการย้าย database

ช่วงที่ 1 — พูดภาษาเดียวกับทีม (สัปดาห์ 1–8)

โมดูล 1 (สัปดาห์ 1–2): คลาวด์คืออะไรกันแน่

แนวคิดใหญ่: คลาวด์คือ คอมพิวเตอร์ของคนอื่น ที่เราเช่าเป็นรายชั่วโมง และบริหารจัดการด้วยซอฟต์แวร์ ก่อนยุคคลาวด์ บริษัทต้องซื้อ server (เซิร์ฟเวอร์ — เครื่องคอมพิวเตอร์แม่ข่าย) มาตั้งไว้ในห้อง แล้วจ้างคนมาดูแล — ซ้ำ แพง และไม่ยืดหยุ่น AWS, Microsoft Azure และ Google Cloud จึงสร้างโกดังที่มีคอมพิวเตอร์นับล้านเครื่อง (data center — ศูนย์ข้อมูล) แล้วเปิดให้ใครก็ได้เช่าใช้เป็นส่วน ๆ คิดเงินเป็นรายวินาที จากที่ไหนก็ได้ ผ่านหน้าเว็บหรือโค้ดเพียงบรรทัดเดียว แค่นั้นเอง ทุกอย่างที่เหลือในหลักสูตรนี้คือรายละเอียดที่ต่อยอดจากแนวคิดเดี๋ยวนี

ทำความรู้จักสามผู้ให้บริการรายใหญ่:

ผู้ให้บริการ	คืออะไร
AWS (Amazon Web Services)	ธุรกิจคลาวด์ของ Amazon และผู้ให้บริการคลาวด์รายใหญ่ที่สุดในโลก (ราว 30% ของตลาดโลก) เริ่มต้นในปี 2549 (ค.ศ. 2006) เมื่อ Amazon เปิดให้เช่าระบบคอมพิวเตอร์ที่สร้างไว้ใช้กับร้านค้าออนไลน์ของตัวเอง ปัจจุบันมีบริการมากกว่า 200 อย่าง — server, storage, database, AI และอื่น ๆ — คิดค่าเช่าเป็นรายวินาที เมื่อหลักสูตรนี้พูดถึง “คลาวด์” AWS คือตัวอย่างหลักที่ใช้ และ AWS ได้เปิด

Thailand Region (กรุงเทพฯ) ของตัวเองเมื่อ
มกราคม 2568

Microsoft Azure

คลาวด์ของ Microsoft อันดับ 2 ของโลก และ
แข็งแกร่งที่สุดในองค์กรขนาดใหญ่ เพราะเชื่อมต่อ
เข้ากับเครื่องมือของ Microsoft ที่บริษัทต่าง ๆ ใช้อยู่
แล้วได้อย่างเป็นธรรมชาติ (Windows, Office,
Active Directory) ขณะนี้กำลังก่อสร้าง datacenter
region แห่งแรกในประเทศไทย

Google Cloud (GCP)

คลาวด์ของ Google อันดับ 3 — แข็งแกร่งที่สุดด้าน
การวิเคราะห์ข้อมูลและเครื่องมือ AI ได้ประกาศ
ลงทุน 1 พันล้านดอลลาร์สหรัฐในการสร้าง data
center ในไทยและ cloud region ที่กรุงเทพฯ

บัญชี AWS ฟรีของคุณ — สมัครให้เรียบร้อยในสัปดาห์ที่ 1 ไปที่ <https://aws.amazon.com/free> แล้ว
คลิก “Create a Free Account” (หน้าสมัครโดยตรงคือ https://signin.aws.amazon.com/signup?request_type=register) คุณจะต้องใช้อีเมล เบอร์โทรศัพท์ และบัตรเครดิต/เดบิตเพื่อยืนยันตัวตน —
Free Tier ให้โควตาบริการพื้นฐานฟรีรายเดือน (รวมถึง server EC2 ขนาดเล็ก 750 ชั่วโมง/เดือนในปี
แรกของคุณ) และแบบฝึกหัดทั้งหมดในหลักสูตรนี้อยู่ในโควตานี้ สองนิสัยความปลอดภัยที่ต้องมีตั้งแต่วันแรก: เปิดใช้ MFA (โมดูล 6 จะอธิบาย) และตั้ง **billing alert** (การแจ้งเตือนค่าใช้จ่าย) ไว้ที่ 5
ดอลลาร์ (โมดูล 7 จะสอนวิธี) เพื่อไม่ให้คุณมีวันถูกบิลจู่โจมแบบไม่รู้ตัว หลังสมัครเสร็จ คุณจะล็อกอิน
ได้ที่ <https://console.aws.amazon.com> — คำว่า “console” ก็คือหน้าเว็บควบคุมของ AWS นั่นเอง

ทำไมบริษัทถึงใช้คลาวด์: ความเร็ว (ได้ server ใหม่ใน 60 วินาที แทนที่จะรอ 6 สัปดาห์), ความยืดหยุ่น
(เช่า server 100 เครื่องสำหรับวันเซลใหญ่ แล้วคืน 90 เครื่องในวันถัดไป), ไม่มีต้นทุนก้อนแรก (เป็นค่า
ใช้จ่ายดำเนินงานแทนที่จะเป็นเงินลงทุน) และการเข้าถึงทั่วโลก (วางแอปของคุณไว้ใกล้ลูกค้าทั้งใน
กรุงเทพฯ โตเกียว และแฟรงก์เฟิร์ต โดยไม่ต้องสร้างอะไรเองเลย)

สามชั้นของบริการ — โมเดลความคิดที่มีประโยชน์ที่สุดในโลกคลาวด์:

ชั้นบริการ	สิ่งที่คุณเช่า	เปรียบเทียบกับครัว	ตัวอย่าง
IaaS (Infrastructure as a Service)	คอมพิวเตอร์เปล่า พื้นที่ จัดเก็บ เครือข่าย — คุณ ดูแลทุกอย่างบนนั้นเอง	เช่าครัวเปล่า: คุณต้อง พาเชฟ สูตรอาหาร และ วัตถุดิบมาเอง	Amazon EC2, Azure Virtual Machines

ชั้นบริการ	สิ่งที่คุณเช่า	เปรียบเทียบกับครัว	ตัวอย่าง
PaaS (Platform as a Service)	แพลตฟอร์มที่มีคนดูแลให้ — คุณนำมาแค่แอปพลิเคชันของคุณ	เช่าครัวพร้อมพนักงาน: คุณนำมาแค่สูตรอาหาร	AWS Elastic Beanstalk, Azure App Service
SaaS (Software as a Service)	ซอฟต์แวร์สำเร็จรูป	สั่งอาหารจากร้าน	Gmail, Salesforce, Canva

Region และ Availability Zone: **Region** (รีเจียน — กลุ่มศูนย์ข้อมูลตามพื้นที่ภูมิศาสตร์) คือกลุ่ม data center ในพื้นที่หนึ่ง (ประเทศไทยมี AWS Region ของตัวเองตั้งแต่มกราคม 2568 อยู่ในและรอบกรุงเทพฯ) ส่วน **Availability Zone (AZ)** คือ data center หนึ่งแห่งหรือมากกว่าที่แยกอิสระจากกันภายใน region เดียวกัน แอปที่สำคัญจริง ๆ จะรันอย่างน้อยสอง AZ เพื่อที่ว่าถ้าอาคารหนึ่งล่ม ระบบก็ยังไม่ดับ เมื่อวิศวกรพูดว่า “เรา multi-AZ” เขาหมายความว่า “ต่อให้ data center หนึ่งแห่งไฟไหม้ เราก็ยังให้บริการได้”

คำศัพท์:

คำศัพท์	ความหมาย
Cloud provider / hyperscaler	บริษัทที่เป็นเจ้าของ data center ขนาดมหึมาและให้เช่าพลังประมวลผลผ่านอินเทอร์เน็ต (AWS, Azure, Google Cloud) คำว่า “hyperscaler” หมายถึงกลุ่มรายใหญ่ที่สุดไม่กี่ราย ที่สร้างระบบให้ขยายได้แทบไร้ขีดจำกัด
Data center	โกดังที่มีระบบรักษาความปลอดภัย เต็มไปด้วย server นับพันเครื่อง พร้อมระบบไฟฟ้าและความเย็นระดับอุตสาหกรรม — สถานที่ทางกายภาพที่ “คลาวด์” อาศัยอยู่จริง ๆ
Region	กลุ่ม data center ตามพื้นที่ภูมิศาสตร์ที่เปิดให้เลือกเป็นหนึ่งตำแหน่งที่ตั้ง เช่น “Asia Pacific (Bangkok)” คุณเป็นคนเลือกว่าระบบของคุณจะรันอยู่ region ไหน
Availability Zone (AZ)	data center หนึ่งแห่งหรือมากกว่าที่แยกอิสระจากกันภายใน region โดยมีระบบไฟฟ้าและเครือข่ายเป็นของตัวเอง การรันในสอง AZ หมายความว่าอาคารหนึ่งพังได้โดยคุณยังออนไลน์อยู่

คำศัพท์	ความหมาย
On-premises (“on-prem”)	วิธีแบบเก่า: server ที่บริษัทของคุณเป็นเจ้าของ ตั้งอยู่ในอาคารของคุณเอง — ขั้วตรงข้ามของคลาวด์
Migration	โปรเจกต์การย้ายระบบจาก on-prem ขึ้นสู่คลาวด์
Workload	แอปพลิเคชันหรือระบบใด ๆ ที่รันอยู่ — “workload ระบบเงินเดือน” “workload เว็บไซต์” เป็นคำสารพัดประโยชน์: ครอบคลุมได้ทุกอย่าง
Provision	การสร้าง/ตั้งค่าทรัพยากรคลาวด์ (server หนึ่งเครื่อง database หนึ่งตัว) “provision a server” = เนรมิตเครื่องขึ้นมาหนึ่งเครื่อง
Scale up / scale out	การรองรับความต้องการที่เพิ่มขึ้น โดยขยายเครื่องให้ใหญ่ขึ้น (up) หรือเพิ่มจำนวนเครื่อง (out) — คลาวด์ถนัดแบบ out
Latency	ความหน่วง — เวลาที่ข้อมูลใช้ในการเดินทางมาถึงวัดเป็นมิลลิวินาที ระยะทางสร้าง latency นี้คือเหตุผลที่ region กรุงเทพฯ สำคัญกับผู้ใช้ชาวไทย
Console	หน้าเว็บควบคุมของผู้ให้บริการ ที่คุณใช้ดูและจัดการทุกสิ่งที่กำลังเช่าอยู่

วิดีโอประจำโมดูลนี้ (ฟรีทั้งหมด ตรวจสอบลิงก์แล้ว):

วิดีโอ	ช่อง	ความยาว	ลิงก์
What is AWS?	Amazon Web Services (ช่องทางการ)	~2 นาที	https://www.youtube.com/watch?v=a9__D53WsUs
Top 50+ AWS Services Explained in 10 Minutes	Fireship	~10 นาที	https://www.youtube.com/watch?v=JlbiYCM48to
What is Microsoft Azure? An Introduction	Eye on Tech	~3 นาที	https://www.youtube.com/watch?v=I9JkLhvaKA8

วิดีโอ	ช่อง	ความยาว	ลิงก์
รู้จัก AWS Cloud คืออะไร (คลิปแนะนำภาษาไทย)	Aware Corporation	สั้น	https://www.youtube.com/watch?v=nrSpZKGxXd0

Fluency drill — พุดจนเป็นธรรมชาติ: “Is that workload on-prem or in the cloud?” (workload ตัวนั้นอยู่ on-prem หรืออยู่บนคลาวด์?) · “Which region are we in — and are we multi-AZ?” (เราอยู่ region ไหน — แล้วเรา multi-AZ หรือเปล่า?) · “Is this an IaaS approach or is there a managed service that removes the maintenance?” (นี่เป็นแนวทาง IaaS หรือมี managed service ที่ช่วยตัดภาระการดูแลออกไปได้?)

แบบฝึกหัด: (1) สมัครบัญชี AWS ฟรีของคุณที่ <https://aws.amazon.com/free> — ขั้นตอนการสมัครเองก็สอนคำศัพท์ให้คุณได้มากกว่าการอ่านหนังสือหนึ่งบท (2) ดูวิดีโอทั้งสี่คลิปในตารางด้านบน (รวมกันไม่ถึง 20 นาที) (3) ในสมุดบันทึกของคุณ: เขียนคำอธิบายคลาวด์แบบกระชับที่คุณจะเล่าให้ผู้อำนวยความสะดวกโรงเรียนไทยฟังได้ในลมหายใจเดียว

Milestone: คุณอธิบาย IaaS กับ PaaS กับ SaaS ด้วยการเปรียบเทียบของคุณเองได้ และอธิบายได้ว่าทำไมการที่ AWS เปิด region ที่กรุงเทพฯ จึงสำคัญกับธนาคารไทย (คำตอบ: latency + data residency — ดูโมดูล 6)

โมดูล 2 (สัปดาห์ 3–4): องค์ประกอบพื้นฐาน — compute, storage และ database

Compute — เครื่องยนต์ที่รันโค้ด Virtual machine (VM/instance) (เครื่องเสมือน) คือส่วนแบ่งของ server จริงที่ทำตัวเหมือนคอมพิวเตอร์เต็มเครื่อง คุณเลือกขนาด (CPU/RAM) แล้วจ่ายตามวินาทีที่มันทำงาน **Container** (คอนเทนเนอร์) คือวิธีบรรจุแอปพลิเคชันหนึ่งตัวที่เบากว่าและเร็วกว่า เพื่อให้มันรันได้เหมือนกันทุกที่ โดย **Docker** เป็นเครื่องมือบรรจุ และ **Kubernetes** (K8s) ทำหน้าที่ควบคุมวงจรชีวิตของ container เป็นกองทัพอัน — เมื่อได้ยินคำว่า “K8s” ให้นึกถึง “ระบบที่รันและซ่อมแซม container หลายร้อยตัวของเราโดยอัตโนมัติ” ส่วน **Serverless** (เซิร์ฟเวอร์เลส — เช่น AWS Lambda) หมายถึงคุณอัปโหลดแค่ฟังก์ชันโค้ด แล้วคลาวด์จะรันมันเมื่อถูกเรียกใช้ คิดเงินต่อครั้งที่เรียก — ไม่มี server ให้ต้องดูแลเลย รูปแบบที่ควรสังเกต: VM → container → serverless คือแถบเลื่อนจาก “ควบคุมได้มาก ดูแลเยอะ” ไปสู่ “ควบคุมได้น้อย แทบไม่ต้องดูแล” ทีมที่ดีจะเลือกตามลักษณะของแต่ละ workload ไม่ใช่ตามกระแส

Storage — สามรูปทรงของพื้นที่จัดเก็บ Object storage (Amazon S3): ถังเก็บไฟล์แบบไร้กันบั้ง — รูปภาพ วิดีโอ ข้อมูลสำรอง ชุดข้อมูล ราคาถูก ทนทานระดับ “สิบเอ็ดเลขเก้า” (11 nines durability) และเป็นคำตอบมาตรฐานของคำถาม “เราจะเก็บไฟล์ไว้ที่ไหน?” **Block storage** (EBS): ฮาร์ดดิสก์เสมือนที่

ต่อกับ VM **File storage** (EFS): ไดรฟ์ที่แชร์ให้หลายเครื่องต่อใช้พร้อมกัน แล้วก็มี **tier** (ระดับชั้นการเก็บข้อมูล): แบบ hot (เข้าถึงบ่อย ราคาแพง) กับ cold/archive (เช่น Glacier — ถูก แต่ช้า) — การย้ายข้อมูลเก่าไปเก็บใน tier เย็นคือหนึ่งในวิธีลดต้นทุนที่ง่ายที่สุดที่ทีมไหนก็ทำได้ทันที

Database — สองตระกูลใหญ่ Relational/SQL (เช่น MySQL, PostgreSQL ซึ่งมีเวอร์ชัน managed คือ Amazon RDS/Aurora): เก็บข้อมูลเป็นตารางที่มีโครงสร้างเข้มงวด เป็นค่าเริ่มต้นสำหรับเงิน คำสั่ง ข้อ ผู้ใช้ — อะไรก็ตามที่ความถูกต้องเป็นเรื่องศักดิ์สิทธิ์ **NoSQL** (DynamoDB, MongoDB): ยืดหยุ่นและขยายขนาดได้มหาศาล เป็นค่าเริ่มต้นสำหรับข้อมูลปริมาณมหาศาล เร็ว และรูปทรงเรียบง่ายกว่า (session, แคตตาล็อกสินค้า, ฟีด) คำว่า “managed database” หมายถึงผู้ให้บริการดูแลการสำรองข้อมูล การอัปเดตแพตช์ และ failover (การสลับไปเครื่องสำรองเมื่อเครื่องหลักล่ม) ให้ทั้งหมด — ทีมที่จะดูแล database เองต้องมีเหตุผลที่หนักแน่นจริง ๆ

คำศัพท์:

คำศัพท์	ความหมาย
Instance	เครื่องเสมือน (VM) ที่เช่าไว้หนึ่งเครื่อง “spin up an instance” = เปิด server ขึ้นมาหนึ่งเครื่อง
Instance type / size	สเปกที่คุณเลือกให้เครื่อง — มีกี่ CPU หน่วยความจำเท่าไร ยิ่งใหญ่ ราคาต่อชั่วโมงยิ่งแพง
Container	แพ็คเกจน้ำหนักรเบ่าที่บรรจุแอปพลิเคชันหนึ่งตัว พร้อมทุกสิ่งที่คุณต้องใช้ เพื่อให้รันได้เหมือนกันบนเครื่องไหนก็ได้ เร็วกว่าและถูกกว่า VM เต็มเครื่อง
Docker	เครื่องมือมาตรฐานสำหรับสร้างและรัน container
Kubernetes (K8s)	ตัวควบคุมวงจร (orchestrator) ที่รันและซ่อมแซม กองทัพ container โดยอัตโนมัติ — ริสตาร์ดตัวที่ลุ่มเพิ่มจำนวนเมื่อโหลดสูง “K8s” คือชื่อเล่นในวงการ
Cluster / node	cluster คือกลุ่มเครื่องที่ทำงานร่วมกันเป็นระบบเดียว แต่ละเครื่องในนั้นคือ node
Serverless	การรันโค้ดโดยไม่ต้องดูแล server ใด ๆ เลย — คลาวด์รันฟังก์ชันของคุณเมื่อถูกเรียกใช้ และคิดเงินต่อครั้งที่รัน
Lambda / function	ผลิตภัณฑ์ serverless ของ AWS ส่วน “function” คือ ชิ้นโค้ดเล็ก ๆ ที่มันรัน

คำศัพท์	ความหมาย
S3 / bucket	object storage ของ AWS สำหรับเก็บไฟล์ โดย bucket คือถังเก็บไฟล์หนึ่งใบที่มีชื่อกำกับ เป็นคำตอบมาตรฐานของ “เราจะเก็บไฟล์ไว้ที่ไหน?”
EBS volume	ฮาร์ดดิสก์เสมือนที่ต่ออยู่กับ instance
Durability	โอกาสที่ข้อมูลซึ่งเก็บไว้จะอยู่รอด “สิบเอ็ดเลขเก้า” ของ S3 (99.999999999%) หมายความว่าโอกาสข้อมูลหายแทบเป็นศูนย์
Storage tier	ระดับชั้นราคา/ความเร็วของข้อมูล: hot (เข้าถึงทันที ราคาแพง) → cold/archive (Glacier — ถูก แต่ใช้เวลาดึงข้อมูลเป็นนาทีถึงชั่วโมง)
RDS	บริการ relational database แบบ managed ของ AWS — AWS ดูแล backup แพตช์ และ failover ให้คุณ
SQL vs NoSQL	SQL: ตารางโครงสร้างเข้มงวด เหมาะที่สุดกับเงินและคำสั่งซื้อ NoSQL: ยืดหยุ่นและขยายขนาดได้มหาศาล สำหรับ session แคตตาล็อก และฟีด
Backup / snapshot	สำเนาข้อมูลที่บันทึกไว้ (snapshot คือสำเนา ณ จุดเวลาหนึ่งของดิสก์หรือ database) ที่คุณกู้คืนได้
Failover	การสลับไปยังสำเนาสำรองโดยอัตโนมัติเมื่อตัวหลักล่ม — เหตุผลที่ managed database รอดพ้นคืนเลวร้ายมาได้

วิดีโอประจำโมดูลนี้:

วิดีโอ	ช่อง	ความยาว	ลิงก์
Getting Started with EC2	AWS Developers (ช่องทางการ)	27 นาที	https://www.youtube.com/watch?v=nJ-djerESW0
Introduction to Amazon S3	Amazon Web Services (ช่องทางการ)	~5 นาที	https://www.youtube.com/watch?v=ecv-19sYL3w

วิดีโอ	ช่อง	ความยาว	ลิงก์
Docker in 100 Seconds	Fireship	2 นาที	https://www.youtube.com/watch?v=Gjnup-PuquQ
Kubernetes explained in 15 mins	TechWorld with Nana	~16 นาที	https://www.youtube.com/watch?v=VnvRFRk_51k
Serverless Computing in 100 Seconds	Fireship	~2 นาที	https://www.youtube.com/watch?v=W_VV2Fx32_Y

Fluency drill: “Should this run on VMs, containers, or serverless — and what’s the operational cost of each choice?” (งานนี้ควรรันบน VM, container หรือ serverless — แล้วต้นทุนด้านการดูแลของแต่ละทางเลือกเป็นอย่างไร?) · “Is that data hot or can it go to a cheaper tier?” (ข้อมูลชุดนั้นถูกใช้บ่อยจริงไหม หรือย้ายไป tier ที่ถูกกว่าได้?) · “Why are we self-managing that database instead of using RDS?” (ทำไมเราถึงดูแล database ตัวนั้นเองแทนที่จะใช้ RDS?)

แบบฝึกหัด: (1) ในบัญชีฟรีของคุณ: เปิดเครื่อง EC2 instance ขนาดเล็กที่สุด แล้วปิดทิ้ง อัปโหลดไฟล์หนึ่งไฟล์ขึ้น S3 — ตอนนี้นักเรียนได้ใช้ IaaS ด้วยมือตัวเองแล้ว (2) ขอให้ AI ช่วยทดสอบคุณ: “ให้โจทย์ฉัน 10 สถานการณ์ ฉันจะตอบว่าควรใช้ VM, container หรือ serverless แล้วช่วยตรวจให้คะแนนด้วย”

Milestone: เมื่อได้ฟังคำอธิบายแอบง่าย ๆ หนึ่งประโยค (“เว็บไซต์ให้นักเรียนสั่งอาหารกลางวัน”) คุณสามารถบอกชื่อส่วนประกอบด้าน compute, storage และ database ของมันออกมาดัง ๆ ได้ภายใน 60 วินาที

โมดูล 3 (สัปดาห์ 5–6): เครือข่าย และวิธีที่ชิ้นส่วนต่าง ๆ คุยกัน

VPC — ย่านส่วนตัวของคุณ Virtual Private Cloud คือพื้นที่ส่วนตัวที่ล้อมรั้วไว้ในเครือข่ายของผู้ให้บริการ ข้างในมี **subnet** (ซับเน็ต — เครือข่ายย่อย) ทั้งแบบ public (เข้าถึงได้จากอินเทอร์เน็ต เช่น web server) และแบบ private (เข้าถึงไม่ได้ เช่น database) ประโยชน์ด้านความปลอดภัยที่คุณจะได้ยินบ่อยที่สุด: “*database อยู่ใน private subnet*” ถ้าคุณเข้าใจว่าทำไม — ผู้โจมตีแต่ละต้องสิ่งที่ไม่มีความปลอดภัยจากอินเทอร์เน็ตไม่ได้ — คุณก็เข้าใจความปลอดภัยด้านเครือข่ายไปครึ่งหนึ่งแล้ว

ชั้นของการจราจร: load balancer (ตัวกระจายโหลด) กระจายทราฟฟิกขาเข้าไปยัง server หลายเครื่อง (และเจียบ ๆ ถอดเครื่องที่ป่วยออกจากวง) · **DNS** (Route 53) แปลชื่ออย่าง yourcompany.com ให้เป็นที่อยู่เครือข่าย · **CDN** (CloudFront) เก็บสำเนาเนื้อหาของคุณไว้ตามเมืองหลายร้อยแห่งเพื่อให้โหลดเร็วทุกที่ · **API** คือประตูที่ซอฟต์แวร์ตัวหนึ่งเปิดให้ซอฟต์แวร์อีกตัวเข้ามาใช้งาน — เมื่อวิศวกรพูด

ว่า “เราจะ expose API” เขາหมายถึง “เราจะเปิดช่องทางที่ควบคุมได้ให้ซอฟต์แวร์อื่นมาใช้ของเรา” . ส่วน **API gateway** คือคอน์เตอร์ประชาสัมพันธ์ที่บริหารประตูเหล่านั้น

การเชื่อมต่อโลก: VPN (อุโมงค์เข้ารหัสผ่านอินเทอร์เน็ต) หรือ **Direct Connect** (สายสัญญาณส่วนตัวทางกายภาพ) ใช้เชื่อมสำนักงานหรือระบบ on-prem ของบริษัทเข้ากับคลาวด์ **Hybrid cloud** = ใช้ทั้ง on-prem และคลาวด์ควบคู่กัน (องค์กรไทยส่วนใหญ่เป็นแบบนี้) ส่วน **multi-cloud** = ใช้ผู้ให้บริการมากกว่าหนึ่งราย

คำศัพท์:

คำศัพท์	ความหมาย
VPC (Virtual Private Cloud)	พื้นที่ส่วนตัวที่ล้อมรั้วไว้ของคุณเองในเครือข่ายของผู้ให้บริการ ที่ซึ่งระบบของคุณอาศัยอยู่
Subnet (public / private)	เครือข่ายย่อยภายใน VPC โดย public subnet เข้าถึงได้จากอินเทอร์เน็ต (web server) ส่วน private subnet เข้าถึงไม่ได้ (database)
IP address	ที่อยู่ตัวเลขของเครื่องบนเครือข่าย — วิธีที่คอมพิวเตอร์หากันเจอ
Firewall / security group	รายการกฎที่กำหนดว่าทราฟฟิกแบบไหนเข้าถึงเครื่องได้บ้าง (“เว็บทราฟฟิกเข้าได้ อย่างอื่นห้าม”) ส่วน security group คือ firewall ระดับรายเครื่องของ AWS
Load balancer	ผู้กำกับการจราจรที่กระจายคำขอขาเข้าไปยัง server หลายเครื่อง และหยุดส่งไปยังเครื่องที่ไม่แข็งแรง
DNS	สมุดโทรศัพท์ของอินเทอร์เน็ต — แปล yourcompany.com เป็น IP address บริการ DNS ของ AWS คือ Route 53
CDN (Content Delivery Network)	สำเนาเนื้อหาของคุณที่ถูกเก็บแคชไว้ตามเมืองหลายร้อยแห่ง เพื่อให้โหลดเร็วทุกที่ ของ AWS คือ CloudFront
Edge location	จุด CDN ระดับเมืองเหล่านั้นแต่ละจุด — “the edge” หมายถึงจุดที่อยู่ใกล้ผู้ใช้

คำศัพท์	ความหมาย
API	ประตูที่ซอฟต์แวร์ตัวหนึ่งเปิดให้ซอฟต์แวร์อีกตัวเข้ามาใช้ “we’ll expose an API” = เราจะเปิดช่องทางที่ควบคุมได้ให้ซอฟต์แวร์อื่นมาใช้ของเรา
API gateway	เคาน์เตอร์ประชาสัมพันธ์ที่บริหารประตูเหล่านั้น — การยืนยันตัวตน การจำกัดอัตราเรียกใช้ (rate limit) และการเก็บ log
VPN	อุโมงค์เข้ารหัสบนอินเทอร์เน็ตสาธารณะที่เชื่อมต่อสองเครือข่ายเข้าด้วยกัน (เช่น สำนักงานของคุณกับ VPC ของคุณ)
Direct Connect	สายสัญญาณส่วนตัวทางกายภาพตรงเข้าสู่คลาวด์ — เร็วกว่าและนิ่งกว่า VPN แลกกับราคาที่ต้องจ่าย
Hybrid / multi-cloud	hybrid: ใช้ on-prem และคลาวด์ควบคู่กัน (องค์กรไทยส่วนใหญ่) multi-cloud: ใช้ผู้ให้บริการมากกว่าหนึ่งราย
Ingress / egress	ทราฟฟิกที่เข้าสู่ / ออกจากคลาวด์ egress มีค่าใช้จ่าย — จำค่านี้ไว้ให้ดีสำหรับโมดูล 7

วิดีโอประจำโมดูลนี้: AWS Networking Basics — VPC & Subnets for Beginners — KodeKloud — https://www.youtube.com/watch?v=QM63dyA_4Pc (ดูครั้งแรกตอนนี้ แล้วกลับมาดูส่วนที่เหลือหลังจบโมดูล 5)

Fluency drill: “Is the database in a private subnet?” (database อยู่ใน private subnet หรือเปล่า?) · “What happens when one web server dies — is the load balancer health-checking?” (ถ้า web server ตายไปหนึ่งเครื่องจะเกิดอะไรขึ้น — load balancer มี health check ไหม?) · “Are we exposing that as an API or is it internal only?” (เราเปิดตัวนั้นเป็น API ให้ภายนอกใช้ หรือใช้ภายในเท่านั้น?)

แบบฝึกหัด: (1) ให้ AI พาคคุณวาดเครือข่ายของแอปสั่งอาหารลงบนกระดาษ: ผู้ใช้ → CDN → load balancer → web server (public subnet) → database (private subnet) วาดซ้ำสามรอบจนวาดจากความจำได้ (2) หาแผนภาพสถาปัตยกรรมของบริษัทคุณหรือตัวอย่างใดก็ได้ แล้ววงกลมทุกกล่องที่ตอนนี้คุณเรียกชื่อได้แล้ว

Milestone: คุณวาดแผนภาพสามชั้น (three-tier) มาตรฐานบนไวต์บอร์ดได้ และเล่าเส้นทางของการคลิกหนึ่งครั้งของลูกค้าที่วิ่งผ่านระบบได้ตั้งแต่ต้นจนจบ

โมดูล 4 (สัปดาห์ 7–8): ทีมคลาวด์ทำงานกันอย่างไร — DevOps, IaC และการอ่านบรรยากาศในห้อง

DevOps (เดฟอ็อปส์) คือวัฒนธรรมการหลอมรวม “คนเขียนซอฟต์แวร์” (Dev) กับ “คนดูแลระบบ” (Ops) ให้เป็นทีมเดียวที่ปล่อยการเปลี่ยนแปลงเล็ก ๆ ออกไปบ่อย ๆ อย่างปลอดภัย โดยให้ระบบอัตโนมัติรับงานหนักแทน หัวใจที่เด่นอยู่ของมันคือ **CI/CD pipeline** (ไปป์ไลน์อัตโนมัติ): ทุกการเปลี่ยนแปลงโค้ดจะถูก build ทดสอบ และ deploy โดยอัตโนมัติ (Continuous Integration / Continuous Delivery) เมื่อทีมพูดว่า “it’s in the pipeline” เขาหมายถึงหุ่นยนต์กำลังทดสอบและส่งมอบมันอยู่

Infrastructure as Code (IaC) — แนวคิดที่เปลี่ยนทุกอย่าง: แทนที่จะคลิกสร้าง server ทีละตัวในหน้าคอนโซล วิศวกรจะเขียนไฟล์ข้อความที่ประกาศโครงสร้างพื้นฐานทั้งหมด (“server สองเครื่อง load balancer หนึ่งตัว database หนึ่งตัว กฎ firewall ชุดนี้”) แล้วให้เครื่องมือ (**Terraform**, **CloudFormation**) เนรมิตความเป็นจริงให้ตรงกับไฟล์ ทำไมผู้นำต้องสนใจ: ไฟล์เหล่านี้อยู่ใน **Git** (ระบบควบคุมเวอร์ชัน) ดังนั้นทุกการเปลี่ยนแปลงโครงสร้างพื้นฐานจะถูกทบทวน ย้อนกลับได้ และตรวจสอบย้อนหลังได้ — นี่คือการต่างระหว่างโรงช่างกับโรงงาน

พิธีกรรมที่คุณจะได้เข้าไปนั่งด้วย: **stand-up** (ประชุมยืนสั้น ๆ วันละ 15 นาที: ทำอะไรเสร็จแล้ว จะทำอะไรต่อ ติดขัดอะไรอยู่ — ให้ฟังหา blocker เพราะการปลดล็อกมันคืองานของคุณ) · **sprint** (สปรินต์ — หน่วยของงานที่วางแผนไว้ครั้งละ 1–2 สัปดาห์) · **retro** (ประชุมทบทวนว่าจะปรับปรุงอะไร) · **post-mortem/incident review** (การวิเคราะห์หลังเหตุการณ์ระบบล่ม: วิเคราะห์แบบ *ไม่กล่าวโทษใคร* ว่าอะไรพัง และอะไรจะป้องกันไม่ให้เกิดซ้ำ — สุขภาพของทีมดูได้จากความตรงไปตรงมาของการประชุมนี้) · **on-call** (เวรรับแจ้งเหตุ — ตารางหมุนเวียนว่าใครจะถูกปลุกตอนตีสาม ถ้า on-call ทรมาณเกินไป คนเก่งที่สุดของคุณจะลาออก — ถามไถ่เรื่องนี้ทุกเดือน)

ตัวชี้วัดสุขภาพที่สำคัญ: **uptime/availability** (สัดส่วนเวลาที่ระบบพร้อมใช้ — “three nines” = 99.9% ≈ ล่มได้ 8.8 ชั่วโมง/ปี เลขเก้าแต่ละตัวที่เพิ่มขึ้นทำให้ต้นทุนทวีคูณ) · **SLA** (คำมั่นสัญญาต่อลูกค้า พร้อมบทปรับ) · **SLO** (เป้าหมายภายใน) · **MTTR** (ความเร็วในการกู้ระบบคืน — ทีมที่โตแล้วจะมุ่งปรับปรุงการกู้คืน ไม่ใช่ฝันเฟื่องว่าจะไม่มีวันพัง) · **deployment frequency** (ความถี่ในการ deploy — ทีมสุขภาพดีปล่อยการเปลี่ยนแปลงเล็ก ๆ บ่อย ๆ ความถี่การ deploy คือกลิ่นไม่ดีอย่างหนึ่ง)

คำศัพท์:

คำศัพท์	ความหมาย
DevOps	วัฒนธรรมทีมเดียวที่ทั้งสร้างและดูแลซอฟต์แวร์ของตัวเอง ปล่อยการเปลี่ยนแปลงเล็ก ๆ บ่อย ๆ ด้วยระบบอัตโนมัติ

คำศัพท์	ความหมาย
CI/CD pipeline	สายพานอัตโนมัติที่ build ทดสอบ และ deploy ทุกการเปลี่ยนแปลงโค้ด (Continuous Integration / Continuous Delivery)
Deploy / rollback	deploy: ปล่องการเปลี่ยนแปลงขึ้นระบบจริง rollback: ถอยกลับอย่างรวดเร็วเมื่อมันมีปัญหา
Git / repository / pull request	Git: ระบบควบคุมเวอร์ชันที่บันทึกทุกการเปลี่ยนแปลง repository ("repo"): บ้านของโค้ดหนึ่งโปรเจกต์ pull request (PR): ข้อเสนอการเปลี่ยนแปลงที่วิศวกรอีกคนรีวิวก่อนรวมเข้าระบบ
IaC (Infrastructure as Code)	การประกาศโครงสร้างพื้นฐานเป็นไฟล์ข้อความที่เครื่องมือแปลงให้เป็นความจริง — รีวิวได้ ย้อนกลับได้ ตรวจสอบย้อนหลังได้
Terraform	เครื่องมือ IaC ที่ได้รับความนิยมสูงสุด (ใช้ได้กับทุกคลาวด์) ส่วนของ AWS เองคือ CloudFormation
Staging vs production ("prod")	staging: สำเนาซ้อมของระบบ prod: ระบบจริงที่ลูกค้าใช้อยู่ "broke prod" (ทำ prod พัง) = วันช่วยของทีม
Stand-up	ประชุมซิงก์วันละ 15 นาที: เสร็จอะไร / ต่อด้วยอะไร / ติดอะไร พังหา blocker — การปลดล็อกมันคืองานของคุณ
Sprint / backlog	sprint: หน่วยของงานที่วางแผนไว้ครั้งละ 1–2 สัปดาห์ backlog: รายการสิ่งที่ต้องทำเรียงลำดับความสำคัญ ที่คอยป้อนงานเข้า sprint
Retro	ประชุมท้าย sprint ว่าจะทำงานให้ดีขึ้นได้อย่างไรในรอบหน้า
Post-mortem	การทบทวนแบบไม่กล่าวโทษใครหลังระบบล่ม: อะไรพัง เพราะอะไร และอะไรจะป้องกันไม่ให้เกิดซ้ำ
On-call	ตารางเวรหมุนเวียนว่าใครเป็นคนรับสัญญาณเตือนตอนตีสาม ถ้า on-call ธรรมดา คนเก่งที่สุดของคุณจะลาออก

คำศัพท์	ความหมาย
Incident / sev-1	เหตุขัดข้องที่ไม่ได้วางแผนไว้ “sev-1” (severity one) = ระดับร้ายแรงที่สุด ทุกคนต้องลงมือช่วย
SLA / SLO	SLA: คำมั่นสัญญาต่อลูกค้าพร้อมบทปรับ (เช่น uptime 99.9%) SLO: เป้าหมายภายในที่เข้มกว่าเพื่อปกป้อง SLA
MTTR	Mean Time To Recovery — ความเร็วในการกลับมาให้บริการหลังระบบพัง ทีมที่โตแล้วจะมุ่งปรับปรุงตัวนี้ ไม่ใช่ฝันเฟื่องว่าจะไม่มีวันพัง
Monitoring / observability	การเฝ้าดูสุขภาพระบบแบบเรียลไทม์ผ่าน logs (บันทึกเหตุการณ์), metrics (ตัวเลขตามช่วงเวลา) และ alerts (การแจ้งเตือนอัตโนมัติเมื่อค่าเกินเกณฑ์)

วิดีโอประจำโมดูลนี้:

วิดีโอ	ช่อง	ความยาว	ลิงก์
What is DevOps? REALLY understand it	TechWorld with Nana	~15 นาที	https://www.youtube.com/watch?v=0yWAtQ6wYNM
DevOps CI/CD Explained in 100 Seconds	Fireship	2 นาที	https://www.youtube.com/watch?v=scEDHsr3APg
Terraform explained in 15 mins	TechWorld with Nana	18 นาที	https://www.youtube.com/watch?v=I5k1ai_GBDE

Fluency drill: “Is that change through the pipeline or was it manual?” (การเปลี่ยนแปลงนั้นผ่าน pipeline หรือทำมือ?) · “What did the post-mortem conclude, and what’s the prevention item?” (post-mortem สรุปว่าอย่างไร แล้วมาตรการป้องกันคืออะไร?) · “What’s our MTTR trending like?” (แนวโน้ม MTTR ของเราเป็นอย่างไร?) · “Is this in Terraform, or did someone click it into existence?” (อันนี้อยู่ใน Terraform หรือมีใครไปคลิกสร้างมันขึ้นมา?) — อย่างหลังเรียกว่า “ClickOps” ซึ่งเวลาพูดต้องพูดพร้อมขมวดคิ้ว

แบบฝึกหัด: (1) อ่านหรือดูการพูดคุยรายงาน post-mortem ของเหตุการณ์จริงสักหนึ่งกรณี (มีเผยแพร่สาธารณะมากมาย — รายงานเหตุล่มของ Cloudflare และ AWS มีชื่อเสียงมาก) แล้วสรุปลงสมุดบันทึกในห้าประโยค (2) เข้าร่วม (หรือดูวิดีโอบันทึก) stand-up สักครั้ง แล้วจด blocker สามข้อที่คุณได้ยิน

Milestone — จบช่วงที่ 1: คุณนั่งฟังการประชุมวางแผนเชิงเทคนิค 30 นาทีแล้วตามทันอย่างน้อย 80% และสมุดบันทึกของคุณคือหลักฐาน: โน้ตจากการประชุมจริงหรือจำลองหนึ่งครั้ง พร้อมด้วยย่อทุกตัวที่ขยายความได้ถูกต้อง **นี่ยังเป็นจังหวะที่คุณควรลงตารางสอบ AWS Cloud Practitioner (CLF-C02) —** การเตรียมตัวแบบเข้มข้น 2–4 สัปดาห์ต่อยอดจากโมดูลเหล่านี้คือมาตรฐานที่มีการเผยแพร่ไว้ และการสอบผ่านคือหลักฐานภายนอกชิ้นแรกของคุณ

ช่วงที่ 2 — คิดแบบคนในห้อง (สัปดาห์ 9–16)

โมดูล 5 (สัปดาห์ 9–10): สถาปัตยกรรม — ตัดสินคุณภาพของแบบโดยไม่ต้องวาดเอง

บทบาทของคุณใน design review ไม่ใช่การออกแบบ — แต่คือการซักถาม กรอบที่ทั้งอุตสาหกรรมใช้คือ **Well-Architected Framework** ของ AWS ซึ่งมีหกเสาหลักที่ทุกการออกแบบต้องตอบให้ได้: operational excellence, **security** (ความปลอดภัย), **reliability** (ความน่าเชื่อถือ), performance efficiency, **cost optimization** (การจัดการต้นทุน), sustainability จงเรียงสามตัวที่เป็นตัวหนาให้ลึก เพราะเงินและความเสี่ยงอาศัยอยู่ตรงนั้น

Reliability พูดย่อภาษาคน: ทุกอย่างพังได้เสมอในที่สุด ระบบที่ดีจึงออกแบบโดยตั้งสมมติฐานว่ามันจะพัง **Redundancy** (ความซ้ำซ้อนเพื่อพัง — ไม่มีจุดตายจุดเดียว ของสำคัญต้องมีสองชุด) · **multi-AZ** (รอดจากการสูญเสีย data center หนึ่งแห่ง) · **auto-scaling** (เพิ่ม/ลดเครื่องอัตโนมัติตามความต้องการ) · **backup ที่ผ่านการทดสอบแล้ว** (backup ที่ไม่เคยทดสอบคือความหวัง ไม่ใช่แผน) · **RTO/RPO** (Recovery Time Objective: เราล้มได้นานแค่ไหน; Recovery Point Objective: เราอมเสียข้อมูลได้มากแค่ไหน — สองตัวเลขนี้คือบทสนทนาเรื่อง disaster recovery ทั้งหมด และมันเป็นการตัดสินใจทางธุรกิจ ซึ่งแปลว่าเป็นของคุณ)

สามเหลี่ยมแห่งการแลกเปลี่ยน: เร็ว ถูก หนานาน — เลือกได้สองอย่าง ทุกการถกเถียงเรื่องสถาปัตยกรรมที่คุณจะได้เป็นกรรมการตัดสิน สุดท้ายจะย่อลงมาเหลือคำถามเดียวว่าธุรกิจต้องการยืนอยู่ตรงไหนบนสามเหลี่ยมนั้นสำหรับ workload ตัวนี้ ระบบชำระเงินกับเว็บไซต์การตลาดไม่ควรได้คำตอบเดียวกัน

เจ็ดคำถามของผู้นำ — ท่องให้ขึ้นใจ เพราะมันทำให้คุณนำเกรงขามใน design review ทุกครั้ง:

1. “What happens when this component fails?” (จะเกิดอะไรขึ้นเมื่อชิ้นส่วนนี้พัง? — คำว่า “เมื่อ” มีอยู่เสมอ)
2. “What are the RTO and RPO, and who signed off on them?” (RTO กับ RPO คือเท่าไร และใครเป็นคนอนุมัติตัวเลขนั้น?)
3. “Where does this run — single AZ, multi-AZ, multi-region — and why?” (ระบบนี้รันที่ไหน — AZ เดียว, multi-AZ, multi-region — และเพราะอะไร?)
4. “What will this cost per month at 10× today’s load?” (ถ้าโหลดโตเป็น 10 เท่าของวันนี้ ระบบนี้จะมีค่าใช้จ่ายเดือนละเท่าไร?)
5. “What’s the simplest version that meets the requirement?” (เวอร์ชันที่เรียบง่ายที่สุดที่ยังตอบโจทย์คืออะไร? — over-engineering คือโรคของมือใหม่ ความเรียบง่ายคือคุณธรรมของมืออาชีพ)
6. “What are we locked into, and what would leaving cost?” (เราติดล็อกอยู่กับอะไร และการย้ายออกจะมีต้นทุนเท่าไร? — vendor lock-in คือราคาอย่างหนึ่ง บางครั้งก็คุ้มที่จะจ่าย แต่ต้องจ่ายอย่างรู้ตัว)
7. “Who else has built this before — are we inventing or assembling?” (ใครเคยสร้างแบบนี้มาก่อนแล้วบ้าง — เรากำลังประดิษฐ์ใหม่หรือประกอบของที่มีอยู่? — จงเลือกแพตเทิร์นที่น่าเบื่อแต่พิสูจน์แล้ว)

Fluency drill: ฝึกถามคำถามข้อ 1, 4 และ 5 ด้วยน้ำเสียงอบอุ่น — คำถามเดียวกันจะฟังดูเป็นปัญหาหรือเป็นการโจมตี ขึ้นอยู่กับวิธีพูดทั้งสิ้น “Help me understand what happens if the cache goes down” (ช่วยอธิบายให้ผมเข้าใจหน่อยว่าถ้า cache ล่มจะเกิดอะไรขึ้น) ดีกว่า “did you think about failure?” (คิดเรื่องระบบพังหรือยัง?)

แบบฝึกหัด: (1) เอาสถาปัตยกรรมตัวอย่างสามแบบ (ขอให้ AI สร้างให้: เว็บ e-commerce, backend ของแอปมือถือ, แพลตฟอร์มวิเคราะห์ข้อมูล) แล้วไล่ถามครบทั้งเจ็ดคำถามกับแต่ละแบบ พร้อมเขียนคำตอบที่คุณคาดหวังจะได้ยิน (2) อ่านสรุปหน้าหนึ่งของเสาหลัก Well-Architected ได้ที่ <https://aws.amazon.com/architecture/well-architected/>

วิดีโอประจำโมดูลนี้:

วิดีโอ	ช่อง	ความยาว	ลิงก์
The Five Pillars of the AWS Well-Architected Framework	Amazon Web Services (ช่องทางการ; เสาหลักที่หก Sustainability ถูกเพิ่มเข้ามาภายหลัง)	~4 นาที	https://www.youtube.com/watch?v=KvEDbPmha6o

วิดีโอ	ช่อง	ความยาว	ลิงก์
What is the AWS Well-Architected Framework?	Tech With Lucy (อดีตพนักงาน AWS)	~10 นาที	https://www.youtube.com/watch?v=MpDJ6TCWKjk

Milestone: ใน design review จำลอง (ทำกับ AI ที่รับบทวิศวกร) คุณถามคำถามที่มีเนื้อหาสาระได้ห้าข้อ และสรุปจุดอ่อนที่สุดของการออกแบบได้ถูกต้องในตอนท้าย

โมดูล 6 (สัปดาห์ 11–12): ความปลอดภัยและการปฏิบัติตามกฎหมาย — บทสนทนาที่อาจทำให้บริษัทถึงจุดจบ

Shared responsibility model (โมเดลความรับผิดชอบร่วม) — สิ่งแรกที่ต้องเข้าใจ: ผู้ให้บริการรับผิดชอบความปลอดภัยของตัวคลาวด์เอง (อาคาร ฮาร์ดแวร์ hypervisor) ส่วนคุณรับผิดชอบความปลอดภัยของสิ่งที่คุณเอาไปใส่ในคลาวด์ (ข้อมูล กฎการเข้าถึง การตั้งค่าต่าง ๆ) การรั่วไหลของข้อมูลส่วนใหญ่เกิดจากลูกคำตั้งค่าผิดเอง — S3 bucket ที่เผลอเปิดเป็น public, access key ที่หลุดออกไป — ไม่ใช่ความผิดพลาดของผู้ให้บริการ ดังนั้น “AWS ปลอดภัย” กับ “เราปลอดภัยบน AWS” จึงเป็นคนละประโยคกัน

คำศัพท์ของการป้องกัน: IAM (Identity and Access Management — ระบบกำหนดว่าใครทำอะไรได้บ้าง เป็นสิ่งที่ถูกตรวจสอบ (audit) บ่อยที่สุดในโลกคลาวด์) · **least privilege** (หลักสิทธิ์ขั้นต่ำ — ทุกคนได้รับสิทธิ์เท่าที่จำเป็นเท่านั้น นี่คือกฎทองคำ) · **MFA** (การยืนยันตัวตนหลายชั้น — ต้องมีในทุกบัญชีของมนุษย์ ไม่มีข้อยกเว้น) · **encryption at rest and in transit** (การเข้ารหัสข้อมูลทั้งตอนเก็บบนดิสก์และตอนวิ่งบนสาย — เป็นมาตรฐานขั้นพื้นฐาน เปิดเสมอ) · **root account** (กุญแจดอกแม่ — เก็บล็อกไว้ ห้ามใช้ในงานประจำวัน) · **secrets management** (การเก็บรหัสผ่าน/กุญแจไว้ในตู้นิรภัยดิจิทัล ไม่ฝังไว้ในโค้ดเด็ดขาด) · **zero trust** (ตรวจสอบทุกอย่าง ไม่เชื่อใจตำแหน่งใดในเครือข่ายโดยปริยาย) · **penetration test** (การจ้างแฮกเกอร์มาทดสอบเจาะระบบเพื่อพิสูจน์แนวป้องกัน) · **ransomware** (มัลแวร์เรียกค่าไถ่ — เหตุผลว่าทำไม backup ที่ทดสอบแล้วและเก็บแบบ offline จึงเป็นมาตรการความปลอดภัย ไม่ใช่แค่เรื่องของฝ่ายปฏิบัติการ)

PDPA — กฎหมายข้อมูลของไทย และความได้เปรียบทางตลาดของคุณ พระราชบัญญัติคุ้มครองข้อมูลส่วนบุคคล (Personal Data Protection Act) คือ GDPR ฉบับประเทศไทย: ต้องได้รับความยินยอมก่อนเก็บข้อมูลส่วนบุคคล มีหน้าที่แจ้งเหตุข้อมูลรั่วไหลภายใน 72 ชั่วโมง ผู้ประมวลผลข้อมูลรายใหญ่ต้องมี Data Protection Officer (DPO) และมีกฎเรื่องการส่งข้อมูลข้ามพรมแดน การบังคับใช้เริ่มจริงจังในปี 2568 — ค่าปรับชุดแรกรวมกว่า 21.5 ล้านบาท รวมถึงกรณีโรงพยาบาลที่ถูกปรับเพราะกำกับดูแล vendor (ผู้รับจ้างภายนอก) ไม่ดีพอ ผลลัพธ์สองข้อสำหรับคุณ: (1) ตอนนี้องค์กรไทยทุกแห่งมีโจทย์ด้าน compliance ที่บริษัทของคุณสามารถรับจ้างจัดการและได้เงิน (2) **data residency** (การเก็บข้อมูลไว้ใน

ประเทศ) — การเก็บข้อมูลคนไทยไว้บนแผ่นดินไทย — คือแรงขับเคลื่อนจริงๆ ที่ผลักดัน workload เข้าสู่ region กรุงเทพฯ ย่อหน้านี้คือครึ่งหนึ่งของบทขายของคุณในประเทศไทย จงรู้มันให้ขึ้นใจ

คำถามด้านความปลอดภัยของผู้เฝ้า: “Who has access to production, and when did we last review the list?” (ใครมีสิทธิ์เข้าถึง production บ้าง และเราทบทวนรายชื่อครั้งสุดท้ายเมื่อไร?) · “Are we alerted on unusual access, or would we find out from the news?” (เรามีระบบแจ้งเตือนการเข้าถึงที่ผิดปกติหรือเราจะรู้จากข่าว?) · “When did we last restore a backup?” (ครั้งสุดท้ายที่เรากู้คืนข้อมูลจาก backup จริงๆ คือเมื่อไร? — ไม่ใช่แค่ถามว่า “มี backup ไหม”) · “If we lost this dataset, is it a PDPA-notifiable breach — and could we notify within 72 hours?” (ถ้าข้อมูลชุดนี้หลุด ถือเป็นเหตุที่ต้องแจ้งตาม PDPA ไหม — และเราจะแจ้งทันใน 72 ชั่วโมงหรือเปล่า?) · “What did the last pen test find, and what’s still open?” (pen test ครั้งล่าสุดพบอะไร และอะไรยังคงค้างอยู่?)

คำศัพท์:

คำศัพท์	ความหมาย
IAM (Identity and Access Management)	ระบบควบคุมว่าใคร (ทั้งคนและซอฟต์แวร์) ทำอะไรได้บ้างในคลาวด์ของคุณ — สิ่งที่ถูก audit บ่อยที่สุดในนั้น
Role / policy	policy คือชุดสิทธิ์ที่เขียนเป็นลายลักษณ์อักษร ส่วน role คือมัดของ policy ที่คนหรือโปรแกรมสวมใส่ได้
Least privilege	กฎทองคำ: ทุกคนได้สิทธิ์ขั้นต่ำเท่าที่งานต้องใช้ ไม่มากไปกว่านั้น
MFA (multi-factor authentication)	หลักฐานยืนยันชั้นที่สอง (รหัสจากโทรศัพท์ กุญแจฮาร์ดแวร์) เพิ่มจากรหัสผ่าน ต้องมีในทุกบัญชีของมนุษย์ ไม่มีข้อยกเว้น
Encryption at rest / in transit	ข้อมูลถูกเข้ารหัสขณะจัดเก็บ / ขณะเดินทางบนเครือข่าย ทั้งคู่ต้องเปิดเสมอ เป็นมาตรฐานขั้นพื้นฐาน
KMS (key management)	บริการที่เก็บรักษาและหมุนเวียนกุญแจเข้ารหัส
Secrets	รหัสผ่าน API key และ token — เก็บในตู้นิรภัยดิจิทัล ห้ามเขียนลงในโค้ดเด็ดขาด
Root account	กุญแจดอกแม่ของบัญชีคลาวด์ทั้งบัญชี ล็อกเก็บไว้ พร้อม MFA ห้ามใช้ในงานประจำวัน

คำศัพท์	ความหมาย
Zero trust	ท่าที่ด้านความปลอดภัยที่ตรวจสอบทุกคำขอ และไม่เชื่อใจตำแหน่งใดในเครือข่ายโดยปริยาย
Vulnerability / patching	ช่องโหว่ที่รู้จักในซอฟต์แวร์ / การติดตั้งตัวแก้ ระบบที่ไม่ได้แพตช์คือจุดเริ่มต้นของการเจาะระบบส่วนใหญ่
Pen test (penetration test)	นักโจมตีสายขาว (ethical hacker) ที่จ้างมาพิสูจน์ว่าแนวป้องกันของคุณพังตรงไหน ก่อนที่ตัวจริงจะมาเจอ
Ransomware	การโจมตีที่เข้ารหัสข้อมูลของคุณเพื่อเรียกค่าไถ่ — เหตุผลที่ backup ที่ทดสอบแล้วและเก็บแบบ offline เป็นมาตรการความปลอดภัย
SOC 2 / ISO 27001	ตราประทับการตรวจสอบความปลอดภัยจากหน่วยงานอิสระ ที่ลูกค้าองค์กรเรียกร้องก่อนเซ็นสัญญา
PDPA	พระราชบัญญัติคุ้มครองข้อมูลส่วนบุคคลของไทย — ความยินยอม การแจ้งเหตุรั่วไหลใน 72 ชั่วโมง กฎการส่งข้อมูลข้ามพรมแดน บังคับใช้พร้อมค่าปรับจริงตั้งแต่ปี 2568
Data residency	การเก็บข้อมูลไว้ภายในพรมแดนของประเทศทางกายภาพ — เหตุผลสำคัญที่ workload ของไทยย้ายเข้าสู่ region กรุงเทพฯ
DPO / breach notification	Data Protection Officer (บังคับสำหรับผู้ประมวลผลข้อมูลรายใหญ่) / หน้าที่แจ้งเหตุข้อมูลส่วนบุคคลรั่วไหลภายใน 72 ชั่วโมง

วิดีโอประจำโมดูลนี้: The AWS Shared Responsibility Model — Digital Cloud Training — 4 นาที — <https://www.youtube.com/watch?v=ESPBBEK-cvo>

Fluency drill: “Is that bucket public? Why?” (bucket นั้นเปิด public อยู่หรือเปล่า? เพราะอะไร?) · “Least privilege — does the intern really need prod access?” (ตามหลัก least privilege — เด็กฝึกงานจำเป็นต้องเข้าถึง prod จริง ๆ หรือ?) · “Where does the PDPA line sit in this design — what’s personal data here and where does it physically live?” (เส้นแบ่งเรื่อง PDPA ในการออกแบบนี้อยู่ตรงไหน — อะไรคือข้อมูลส่วนบุคคลในระบบนี้ และตัวข้อมูลอยู่ที่ไหนจริง ๆ ทางกายภาพ?)

แบบฝึกหัด: (1) อ่านเรื่องราวการรั่วไหลจากการตั้งค่าคลาวด์ผิดที่โด่งดังสักหนึ่งกรณี (Capital One ปี 2019 คือกรณีคลาสสิก) แล้วเขียนเวอร์ชันสามประโยคลงสมุดบันทึก (2) ให้ AI จำลองเป็นลูกค้าที่ถามว่า “ทำไมผมต้องใส่ใจให้คุณดูแลข้อมูลของผม?” แล้วฝึกตอบโดยใช้ภาษาของ shared responsibility + PDPA

Milestone: คุณอธิบาย shared responsibility model และความหมายในทางปฏิบัติของ PDPA ให้เจ้าของธุรกิจไทยที่ไม่ใช่สายเทคนิคเข้าใจได้ภายในสามนาที — เพราะคำอธิบายนั้นแหละคือการประชุมขายงานของ MSP

โมดูล 7 (สัปดาห์ 13–14): เรื่องเงิน — เศรษฐศาสตร์คลาวด์และ FinOps

นี่คือจุดที่ผู้นำที่ไม่ใช่วิศวกรสร้างคุณค่าได้เร็วที่สุด เพราะวิศวกรส่วนใหญ่ไม่เคยถูกสอนเรื่องนี้ และความสูญเปล่าในมหาศาล — ผลสำรวจหลายชิ้นพบตรงกันว่าราวหนึ่งในสามของค่าใช้จ่ายคลาวด์คือความสูญเปล่า

มิเตอร์เงินอย่างไร: ค่า compute คิดต่อวินาทีที่เครื่องเปิดอยู่ (ไม่ใช่ต่อการใช้งาน — server ที่เปิดทิ้งว่าง ๆ ก็เก็บเงินเต็ม “เราลืมปิดเครื่องทิ้งไว้” คือความสูญเปล่าคลาสสิก) ค่า storage คิดต่อ GB ต่อเดือน และ — กับดักชื่อดัง — **egress** (ค่าข้อมูลขาออก): ข้อมูลที่ไหลออกจากคลาวด์มีค่าใช้จ่าย ขณะที่ข้อมูลขาเข้าฟรี บิล egress ก่อนโตเคยทำให้ทุกคนตกใจมาแล้วคนละครึ่ง — แต่หลังจบโมดูลนี้ จะไม่ใช่คุณ

เมนูราคา: **on-demand** (ราคาเต็ม ยืดหยุ่นเต็มที่) · **reserved instances / savings plans** (ผูกสัญญา 1–3 ปี แลกส่วนลด 30–70% — คำนโยกที่ใหญ่ที่สุดสำหรับ workload ที่รันคงที่) · **spot instances** (ลดได้ถึง 90% สำหรับงานที่ยอมให้ขัดจังหวะได้ เช่นงาน batch) · **rightsizing** (การปรับขนาดให้พอดี — server ส่วนใหญ่ถูกจองเพื่อไว้เกินจำเป็น การลดขนาดลงคือเงินฟรี ๆ) · **storage tiering** (โมดูล 2) · **auto-scaling** **ในฐานะเครื่องมือลดต้นทุน** (จะจ่ายค่าความจุระดับเที่ยงคืนในราคาช่วงเที่ยงวันไปทำไม?)

FinOps (ฟินอ็อปส์ — แนวปฏิบัติด้านการเงินคลาวด์) คือการทำให้ค่าใช้จ่ายคลาวด์มองเห็นได้ ระบุเจ้าของได้ และถูกปรับให้เหมาะสมอย่างต่อเนื่อง: **tagging** (การติดป้ายกำกับ) ทุกทรัพยากรด้วยชื่อเจ้าของและโปรเจกต์ (ค่าใช้จ่ายที่ไม่มี tag คือค่าใช้จ่ายที่ไม่มีใครรับผิดชอบ) ·

showback/chargeback (การแสดงบิลให้แต่ละทีมเห็นของตัวเอง — พฤติกรรมจะเปลี่ยนทันที) · การตั้งงบประมาณพร้อม **alert** (อย่ารู้ว่าใช้เงินเกินจากใบแจ้งหนี้เด็ดขาด) · และ **unit economics**

(เศรษฐศาสตร์ต่อหน่วย) — ตัวชี้วัดของผู้นำ: ไม่ใช่ “บิลของเราเดือนละ 800,000 บาท” แต่เป็น “ต้นทุนต่อธุรกรรมของลูกค้าหนึ่งรายการของเรากำลังลดลง” ใบรับรอง FinOps Certified Practitioner ใช้เวลาเตรียมราวสองสัปดาห์ และสำหรับผู้นำฝั่งธุรกิจ มันคือใบรับรองที่ให้ความน่าเชื่อถือต่อชั่วโมงที่ลงทุนสูงที่สุดในโลกคลาวด์ทั้งใบ ไปสอบมาเถอะ

คำถามเรื่องเงินของผู้นำ: “What’s our cost per [customer/transaction/tenant], and which direction is it moving?” (ต้นทุนต่อ [ลูกค้า/ธุรกรรม/tenant] ของเราคือเท่าไร และกำลังขยับไปทางไหน?) · “What percentage of our steady workload is on reservations?” (workload ที่รันคงที่ของเราอยู่บน reservation กี่เปอร์เซ็นต์?) · “What’s untagged?” (มีอะไรที่ยังไม่ติด tag บ้าง?) · “What died but is still billing?” (อะไรที่ตายไปแล้วแต่ยังเก็บเงินอยู่? — ดิสก์กำพร้าวและ IP ที่จองทิ้งไว้ มีอยู่ในทุกบัญชี) · “What would this bill look like at 10× growth — does our architecture get cheaper or more expensive per unit?” (ถ้าธุรกิจโต 10 เท่า บิลนี้จะหน้าตาเป็นอย่างไร — สถาปัตยกรรมของเราจะถูกลงหรือแพงขึ้นต่อหน่วย?)

คำศัพท์:

คำศัพท์	ความหมาย
On-demand	ราคาแบบจ่ายตามใช้จริง: ราคาเต็ม ยกเลิกได้ทุกเมื่อ เป็นค่าเริ่มต้น และเป็นวิธีที่แพงที่สุดในการรัน workload ที่คงที่
Reserved / savings plan	การผูกสัญญาใช้งานคงที่ 1–3 ปี แลกกับส่วนลด 30–70% — คำนโยกลดต้นทุนที่ใหญ่ที่สุด
Spot	ความจุส่วนเกินของผู้ให้บริการในราคาดสูงสุดถึง 90% ที่ผู้ให้บริการเรียกคืนได้โดยแจ้งล่วงหน้าเป็นนาที — เหมาะที่สุดกับงาน batch ที่ยอมให้ขัดจังหวะได้
Rightsizing	การหดขนาด server ที่จองเผื่อเกินจำเป็นให้เท่ากับที่ใช้จริง เงินฟรี ๆ ที่มีอยู่ในแทบทุกบัญชี
Egress	ข้อมูลที่ออกจากคลาวด์ — คิดเงินต่อ GB ขณะที่ข้อมูลขาเข้าฟรี ตัวสร้างความประหลาดใจชื่อดังบนใบแจ้งหนี้
Tagging	การติดป้ายทรัพยากรด้วยเจ้าของ/โปรเจกต์/สภาพแวดล้อม เพื่อให้เงินทุกบาทที่จ่ายมีที่มาที่ไป ค่าใช้จ่ายไม่มี tag = ค่าใช้จ่ายไม่มีใครรับผิดชอบ
Showback / chargeback	การแสดงบิลคลาวด์ของแต่ละทีมให้ทีมนั้นเห็น (showback) หรือเรียกเก็บเงินภายในจริง ๆ (chargeback) พฤติกรรมจะเปลี่ยนทันที

คำศัพท์	ความหมาย
Budget alert	การแจ้งเตือนอัตโนมัติเมื่อค่าใช้จ่ายถึงเกณฑ์ที่ตั้งไว้ — เพื่อไม่ให้คุณรู้เรื่องใช้เงินเกินจากใบแจ้งหนี้
Unit economics	ต้นทุนต่อหน่วยธุรกิจ — ต่อลูกค้า ต่อธุรกรรม — ตัวชี้วัดของผู้นำ มีความหมายมากกว่ายอดบิลรวม
TCO (total cost of ownership)	ต้นทุนเต็มของทางเลือกหนึ่งตลอดอายุการใช้งาน: ค่า license คน ค่าไฟ การ migrate — ไม่ใช่แค่ราคาป้าย
FinOps	วินัย (และวัฒนธรรมทีม) ของการทำให้ค่าใช้จ่ายคลาวด์มองเห็นได้ ระบุเจ้าของได้ และถูกปรับให้เหมาะสมอย่างต่อเนื่อง

วิดีโอประจำโมดูลนี้: What is FinOps? — FinOps Foundation (ช่องทางการ) — 2 นาที — https://www.youtube.com/watch?v=Y-c_xw9bHFW · จากนั้นเรียนคอร์สฟรี “Introduction to FinOps” ได้ที่ <https://learn.finops.org>

แบบฝึกหัด: (1) เปิด AWS pricing calculator แล้วลองคำนวณราคาระบบเล็ก ๆ ของจริง (server สองเครื่อง, database หนึ่งตัว, storage 500GB, egress 1TB) — การได้ลงมือทำสักครั้งจะปลดปล่อยความสับสนของทุกบทสนทนาเรื่องต้นทุนในอนาคต (2) ขอให้ AI สวมบทวิศวกรที่ปกป้อง server ขนาดใหญ่เกินจำเป็น แล้วฝึกเจรจา rightsizing อย่างนุ่มนวล

Milestone: สอบ (หรือลองตารางสอบ) FinOps Practitioner และในการรีวิวจำลอง ค้นหาปัญหาต้นทุนได้สี่จุดจากบิลตัวอย่างที่ AI สร้างให้

โมดูล 8 (สัปดาห์ 15–16): ค่ายฝึกความคล่อง — ร้อยเรียงภาษาเข้าด้วยกัน

สองสัปดาห์นี้คือการบูรณาการล้วน ๆ — ความต่างระหว่างการรู้คำศัพท์ กับการพูดคล่องในห้องประชุมจริง

แบบฝึกประจำวัน (30 นาที): ให้ AI สร้างบันทึกการประชุมเสมือนจริง (design review, incident retro หรือ cost review) โดยฝังข้อผิดพลาดทางเทคนิคไว้สองจุดโดยตั้งใจ งานของคุณ: สรุปการประชุมในห้าประโยค จับข้อผิดพลาดให้ได้ และเขียนคำถามสามข้อที่คุณจะถามในที่ประชุม สลับประเภทการประชุมทุกวัน

กลั่นเนื้อการแปลความ (15 นาที): หยิบประโยคเชิงเทคนิควันละหนึ่งประโยค แล้วแปลให้ผู้ฟังสามกลุ่ม — CFO (มุมนเงิน), ลูกค้า (มุมความเสี่ยง/ประโยชน์) และวิศวกรจูเนียร์หน้าใหม่ (มุมการสอน) ตัวอย่าง:

“เรากำลังย้าย session store จาก database ไปไว้บน Redis” → CFO: “ลดภาระ database ทำให้เราเลื่อนการอัปเดตมูลค่า 2 ล้านบาทออกไปได้” → ลูกค้า: “หน้าเว็บโหลดเร็วขึ้นในช่วงพีค” → จูเนียร์: “Redis เก็บข้อมูลที่ถูกใช้บ่อยไว้ในหน่วยความจำ เราจึงเลิกกระหน่ำยิง database ทุกครั้งที่มิลิก”

ฝึกอ่าน (15 นาที): อ่านบล็อกวิศวกรรมของจริงวันละหนึ่งบทความ (AWS Architecture Blog หรือบล็อกวิศวกรรมของ Netflix/Grab — Grab เกี่ยวข้องเป็นพิเศษ: สเกลระดับเอเชียตะวันออกเฉียงใต้ ตลาดใกล้เคียงไทย) ตอนนี้คุณ sẽ อ่านเข้าใจ 70–80% แล้ว ที่เหลือให้ค้นเพิ่ม

เพื่อนคู่ใจเตรียมสอบสำหรับช่วงนี้: คอร์สเต็มพรีอันโด่งดัง — *AWS Certified Cloud Practitioner (CLF-C02) 2026* โดย Andrew Brown บน freeCodeCamp — <https://www.youtube.com/watch?v=7HKot-brXFE> (ฉบับก่อนหน้าความยาว 14 ชั่วโมง ซึ่งยังใช้ได้กับรหัสสอบเดียวกัน อยู่ที่ <https://www.youtube.com/watch?v=NhDYbskXRgc>) ดูที่ความเร็ว 1.25× กระจายไปตลอดสัปดาห์ 13–16 — หลังผ่านโมดูล 1–7 มาแล้ว เนื้อหาส่วนใหญ่จะให้ความรู้สึกเหมือนการทบทวน ซึ่งนั่นแหละคือสัญญาณว่าคุณพร้อมแล้ว

Milestone — จบช่วงที่ 2 ทบทวนจบการศึกษาของคุณ: (1) สอบผ่าน **AWS Cloud Practitioner** ถ้ายังไม่ได้สอบ (2) ด้านทดสอบจำลอง (mock gauntlet): AI รับบทวิศวกรอาวุโสที่พาคุณดูสถาปัตยกรรมที่มีข้อบกพร่องสำหรับลูกค้า e-commerce ไทย คุณต้องหาให้เจอทั้ง database ที่อยู่ AZ เดียว, S3 bucket ที่เปิด public, ประสิทธิภาพ egress ที่หายไป และประเด็น PDPA ที่ไม่ถูกพูดถึง — แล้วส่งมอบ feedback ด้วยน้ำเสียงที่ทำให้ “วิศวกร” รู้สึกว่าได้รับความช่วยเหลือ ไม่ใช่ถูกจับผิด เมื่อคุณทำได้ คุณก็เป็นคนที่ “อันตราย” ในวงสนทนาแล้ว — ภายในสิบหกสัปดาห์

ช่วงที่ 3 — นำทีม (เดือน 5–24)

โมดูล 9 (เดือน 5–8): ใ้รับรองเชิงลึกและพื้นฐานทางเทคนิคขั้นต่ำ

เรียนเนื้อหา **AWS Solutions Architect Associate (SAA-C03)** — ใช้เวลา 2–3 เดือนตามจังหวะวันละหนึ่งชั่วโมงของคุณ จะเข้าสอบหรือไม่ก็ได้ (สำหรับผู้นำ เนื้อหาคือคุณค่า ส่วนตราใบรับรองเป็นเพียงพิธีการเสริม) แต่นี่คือจุดที่ region, VPC, IAM และเรื่องราคาจะเล็กลงเป็นแค่คำศัพท์ และกลายเป็นระบบที่เชื่อมโยงกันในหัวของคุณ ระหว่างนี้ให้คงแบบฝึกประจำวันของโมดูล 8 ไว้ครึ่งโดส และนี่ยังเป็นช่วงเหมาะที่จะเสริม **พื้นฐาน Azure (ระดับ AZ-900)** — ประเทศไทยเป็นตลาดองค์กรที่ใช้ Microsoft หนักมาก และความรู้สองภาษา (AWS+Azure) จะขยายวงบทสนทนายกับลูกค้าของคุณให้กว้างขึ้น

โมดูล 10 (เดือน 5–12 ดำเนินต่อเนื่อง): การจ้างและบริหารวิศวกร

การจ้างคนที่คุณยังประเมินฝีมือเชิงลึกไม่ได้เต็มที่: โครงสร้างชนะเลิศของสัญญาฉบับแรก ใช้กระบวนการสัมภาษณ์ที่ค่อนข้างคงที่: รอบคัดกรองที่คุณเป็นผู้สัมภาษณ์เอง (แรงจูงใจ การสื่อสาร และการให้เขาอธิบายโปรเจกต์ในอดีตให้คนไม่ใช่สายวิศวกรฟัง — ถ้าอธิบายไม่ได้ เขาก็จะสื่อสารกับลูกค้าของคุณไม่ได้เช่นกัน) + รอบสัมภาษณ์เทคนิคที่ดำเนินการโดย **bar-raiser** ของคุณ (มือขวาสายเทคนิค หรือผู้รับจ้างอาวุโสที่จ้างมาช่วย) + การโทรเช็กบุคคลอ้างอิงด้วยคำถามเดียวที่สำคัญจริง: “would you hire this person again for this role?” (คุณจะจ้างคนนี้อีกครั้งสำหรับตำแหน่งนี้ไหม?) ระว่างสองบุคคลที่มักพลาด: นักพูดคล่องที่ไม่มีควมลึก (bar-raiser ของคุณจะจับได้) และผู้เชี่ยวชาญตัวจริงที่พูดน้อย (ในทีมไทยมักเป็นทองคำซ่อนอยู่ เพราะความถ่อมตัวเป็นวัฒนธรรม — อย่าให้ความสั่นไหวในห้องสัมภาษณ์มีน้ำหนักมากกว่าหลักฐานของผลงานจริง)

มือขวาสายเทคนิค (technical #2) — การตัดสินใจที่สำคัญที่สุดของกิจการนี้: จ้างเขาเป็นคนแรก ง่ายด้วยหุ้นที่มีความหมาย (10–20% หากเป็น co-founder ตัวจริง) และนิยามข้อตกลงให้ชัดเจน: เขาถือมาตรฐานทางเทคนิคและเป็นเจ้าของการตัดสินใจด้านสถาปัตยกรรม ส่วนคุณเป็นเจ้าของลูกค้า เงินลำดับความสำคัญ และคน ความเห็นต่างระหว่างคุณสองคนต้องเกิดในที่ลับ และคลี่คลายก่อนที่ทีมจะเห็น

พิธีกรรมที่ทำให้วิศวกรอยู่กับเรา: 1:1 (ประชุมตัวต่อตัว) รายสัปดาห์หรือรายปักษ์ที่เนื้อหาเป็นเรื่องของตัวเขา (เส้นทางอาชีพ อุปสรรค พลังใจ — ไม่ใช่การรายงานสถานะงาน) · เส้นทางเรดปิตช์ที่เขียนเป็นลายลักษณ์อักษรรายบุคคล (ในตลาดไทยที่ขาดแคลนบุคลากรปีละ 70,000 คน งบการเติบโตและใบรับรองรักษาคนได้ดีกว่าเงินเดือนเพียงอย่างเดียว — จ่ายค่าสอบทุกใบ พร้อมโบนัสเมื่อสอบผ่านแบบ vesting 12 เดือน) · ชมในที่แจ้ง ตักเตือนในที่ลับ · และปกป้องเวลาโฟกัสของพวกเขายิ่งที่สุด — ผู้นำที่ยกเลิกการประชุมกลาง sprint คือฮีโร่

เวทีการตัดสินใจ — วิธีตัดสินใจเรื่องเทคนิคที่คุณประเมินเองได้ไม่ครบถ้วน: สำหรับการตัดสินใจใหญ่ทุกครั้ง ให้วิศวกรผู้เสนอเขียน **decision doc** (เอกสารการตัดสินใจ) หนึ่งหน้า: ปัญหา ทางเลือก 2–3 ทาง ต้นทุน ความเสี่ยง และข้อเสนอแนะ จากนั้นดำเนินการประชุมด้วยเจ็ดคำถามจากโมดูล 5 คุณไม่ใช่สถาปนิกที่เก่งที่สุดในห้องและไม่จำเป็นต้องเป็นเลย — คุณคือคนที่ทำให้ทางเลือกที่มีเหตุผลรองรับดีที่สุดชนะ ตรงตามกำหนดเวลา โดยที่การแลกเปลี่ยนเรื่องเงินและความเสี่ยงถูกพูดออกมาอย่างชัดเจน วิศวกรเคารพสิ่งอย่างลึกซึ้งเมื่อทำอย่างตรงไปตรงมา จดไว้ว่าใครทำนายอะไร (Decision Journal ของคุณอีกแล้ว) และทบทวนคำทำนายทุกไตรมาส — มันช่วยปรับเข็มทิศทั้งของคุณและของพวกเขา

โมดูล 11 (เดือน 6–24): เส้นโค้งความน่าเชื่อถือ ฉบับตรงไปตรงมา

ประสบการณ์ที่ตีพิมพ์ในวรรณกรรมด้านการนำทีมวิศวกรลงเอยตรงกันที่ไทม์ไลน์นี้ และการเสริมว่าไม่จริงคือสาเหตุที่ founder ที่ไม่ใช่สายเทคนิคล้มเหลว: **~90 วัน** กว่าที่จะดำเนินจังหวะธุรกิจได้อย่างมีประสิทธิภาพ · **~6 เดือน** กว่าที่จะมีประสิทธิภาพระดับพื้นฐาน (ประชุมคล่อง ตัดสินใจอย่างมีโครงสร้าง ทีมมั่นคง) · **12–18 เดือน** กว่าสัญญาฉบับแรกทางเทคนิคของคุณจะเริ่มมีค่าด้วยตัวมันเอง · **~2 ปี** กว่าที่ยินยอมแลกเรื่อง trade-off ทางสถาปัตยกรรมกับวิศวกรอาวุโสได้อย่างแท้จริง วิธีประคองระหว่างที่เส้นโค้ง

กำลังไต่: ยืมความน่าเชื่อถือ (ให้มือขวาของคุณนำเสนอครึ่งเทคนิคของการประชุมขายงาน — อย่างไรลูกค้าก็ต้องการเห็นทีมงานอยู่แล้ว) · ห้ามโม้เด็ดขาด (ตัวทำลายความน่าเชื่อถือที่เร็วที่สุด — “I don’t know — walk me through it” (ผมยังไม่รู้เรื่องนี้ — ช่วยอธิบายให้ฟังหน่อย) คือประโยคของผู้นำเสนอ) · และปล่อยให้คำถามของคุณพูดแทนตัวคุณ: ผู้นำที่ถามว่า “what’s our RPO and who signed off on it?” (RPO ของเราคือเท่าไร และใครอนุมัติ?) ฟังดูเหมือนผ่านสนามรบมาสามสิบปี — และหลังจบหลักสูตรนี้ คุณจะถามอย่างเข้าใจมันจริง ๆ

โมดูล 12 (ต่อเนื่อง): ชั้นความรู้เฉพาะตลาดไทย

หลอมรวมรายละเอียดจาก TSI Part 5 ให้เป็นความรู้ปฏิบัติการ: **PDPA** ในฐานะทั้งหน้าที่ด้าน compliance และตัวสินค้า (ดูโมดูล 6) · **บันไดพาร์ทเนอร์** (AWS Select ต้องมีพนักงานถือใบรับรองจำนวนหนึ่ง + ดีลที่เปิดตัวแล้ว 3 ดีล; Microsoft Solutions Partner ต้องได้คะแนนศักยภาพ 70/100 — ใบรับรองของทีมคุณจึงเป็นสินทรัพย์การขายโดยแท้ ซึ่งเป็นอีกเหตุผลที่ควรออกเงินสนับสนุน) · การส่งเสริมการลงทุนจาก **BOI** เพื่อให้ต่างชาติถือหุ้นบริษัทได้ 100% · ช่วงเงินเดือนตลาดกรุงเทพฯ (จูเนียร์ ฿50,000–75,000 → สถาปนิก ฿180,000–280,000 ต่อเดือน) เพื่อให้คุณตั้งราคางานประมูลและยื่นข้อเสนอจ้างงานได้ถูกต้อง · และคณิตศาสตร์การขายของช่วงเวลานี้ — เงินลงทุน data center ที่อนุมัติแล้วกว่า 27,000 ล้านดอลลาร์สหรัฐ นโยบาย cloud-first ของภาครัฐ และช่องว่างทักษะปีละ 70,000 คนที่ทีมงานถือใบรับรองของคุณดำรงอยู่ก็เพื่อเติมเต็มมัน

หลักสูตรย่อหน้า

ช่วงเวลา	จุดเน้น	หลักฐานภายนอก
สัปดาห์ 1–2	คลาวด์คืออะไร; IaaS/PaaS/SaaS; region/AZ	—
สัปดาห์ 3–4	Compute, storage, database	—
สัปดาห์ 5–6	เครือข่าย; การอ่านแผนภาพ สถาปัตยกรรม	—
สัปดาห์ 7–8	DevOps, IaC, พิธีกรรมของทีม, ตัวชี้วัดด้านปฏิบัติการ	ลงตารางสอบ Cloud Practitioner

ช่วงเวลา	จุดเน้น	หลักฐานภายนอก
สัปดาห์ 9–10	วิจารณ์ญาณด้านสถาปัตยกรรม; เจ็ดคำถาม	—
สัปดาห์ 11–12	ความปลอดภัย; PDPA; shared responsibility	สอบ AWS Cloud Practitioner
สัปดาห์ 13–14	เศรษฐศาสตร์คลาวด์; FinOps	FinOps Practitioner (เตรียมตัว ≈2 สัปดาห์)
สัปดาห์ 15–16	ค่ายฝึกความคล่อง; ด้านทดสอบ จำลอง	จบการศึกษา: ผ่านด้านทดสอบ
เดือน 5–8	เนื้อหา SAA-C03; Azure AZ-900	สอบ SAA (เลือกได้)
เดือน 5–12	การจ้างงาน; มือขวาสายเทคนิค; เวทีการตัดสินใจ	จ้างพนักงานชุดแรกได้อย่างดี
เดือน 6–24	เส้นโค้งความน่าเชื่อถือ; ชั้น ความรู้ตลาดไทย	ระดับพาร์ทเนอร์; สัญญา retainer แรก ๆ

คำส่งท้ายจากครูของคุณ อีกสิบหกสัปดาห์นับจากนี้ คุณจะตามทันทุกบทสนทนาในห้อง นั่นไม่ใช่เส้นชัย — มันคือใบอนุญาตให้เริ่มต้น เส้นโค้งสองปีสู่วิจารณ์ทางเทคนิคที่แท้จริงไม่ใช่กำแพง แต่เป็นคู่มือ: ทุกสัปดาห์ที่คุณเดินผ่านมันไป คือหนึ่งสัปดาห์ที่ founder สายไม่เทคนิคของคุณแข่งไม่ได้เดิน จงเรียนทุกวัน จดบันทึกทุกการตัดสินใจ อย่ามัวเป็นอันขาด และจ้างคนที่เก่งกว่าคุณแล้วทำให้พวกเขาดีใจที่มาร่วมทีม — นั่นแหละคืองานทั้งหมด

เอกสารประกอบของ “The Thailand Strategic Investment (TSI)” Part 5 ประมาณการเวลาเตรียมสอบ ใบรับรองและไหม์ไลน์ด้านภาวะผู้นำอ้างอิงจากแหล่งข้อมูลที่จะระบุไว้ในเอกสารดังกล่าว (CBT Nuggets, StudyTech, FinOps Foundation, First Round Review, The Pragmatic Engineer)

บทเรียนใหม่ — ฉบับเดือนสิงหาคม 2026

ในเดือนสิงหาคม 2026 มีการเพิ่มบทเรียนใหม่เข้าไปในหลักสูตรแบบอินเทอร์แอคทีฟทั้งสามที่ b4ilc.org — หลักสูตรผู้นำด้านคลาวด์ (Cloud Leader) หลักสูตรวิศวกรคลาวด์ (Cloud Engineer) และ

หลักสูตรสถาปนิกคลาวด์ (Cloud Architect) ภาคผนวกนี้แนะนำสำหรับอ่านให้ทั่วทั้งเนื้อหาใหม่ นั่นด้วยฉบับย่อที่เก็บความอย่างชัดเจน ในน้ำเสียงเดียวกับส่วนอื่น ๆ ของหนังสือเล่มนี้

ขอพูดถึงสิ่งที่คุณถืออยู่ในมือสักนิด หน้ากระดาษเหล่านี้คือฉบับสำหรับอ่าน: แนวคิด อุปมา และประโยคที่ควรค่าแก่การพูดออกเสียง ฉบับอินเทอร์แอคทีฟเต็มรูปแบบบนเว็บมีมากกว่านี้มากทีเดียว — บทเรียนใหม่แต่ละบทมาพร้อมแบบทดสอบที่ตรวจสอบความเข้าใจก่อนให้คุณไปต่อ ห้องปฏิบัติการลงมือทำจริง (รวมถึง terminal สำหรับฝึกที่ทำงานเหมือนของจริงแต่ไม่มีทางทำให้เสียเงิน) วิดีโอ แบบฝึกคำศัพท์ และ — สำหรับเนื้อหา capstone — แบบฝึกออกแบบพร้อมรายการตรวจสอบข้อกำหนดและเฉลยตัวอย่างที่ทำให้ดูเป็นขั้นเป็นตอน อ่านฉบับย่อเหล่านี้เพื่อให้โครงร่างของแต่ละแนวคิดเข้าไปอยู่ในหัวก่อน แล้วเมื่อมีโอกาส จงกลับไปนั่งเรียนฉบับอินเทอร์แอคทีฟเพื่อให้ความรู้ฝังแน่น

เนื้อหาที่เพิ่มเข้ามาแบ่งเป็นสามกลุ่ม สำหรับหลักสูตรผู้นำด้านคลาวด์: บทแนะนำ terminal อย่างนุ่มนวล กฎหมายความเป็นส่วนตัวของไทยเราเอง คู่มือผู้นำสำหรับข้อเสนอโครงการ AI และมูลค่าใช้จ่าย AI และแบบฝึก capstone ที่ประกอบทั้งหลักสูตรเข้าเป็นหนึ่งหน้ากระดาษ สำหรับหลักสูตรวิศวกรคลาวด์: serverless computing — แพตเทิร์นที่เปลี่ยนสคริปต์งานปฏิบัติการเล็ก ๆ ให้กลายเป็นฟังก์ชันที่รันโดยไม่มีค่าใช้จ่าย สำหรับหลักสูตรสถาปนิกคลาวด์: เลขคณิตของคำสัญญาเรื่องความพร้อมใช้งาน แพตเทิร์น Retrieval-Augmented Generation ที่เดี๋ยวนี้ถูกคำถามถึงกันเป็นรายสัปดาห์ และแบบฝึกประเมินต้นทุนเต็มรูปแบบ — ตีราคางานออกแบบก่อนที่จะใครจะลงมือสร้าง

สำหรับหลักสูตรผู้นำด้านคลาวด์ (Cloud Leader)

หน้าต่างสีดำ — วิธีพูดกับคอมพิวเตอร์โดยตรง

ก่อนเข้าห้องปฏิบัติการครั้งแรก มาทำความรู้จักเครื่องมือที่ห้องปฏิบัติการนั้นทำงานอยู่ข้างในกันก่อน **terminal** (เทอร์มินัล — เรียกอีกอย่างว่า *command line* หรือ *console*) คือโปรแกรมหน้าต่างเหมือนหน้าต่างสีดำเรียบ ๆ ที่มีตัวอักษรกะพริบ มันเป็นวิธีใช้คอมพิวเตอร์ที่เก่าแก่ที่สุด และยังคงเป็นวิธีที่วิศวกรใช้ทำงานจริงจรงมาจนทุกวันนี้ — เพราะคำสั่งที่พิมพ์ลงไปนั้นแม่นยำ ทำซ้ำได้ และจดบันทึกส่งต่อกันได้ ในแบบที่การคลิกไล่ตามเมนูไม่มีวันทำได้

terminal จะแสดง **prompt** (พรอมต์) — มักเป็นบรรทัดสั้น ๆ ลงท้ายด้วย \$ — ซึ่งมีความหมายเพียงหนึ่งเดียว: ฉันกำลังฟังอยู่ ส่วน **command** (คำสั่ง) คือประโยคที่มีไวยากรณ์ของมันเอง: คำแรกคือเครื่องมือที่คุณเรียกใช้ (เช่น `git` หรือ `aws`) คำถัด ๆ ไปบอกว่าให้ทำอะไร และชิ้นส่วนที่ขึ้นต้นด้วยขีด ซึ่งเรียกว่า **flag** (แฟล็ก — เช่น `--output table`) ใช้ปรับว่าทำอะไร

มีธรรมเนียมอยู่สองข้อที่ทำให้มือใหม่ทุกคนแปลกใจ ข้อแรก: ความเจียบมักหมายถึงความสำเร็จ คำสั่งจำนวนมาก เมื่อทำงานสำเร็จอย่างสมบูรณ์แล้ว จะไม่พิมพ์อะไรออกมาเลย — เครื่องถือว่าการทำงาน

ให้เสร็จก็เป็นคำตอบที่เพียงพอแล้ว ข้อสอง: terminal ไม่เข้าใจ ในขณะที่ search engine ให้อภัยการสะกดผิด terminal ไม่ให้อภัยอะไรทั้งนั้น — ผิดตัวอักษรเดียว มันจะบอกคุณอย่างสุภาพว่าคำสั่งนี้ไม่มีอยู่ ความขมขื่นนี้จะรู้สึกโหดร้ายอยู่สักวันหนึ่ง แล้วจากนั้นจะกลายเป็นความอُنใจ: เครื่องไม่มีวันทำ *ประมาณ* ๆ ตามที่คุณขอ

ไวยากรณ์อีกหนึ่งชั้น: คอมพิวเตอร์จัดระเบียบทุกสิ่งเป็น **file** (ไฟล์) ที่อาศัยอยู่ใน **folder** (โฟลเดอร์ — เรียกอีกอย่างว่า *directory*) — ดังนั้น `mkdir decisions` จึงแปลว่า “*make directory* — สร้าง directory ชื่อ decisions” ห้องปฏิบัติการของหลักสูตรรันอยู่ใน **simulator** (เครื่องจำลอง): terminal สำหรับฝึกที่สร้างอะไรไม่ได้ ลบอะไรไม่ได้ และทำให้เสียเงินไม่ได้ — แต่คำสั่งคือของจริง ตรงกันทุกตัวอักษร ฝึกให้ชำนาญที่นี่ แล้ววันที่คุณได้นั่งหน้า terminal ของจริง มือของคุณจะรู้ทางเองอยู่แล้ว

PDPA ของไทย — กฎหมายความเป็นส่วนตัวที่ใกล้บ้านที่สุด

โมดูล **compliance** ส่วนที่เหลือว่าด้วยเรื่องระดับโลก แต่นี่คือกฎหมายที่หน่วยงานกำกับดูแลสามารถมาเคาะประตูถึงหน้าบ้านได้จริง PDPA ของไทย — พระราชบัญญัติคุ้มครองข้อมูลส่วนบุคคล พ.ศ. 2562 (**Personal Data Protection Act, B.E. 2562 (2019)**) — มีผลบังคับใช้เต็มรูปแบบในปี 2022 และครอบคลุมทุกองค์กรในประเทศไทยที่จัดการข้อมูลส่วนบุคคล องค์กรการกุศลในแม่สอดที่ถือทะเบียนประวัตินักเรียนอยู่ในข่ายเต็ม ๆ

ข่าวดีนั้นจริง ๆ: PDPA แทบจะเป็นสำเนาของ GDPR — กฎหมายคุ้มครองข้อมูลทั่วไปของยุโรป (General Data Protection Regulation) ผู้ร่างกฎหมายไทยใช้กฎหมายยุโรปฉบับนี้เป็นแม่แบบ ดังนั้น GDPR ได้สอนกฎหมายฉบับนี้ให้คุณไปแล้วราวร้อยละเก้าสิบ ข้อมูลส่วนบุคคลเป็นของบุคคลที่ข้อมูลนั้นพูดถึง; คุณต้องมีฐานทางกฎหมาย — บ่อยที่สุดคือ **ความยินยอม (consent)** ที่ให้โดยสมัครใจและถอนคืนได้ — จึงจะเก็บข้อมูลได้; และหลักการ **จำกัดวัตถุประสงค์ (purpose limitation)** ห้ามนำข้อมูลที่เก็บมาเพื่อวัตถุประสงค์หนึ่งไปใช้เพื่ออีกวัตถุประสงค์หนึ่ง: ทะเบียนที่เก็บมาเพื่อรับนักเรียนเข้าเรียน จะแอบกลายร่างเป็นรายชื่อส่งจดหมายระดมทุนเจียบ ๆ ไม่ได้

หน่วยงานกำกับดูแลคือ **PDPC — คณะกรรมการคุ้มครองข้อมูลส่วนบุคคล (Personal Data Protection Committee)** — ซึ่งบทลงโทษไปได้ไกลถึงความรับผิดทางอาญา ไม่ใช่เพียงค่าปรับ นาฬิกาเรื่องข้อมูลรั่วไหลก็หน้าตาคุ้นเคยเช่นกัน: เหตุรั่วไหลที่มีแนวโน้มก่อความเสี่ยงต่อบุคคล ต้องรายงานต่อ PDPC ภายใน **72 ชั่วโมง**

ประเด็นสำหรับผู้นำเสนอควรพูดออกมาให้ชัดเจน ๆ องค์กรที่ถือทะเบียนประวัตินักเรียน — ชื่อ ครอบครัว และในภูมิภาคนี้บางครั้งรวมถึงสถานะเอกสารประจำตัว ซึ่งนับว่าอ่อนไหวแทบจะที่สุดเท่าที่ข้อมูลจะเป็นได้ — **ต้องตอบกฎหมายฉบับนี้ก่อนใคร**: คำตอบต้องผ่านกรุงเทพฯ ให้ได้ก่อน แล้วค่อยไปกังวลเรื่องบริษัทเซลล์ คำถามที่ควรถามทีมของคุณ: เราถือข้อมูลส่วนบุคคลอะไรบ้าง และตอนเก็บเราแจ้งวัตถุประสงค์ไว้หรือไม่? ถ้ามีคนขอ เราลบทะเบียนประวัติของนักเรียนคนหนึ่งได้หมดจดจริงหรือไม่? ถ้าข้อมูลรั่วคืนนี้ เราบอก PDPC ภายใน 72 ชั่วโมงได้หรือไม่ว่าอะไรถูกเอาไปและเป็นของใคร? ไม่มีข้อไหนต้องอาศัยนักกฎหมายในการถาม แต่ทุกข้อต้องอาศัยผู้นำในการยื่นกรานให้ถาม

นำองค์กรฝ่าคลื่น AI

ไม่ช้าก็เร็ว — และหมูนี้มักจะเร็ว — จะมีงานออกแบบที่มี AI อยู่ข้างในมาวางบนโต๊ะคุณ หน้าที่ของคุณไม่เปลี่ยน: คุณชักใช้ ไม่ใช่ใช้ออกแบบเอง

เครื่องยนต์เบื้องหลังพีเจเออร์ AI ส่วนใหญ่คือ **LLM — Large Language Model (โมเดลภาษาขนาดใหญ่)** — และข้อเท็จจริงเชิงพาณิชย์ที่สำคัญที่สุดคือ: คุณแทบไม่มีวันได้เป็นเจ้าของโมเดลเอง คุณเช่าสิทธิ์เข้าถึงผ่าน **AI API (Application Programming Interface)** ซึ่งคิดเงินตาม **token** — ชิ้นส่วนของคำ ขนาดราวสามในสี่ของคำภาษาอังกฤษหนึ่งคำ — พีเจเออร์จึงแพงขึ้นทุกครั้งที่คำถามหรือคำตอบยาวขึ้น ขอดูการคำนวณ token ต่อผู้ใช้ ต่อเดือน; ถ้ายังไม่มีการทำ แปลว่าพีเจเออร์นั้นยังไม่มีราคา มีแต่ความหวัง

อีกทางเลือกหนึ่ง — รันโมเดลของตัวเอง — หมายถึงการเช่า **GPU (Graphics Processing Unit)** ชิปเฉพาะทางที่ AI จำเป็นต้องใช้: หายาก ราคาสูงกว่า server ธรรมดาหลายเท่าตัว และคิดเงินทุกชั่วโมงที่เปิดอยู่ ไม่ว่าจะกำลังคิดงานหรือนั่งว่าง กฎข้อแรกของสายงาน AI: **GPU ที่นั่งว่างคือความว่างที่แพงที่สุดในตึก** server ทดสอบที่ถูกล้มผลาญเงินแค่ค่ากาแฟ แต่ GPU instance ที่ถูกล้มผลาญเงินเท่าเงินเดือน คนหนึ่งคน และเมื่อมีใครพูดว่า *มา fine-tune โมเดลของเราเองกันเถอะ* จงจำลำดับที่ซื้อตรงเอาไว้: เริ่มจากการเขียน prompt อย่างพิถีพิถันบน API ที่เช่าก่อน จากนั้นจึงป้อนเอกสารของคุณให้โมเดล ณ เวลาถามคำถาม และต่อเมื่อทั้งสองทางยังไม่พอ จึงค่อย fine-tune

คำถามสามข้อจะพาคุณผ่านข้อเสนอลง AI ไปได้ทุกฉบับ หนึ่งใน: คำตอบที่ผิดมีราคาเท่าไร? LLM ตอบผิดอย่างมั่นใจจนน่าเป็นสับสนเล็กน้อย ๆ ที่ลัดให้เหลือศูนย์ไม่ได้; อัตราความผิดพลาดที่ยอมรับได้เป็นการตัดสินใจทางธุรกิจ — ซึ่งแปลว่าเป็นของคุณ **สอง: ข้อมูลอะไรออกไปจากเรา?** ทุกคำถามที่ส่งไปยัง AI API เดินทางไปถึง server ของผู้ให้บริการ; ถ้าในนั้นมีชื่อนักเรียนหรือสถานะผู้ลี้ภัย โมดูล compliance บังคับใช้เต็มรูปแบบ **สาม: ต้นทุนต่อผู้ใช้ต่อเดือนเมื่อผู้ใช้เพิ่มเป็นสิบเท่าคือเท่าไร?** ราคาแบบ token ขยายตัวเป็นเส้นตรง — โครงการนำร่องที่ \$30 อย่างสบายใจจะกลายเป็น \$300 เมื่อมันประสบความสำเร็จ โดยไม่มีส่วนลดตามปริมาณมาช่วยกอบกู้คุณ ก็คือสามเสาหลักเดิมนั่นแหละ — ความน่าเชื่อถือ ความปลอดภัย ต้นทุน — เพียงแต่สวมคำศัพท์ชุดใหม่ เทคโนโลยีเป็นของใหม่ แต่การชักใช้ไม่ใช่

Capstone — แบบฝึกหัดหนึ่งที่ประกอบทั้งหลักสูตรเข้าด้วยกัน

บัดนี้หลักสูตรปิดท้ายด้วยแบบฝึกออกแบบชิ้นเดียวที่ใช้ทุกสิ่งในหลักสูตร โจทย์: คุณเพิ่งได้รับแต่งตั้งเป็นผู้นำด้านคลาวด์ขององค์กรการกุศลด้านการศึกษาในเขตชายแดนที่แม่สอด ทรัพยากรขององค์กร — เว็บไซต์สาธารณะหนึ่งแห่ง ฐานข้อมูลทะเบียนนักเรียนราว 2,000 รายการ (ซึ่งบางรายการมีข้อมูลสถานะเอกสารประจำตัวด้วย) ศูนย์การเรียนรู้สามแห่ง และวิศวกรฝีมือดีแต่เหนื่อยล้าหนึ่งคน ผู้ถือสิทธิ์เข้าถึงระบบ production เพียงผู้เดียวและถือโทรศัพท์แวน on-call เพียงเครื่องเดียว บิลค่าคลาวด์เดือนที่แล้วมาเกือบสองเท่าของยอดปกติและไม่มีใครบอกได้ว่าเพราะอะไร วาระด้านเทคโนโลยีข้อเดียวของบอร์ด: “เราใช้ AI ช่วยติวนักเรียนได้ไหม?” ภารกิจ: 90 วันแรกของคุณในหนึ่งหน้ากระดาษ พร้อมเหตุผลรองรับลำดับก่อนหลัง

ฉบับอินเทอร์แอคทีฟให้คะแนนแผนตามรายการตรวจสอบ: สำรองของที่มีก่อนออกความเห็น; แผนที่สามารถรายการ — เครื่อง ป้ายกำกับ และเงิน; ชัยชนะเร็วหนึ่งรายการ; การกวาดตรวจความปลอดภัยและการทดสอบกู้คืน backup พร้อมระบุวันที่ชัดเจน; สัญญาณเตือนงบประมาณ; ทำที่ที่ตรงไปตรงมาต่อความเสี่ยงจากการมีวิศวกรเพียงคนเดียว; การระบุ PDPA เป็นกรอบสำหรับทะเบียนนักเรียน; การตอบคำถามเรื่อง AI ด้วยคำถาม AI สามข้อ; และการทบทวนกับบอร์ดในวันที่ 90 โดยรายงานเป็นตัวเลขต่อหน่วย
