

ຫຼັກສູດຜູ້ນຳດ້ານ Cloud (The Cloud Leader Course)

ຈາກຄວາມຮູ້ສູນ ສູ່ການນຳພາທີມວິສະວະກອນ cloud — ກະກຽມສຳລັບ Fedly Bonneau

13 ສິງຫາ 2026

ຫຼັກສູດຜູ້ນຳດ້ານ Cloud (The Cloud Leader Course)

ຈາກຄວາມຮູ້ສູນ ສູ່ການນຳພາທີມວິສະວະກອນ cloud ຢ່າງຄ່ອງແຄ້ວ.

ຫຼັກສູດນິເຮັດວຽກແນວໃດ

ທ່ານບໍ່ໄດ້ຝຶກເພື່ອ ເປັນ ວິສະວະກອນ cloud. ທ່ານກຳລັງຝຶກເພື່ອ ນຳພາ ວິສະວະກອນ cloud: ຕິດຕາມທຸກບົດ ສິນທະນາໃນຫ້ອງປະຊຸມໄດ້, ຖາມຄຳຖາມທີ່ສຳຄັນ, ຕັດສິນໃຈໄດ້ດີກ່ຽວກັບເງິນ, ຄວາມສ່ຽງ, ແລະ ຄົນ, ພ້ອມ ທັງໄດ້ຮັບຄວາມນັບຖືຈາກຜູ້ຊ່ຽວຊານ ໂດຍບໍ່ຕ້ອງທຳທຳວ່າຕົນເອງເປັນຜູ້ຊ່ຽວຊານ. ນັ້ນແມ່ນທັກສະຄົນລະຢ່າງ ທີ່ຮຽນຮູ້ໄດ້ຢ່າງແທ້ຈິງ — ແລະ ມັນຄືທັກສະທີ່ຫຼັກສູດນີ້ສອນ.

ຫຼັກສູດນີ້ມີສາມໄລຍະ. **ໄລຍະທີ 1 (ອາທິດທີ 1–8): ເວົ້າພາສານິເຮັດເປັນ.** ທ່ານຈະຮຽນຮູ້ວ່າ cloud ແທ້ຈິງແລ້ວ ແມ່ນຫຍັງ ແລະ ຄຳສັບຂອງທຸກອົງປະກອບພື້ນຖານ, ເພື່ອໃຫ້ການປະຊຸມບໍ່ຟັງຄືສຽງລົບກວນອີກຕໍ່ໄປ. **ໄລຍະທີ 2 (ອາທິດທີ 9–16): ຄິດຄືກັບຄົນໃນຫ້ອງປະຊຸມ.** ສະຖາປັດຕະຍະກຳ (architecture), ຄວາມປອດໄພ, ແລະ ເງິນ — ສາມາດສິນທະນາທີ່ທຸກທີມ cloud ມີທຸກອາທິດ, ແລະ ສາມາດບ່ອນທີ່ຜູ້ນຳຈະສ້າງຄຸນຄ່າ ຫຼື ຖືກເມີນເສີຍ. **ໄລຍະ ທີ 3 (ເດືອນທີ 5–24): ນຳພາ.** ໃບຢັ້ງຢືນ (certification), ການຈ້າງຄົນ, ການດຳເນີນກອງປະຊຸມຕັດສິນໃຈ, ແລະ ເສັ້ນທາງສອງປີແບບກົງໄປກົງມາ ສູ່ການຍືນຢູ່ໄດ້ຢ່າງໝັ້ນຄົງຕໍ່ໜ້າວິສະວະກອນອາວຸໂສ.

ກົດລະບຽບຕະຫຼອດຫຼັກສູດ: ຮຽນ **ມີລະໝັ້ງຊົ່ວໂມງ, ຫົກມື້ຕໍ່ອາທິດ** — ຄວາມສະໜ່າສະເໝີຊະນະຄວາມໝັ້ນ ໝັ້ງ. ທຸກ module ຈົບລົງດ້ວຍ **Fluency Drill** (ບົດຝຶກເວົ້າ — ປະໂຫຍກທີ່ທ່ານເວົ້າອອກສຽງຈົນເປັນ ທຳມະຊາດ) ແລະ **Milestone** (ຫຼັກໝາຍ — ຫຼັກຖານວ່າທ່ານພ້ອມກ້າວຕໍ່ໄປ). ຈົ່ງເຮັດມັນ; ການອ່ານຢ່າງດຽວ ສ້າງພຽງການຈື່ຈຳ, ບໍ່ແມ່ນຄວາມຄ່ອງແຄ້ວ. ແລະ ຕັ້ງແຕ່ອາທິດທີ 1, ໃຫ້ຮັກສາ **Decision Journal** (ປຶ້ມບັນທຶກ

ການຕັດສິນໃຈ): ທຸກຄັ້ງທີ່ຮຽນຮູ້ແນວຄິດໃໝ່, ຂຽນໜຶ່ງປະໂຫຍກວ່າມັນກະທົບເຖິງ, ຄວາມສ່ຽງ, ຫຼື ຄົນແນວໃດ — ເພາະການແປຄວາມນັ້ນເອງຄືວຽກທັງໝົດຂອງທ່ານ.

ໄລຍະທີ 0 — ຂັ້ນພື້ນຖານ (ເລີ່ມທີ່ນີ້ ຖ້າເທັກໂນໂລຊີເປັນເລື່ອງໃໝ່ສໍາລັບທ່ານ)

ສອງ module ນີ້ແມ່ນສໍາລັບຜູ້ອ່ານທີ່ບໍ່ເຄີຍເຮັດວຽກໃນສາຍເທັກໂນໂລຊີມາກ່ອນເລີຍ. ມັນຄືກັນກັບ Module 1 ແລະ 2 ຂອງຫຼັກສູດແບບໂຕ້ຕອບທີ່ b4lcilc.org. ຖ້າທ່ານຮູ້ຢູ່ແລ້ວວ່າ web server ແລະ database ແມ່ນຫຍັງ, ໃຫ້ອ່ານຜ່ານໆກໍໄດ້; ແຕ່ຖ້າຍັງບໍ່ຮູ້, ນີ້ຄືໜ້າທີ່ສໍາຄັນທີ່ສຸດໃນປຶ້ມເຫຼົ່ານີ້ — ທຸກຢ່າງທີ່ຕາມມາລ້ວນຕັ້ງຢູ່ເທິງພື້ນຖານນີ້.

Module A: ເວັບເຮັດວຽກແນວໃດ — ເກີດຫຍັງຂຶ້ນເມື່ອທ່ານພິມທີ່ຢູ່ເວັບ

ແນວຄິດຫຼັກ: ໜ້າເວັບບໍ່ແມ່ນຮູບພາບທີ່ຖືກຍົງສົງມາໃສ່ໂທລະສັບຂອງທ່ານ. ມັນຄືຊຸດ **ຄໍາສັ່ງທີ່ຂຽນໄວ້** ສົງມາຈາກຄອມພິວເຕີທີ່ຢູ່ຫ່າງໄກ, ແລ້ວ browser (ບຣາວເຊີ — ໂປຣແກຣມເປີດເວັບ) ຂອງທ່ານເອງເປັນຜູ້ປະຕິບັດຕາມ ເພື່ອສ້າງໜ້າເວັບຂຶ້ນເທິງໜ້າຈໍຂອງທ່ານ. ຮຽນຮູ້ຫຼັກຂັ້ນຕອນລະຫວ່າງການພິມທີ່ຢູ່ ກັບການເຫັນໜ້າເວັບ, ແລ້ວທຸກກອງປະຊຸມດ້ານເທັກໂນໂລຊີຕະຫຼອດອາຊີບຂອງທ່ານຈະຕິດຕາມໄດ້ງ່າຍຂຶ້ນ.

ການເດີນທາງ, ໃນຫົກຂັ້ນຕອນທີ່ທ່ານຄວນເວົ້າອອກສຽງໄດ້:

1. ທ່ານພິມຊື່ — b4lcilc.org — ໃສ່ **browser** (Chrome, Safari, Edge, Firefox).
2. **DNS**, ປຶ້ມໂທລະສັບຂອງອິນເຕີເນັດ, ປ່ຽນຊື່ນັ້ນໃຫ້ເປັນ **IP address** — ທີ່ຢູ່ຕົວເລກຂອງຄອມພິວເຕີໜ່ວຍໜຶ່ງ, ເຊັ່ນ 142.250.66.14.
3. Browser ຂອງທ່ານ — ຝ່າຍ **client** (ໄຄລເອັນ — ຜູ້ຖາມ) — ສົ່ງ **HTTP request** (ຄໍາຮ້ອງຂໍ) ໄປຫາຄອມພິວເຕີທີ່ຢູ່ນັ້ນ: “GET — ເອົາໜ້ານີ້ໃຫ້ຂ້ອຍແດ່.”
4. ຄໍາຮ້ອງຂໍນັ້ນເດີນທາງເປັນ **packet** (ແຝັກເກັດ — ຊອງຂໍ້ມູນນ້ອຍໆ) ພາຍໃຕ້ກົດຂອງ **TCP/IP**, ລະບົບທີ່ສົ່ງຂອງອິນເຕີເນັດ, ເຊິ່ງໃສ່ເລກທຸກ packet, ຍືນຢັນການສົ່ງເຖິງ, ແລະ ສົ່ງຄືນໃໝ່ທຸກອັນທີ່ຫາຍໄປ.
5. ຄອມພິວເຕີທີ່ຢູ່ຫ່າງໄກ — ຝ່າຍ **server** (ເຊີບເວີ — ຜູ້ຕອບ) — ສົ່ງ **response** (ຄໍາຕອບ) ກັບຄືນມາ: ຄໍາສັ່ງສ້າງໜ້າເວັບ, ຫຼື ຄໍາແກ້ຕົວທີ່ມີເລກກໍາກັບ ເຊັ່ນ 404 (“ບໍ່ມີໜ້ານີ້”).
6. Browser ຂອງທ່ານປະຕິບັດຕາມຄໍາສັ່ງເຫຼົ່ານັ້ນ ແລ້ວສ້າງໜ້າເວັບຂຶ້ນມາ. ຄໍາສັ່ງມາໃນສາມພາສາ: **HTML** (ໂຄງສ້າງ — ຫົວຂໍ້, ວັກ, ປຸ່ມ), **CSS** (ຮູບລັກສະນະ — ສີ, ຂະໜາດ, ການຈັດວາງ), ແລະ **JavaScript** (ພຶດຕິກຳ — ເກີດຫຍັງຂຶ້ນເມື່ອທ່ານກົດ).

Client-side ແລະ server-side — ກອບຄິດທີ່ຈັດລະບຽບທຸກຢ່າງ: ວຽກທີ່ເຮັດເທິງອຸປະກອນຂອງທ່ານ, ໃນ browser, ແມ່ນ *client-side* (ຝັ່ງ **front end**); ວຽກທີ່ເຮັດເທິງຄອມພິວເຕີທີ່ຢູ່ຫ່າງໄກ ແມ່ນ *server-side* (ຝັ່ງ **back end**). ຫຼັກຄຳຕັ້ງທີ່ຄວນຈື່ຂຶ້ນໃຈ: **ສິ່ງໃດກ່ຽວກັບຄວາມໄວ້ວາງໃຈ ຫຼື ຄວາມຈໍາ ຕ້ອງຢູ່ຝັ່ງ server; ສິ່ງໃດກ່ຽວກັບຄວາມຮູ້ສຶກວ່ອງໄວ ຕ້ອງຢູ່ຝັ່ງ client.** ກວດລະຫັດຜ່ານ, ບັນທຶກຂໍ້ມູນ, ຕັດເງິນຈາກບັດ: server. ແຕ່ມໝ້າເວັບ, ການຕອບສະໜອງທັນທີຂະນະທ່ານພິມ: client. ລະຫວ່າງສອງຝັ່ງນີ້ ມັກຈະມີ **proxy** (ພຣັອກຊີ) ຕັ້ງຢູ່ — server ຄົນກາງທີ່ກັນຕອງ, ປົກປ້ອງ, ຫຼື ກະຈາຍການສັນຈອນຂໍ້ມູນ; “load balancer” ແລະ “CDN” ໃນ module ຕ່າງໄປ ກໍ່ຄື proxy ທີ່ມີຄວາມຊໍານານສະເພາະທາງນັ້ນເອງ.

ເປັນຫຍັງ DNS ຈຶ່ງສໍາຄັນຕໍ່ຜູ້ນຳ: DNS ລົມເຫຼວແບບພິເສດ — ເມື່ອມັນເພ, ທຸກເວັບໄຊ “ຫາຍໄປ” ພ້ອມກັນໝົດທັງໆທີ່ທຸກ server ຍັງແລ່ນຢູ່. ປຶ້ມໂທລະສັບຕ່າງຫາກທີ່ລົມ, ບໍ່ແມ່ນໂທລະສັບ. ເມື່ອວິສະວະກອນເວົ້າວ່າ “it might be DNS” (ອາດຈະແມ່ນ DNS), ນັ້ນຄືສິ່ງທີ່ເຂົາໝາຍເຖິງ, ແລະ ເຫດການລົມຂອງລະບົບອິນເຕີເນັດທີ່ໂດ່ງດັງທີ່ສຸດບາງຄັ້ງ ກໍ່ແມ່ນເລື່ອງນີ້ແທ້ໆ.

ແລະ ຮູບແມ່ກະແຈ: HTTPS ຄື HTTP ບວກກັບການເຂົ້າລະຫັດ — ບົດສົນທະນາທັງໝົດຖືກເຮັດໃຫ້ອ່ານບໍ່ອອກສໍາລັບໃຜກໍ່ຕາມທີ່ຢູ່ເຄິ່ງກາງ, ເຊິ່ງນັ້ນຄືສິ່ງທີ່ຮູບແມ່ກະແຈໃນແຖບທີ່ຢູ່ຢືນໃຫ້. ກົດຍຸກໃໝ່: ທຸກເວັບໄຊ, ທຸກເວລາ.

ຄໍາສັບ:

ຄໍາສັບ	ຄວາມໝາຍ
Web browser	ໂປຣແກຣມທີ່ໄປເອົາໜ້າເວັບມາ ແລ້ວແຕ້ມມັນຂຶ້ນຈໍ: Chrome, Safari, Edge, Firefox.
URL	ທີ່ຢູ່ເຕັມທີ່ທ່ານພິມ: protocol + ຊື່ຄອມພິວເຕີ + ເອກະສານໃດທີ່ທ່ານຕ້ອງການ.
IP address	ທີ່ຢູ່ຕົວເລກຂອງຄອມພິວເຕີໃນເຄືອຂ່າຍ, ເຊັ່ນ 142.250.66.14.
DNS	ປຶ້ມໂທລະສັບຂອງອິນເຕີເນັດ: ປ່ຽນຊື່ (google.com) ໃຫ້ເປັນ IP address.
HTTP / HTTPS	ກົດການຖາມ-ຕອບທີ່ browser ແລະ server ໃຊ້ເວົ້າກັນ; ຕົວ S ເພີ່ມການເຂົ້າລະຫັດ (ຮູບແມ່ກະແຈ).
TCP/IP	ລະບົບທີ່ສົ່ງຂອງອິນເຕີເນັດ: ຂົນສົ່ງ packet ທີ່ໃສ່ເລກໄວ້ຢ່າງເຊື່ອຖືໄດ້ ລະຫວ່າງທີ່ຢູ່ຕ່າງໆ.
Client / server	ຜູ້ຖາມ ແລະ ຜູ້ຕອບ. ເປັນຄວາມສໍາພັນ, ບໍ່ແມ່ນປະເພດຂອງເຄື່ອງ.

ຄຳສັບ	ຄວາມໝາຍ
Client-side (front end)	ວຽກທີ່ເຮັດໃນ browser ຂອງຜູ້ໃຊ້: ການແຕ້ມໜ້າເວັບ, ການຕອບສະໜອງທັນທີ.
Server-side (back end)	ວຽກທີ່ເຮັດເທິງເຄື່ອງທີ່ຢູ່ຫ່າງໄກ: ຄວາມໄວ້ວາງໃຈ, ກິດລະບຽບ, ບັນທຶກຖາວອນ.
Proxy	Server ຄົນກາງທີ່ຈັດການການສັນຈອນຂໍ້ມູນແທນຝ່າຍໃດຝ່າຍໜຶ່ງ.
HTML / CSS / JavaScript	ສາມພາສາຂອງໜ້າເວັບ: ໂຄງສ້າງ / ຮູບລັກສະນະ / ພຶດຕິກຳ.

Fluency drill — ເວົ້າຈົນເປັນທຳມະຊາດ: ເວົ້າການເດີນທາງທີ່ກະຕືລືລົ້ນອອກສຽງ, ຄົນດຽວ, ຈົນກວ່າທ່ານໃຊ້ເວລາບໍ່ຮອດສາມສິບວິນາທີ ແລະ ບໍ່ຕ້ອງຢຸດຊອກຫາຄຳໃດເລີຍ. ຈາກນັ້ນເວົ້າໃຫ້ຄົນອື່ນຟັງ.

Milestone: ທ່ານສາມາດຕອບໄດ້, ຢູ່ໂຕະອາຫານ, ໂດຍບໍ່ຕ້ອງກຽມຕົວ: “ແທ້ຈິງແລ້ວເກີດຫຍັງຂຶ້ນ ເມື່ອຂ້ອຍພິມຊື່ເວັບໄຊໃສ່ໂທລະສັບ?” — ເປັນເລື່ອງເລົ່າ, ບໍ່ແມ່ນລາຍການຄຳສັບວິຊາການ.

Module B: ເວັບໄຊຖືກສ້າງ ແລະ ໂຮສແນວໃດ — ຈາກຄອມພິວເຕີໜ່ວຍດຽວ ສູ່ cloud

ແນວຄິດຫຼັກ: ເວັບໄຊຈຶ່ງທຸກອັນຖືກສ້າງເປັນ **ສາມຊັ້ນ** — front end ທີ່ຜູ້ຄົນສຳຜັດ, ຊັ້ນກາງບ່ອນທີ່ກິດລະບຽບຂອງອົງກອນເຮັດວຽກ, ແລະ ຊັ້ນຂໍ້ມູນທີ່ບໍ່ເຄີຍລືມຫຍັງ — ແລະ ຕະຫຼອດຊາວຫ້າປີທີ່ຜ່ານມາ ເກືອບທັງໝົດຖືກປະກອບຂຶ້ນຈາກໜຶ່ງໃນສອງຕະກູນເຄື່ອງມືທີ່ມີຊື່ສຽງ. ຮຽນຮູ້ສາມຊັ້ນ ແລະ ສອງຕະກູນນີ້, ແລ້ວກອງປະຊຸມກັບ vendor ຈະປ່ຽນຈາກພາສາຕ່າງດາວ ກາຍເປັນເມນູອາຫານ.

ຮ້ານອາຫານ, ເຊິ່ງທ່ານຈະໃຊ້ຕະຫຼອດຫຼັກສູດນີ້: front end (UI) ຄືຫ້ອງອາຫານ — ທຸກຢ່າງທີ່ລູກຄ້າເຫັນ, ສ້າງດ້ວຍ HTML, CSS ແລະ JavaScript. **ຊັ້ນກາງ (business logic — ເຫດຜົນທຸລະກິດ)** ຄືເຮືອນຄົວ — ໂຄງເທິງ server ທີ່ບັງຄັບໃຊ້ກິດຂອງຮ້ານ: “ນັກຮຽນຄົນໜຶ່ງລົງທະບຽນຊ້າສອງເທື່ອບໍ່ໄດ້,” “ຄຳສັ່ງຊື້ເກີນວົງເງິນຕ້ອງໄດ້ຮັບອະນຸມັດ.” **ຊັ້ນຂໍ້ມູນ (data layer)** ຄືສາງເຄື່ອງ ແລະ ປຶ້ມສູດອາຫານ — **database** (ຖານຂໍ້ມູນ), ບ່ອນເກັບຮັກສາທຸກສິ່ງທີ່ທຳມະຊາດ. ວົງການອຸດສາຫະກຳເອີ້ນຮູບແບບນີ້ວ່າ **N-tier architecture** (ສະຖາປັດຕະຍະກຳແບບຫຼາຍຊັ້ນ). ເປັນຫຍັງຕ້ອງແບ່ງຊັ້ນ? ເພື່ອໃຫ້ແຕ່ລະຊັ້ນຖືກປັບປຸງໃໝ່ໄດ້ ໂດຍບໍ່ຕ້ອງຮືສ້າງຊັ້ນອື່ນ — ແລະ ເມື່ອວິສະວະກອນເວົ້າວ່າ “ຄໍາຂວດຢູ່ຊັ້ນຂໍ້ມູນ,” ເຂົາກຳລັງຍືນແຜນທີ່ອັນນີ້ໃຫ້ທ່ານ ພ້ອມວົງມົນຂົດໄວ້ບອກຈຸດ.

ເຄື່ອງ ແລະ ໂປຣແກຣມ: server ຄືຄອມພິວເຕີຕົວຈິງ — ກ່ອງໂລຫະບໍ່ມີຈຳ ວາງຮຽງໃນຖັງ rack ຮ່ວມກັບອີກຫຼາຍຮ້ອຍໜ່ວຍໃນ **data center** (ສູນຂໍ້ມູນ), ສ້າງມາເພື່ອແລ່ນທຸກວິນາທີຂອງປີ. ເທິງມັນມີ **ໂປຣແກຣມ** ທີ່ຖືກເອີ້ນວ່າ web server ຄືກັນ ແລ່ນຢູ່, ເຮັດໜ້າທີ່ດຳເນີນບົດສົນທະນາ HTTP. ຈີຊີສອງອັນທີ່ໂດ່ງດັງທີ່ສຸດ: **IIS** (ຂອງ

Microsoft, ແລ່ນເທິງ Windows) ແລະ **Apache** (ຟຣີ, ສ່ວນຫຼາຍແລ່ນເທິງ Linux). ຢ່າລືບສົນຄໍາ: *browser ຖາມ; web server ຕອບ*. Apache ຄື **web server** — ຝ່າຍຕອບ — ບໍ່ແມ່ນ browser. **Web hosting** (ເວັບໂຮສຕິງ) ຄືການເຊົາພື້ນທີ່ເທິງ server ຂອງຄົນອື່ນ ເພື່ອໃຫ້ເຄື່ອງຂອງເຂົາຕອບແທນໃນນາມຂອງທ່ານ.

Database ແລະ SQL, ແບບຄ່ອຍເປັນຄ່ອຍໄປ: database ຄືໂປຣແກຣມທີ່ເກັບຊັ້ນ ແລະ ຕອບຄໍາຖາມກ່ຽວກັບຊັ້ນນັ້ນ — ຄືເສັ້ນໝຽນຈັດແປ້ມທີ່ລະບຽບສຸດຂີດ ແລະ ບໍ່ເຄີຍນອນ. **SQL** (ອ່ານວ່າ “ຊີໂຄວ”) ກໍຄື **ພາສາສໍາລັບຖາມຄໍາຖາມກັບ database** ນັ້ນເອງ. *Query* (ຄົວຣີ) ຄືຄໍາຖາມໜຶ່ງຊໍ້. ລອງອ່ານແຖວນີ້ — ທ່ານອ່ານໄດ້ຢູ່ແລ້ວ: `SELECT name FROM students WHERE village = 'Mae La'` — *ຈາກຕາຕະລາງ students, ເອົາຊື່ຂອງທຸກຄົນທີ່ມາຈາກ Mae La ໃຫ້ຂ້ອຍ*. ມີສອງຍີ່ຫໍ້ຄອງຕະຫຼາດ: **SQL Server** (ຂອງ Microsoft, ເສຍຄ່າ license) ແລະ **MySQL** (ຟຣີ, open source). ທັງສອງເວົ້າພາສາ SQL ຄືກັນ. ຈຸດສໍາຄັນສໍາລັບຜູ້ນໍາ: **database ຄືບ່ອນທີ່ອົງກອນອາໄສຢູ່ແທ້ໆ** — ໜ້າເວັບສ້າງຄືນໃໝ່ໄດ້, ແຕ່ຖ້າເສຍບັນທຶກການລົງທະບຽນນັກຮຽນໄປ ກໍບໍ່ມີໂຮງຮຽນອີກຕໍ່ໄປ. ນັ້ນຄືເຫດຜົນທີ່ database ໄດ້ຮັບການປົກປ້ອງທາງວິສະວະກໍາຫຼາຍທີ່ສຸດໃນອາຄານ.

ສອງຕະກຸນ:

ໜ້າທີ່ໃນເຄື່ອງ	ຕະກຸນ Microsoft	ຕະກຸນ open-source LAMP
ລະບົບປະຕິບັດການ	Windows Server	Linux
Web server (ຕອບ HTTP)	IIS	Apache
ພາສາ business logic	C# ເທິງ .NET	PHP
Database	SQL Server	MySQL
ຮູບແບບຄ່າໃຊ້ຈ່າຍ	ຄ່າ license, ຜູກກັບ vendor ດຽວ	ຟຣີ ແລະ open source; ທ່ານຈ່າຍຄ່າຄົນ
ຖິ້ນຄຸ້ນເຄີຍດັ່ງເດີມ	ທະນາຄານ, ລັດຖະບານ, ວິສາຫະກິດ	ເວັບເປີດກວ້າງ — WordPress ຢ່າງດຽວກໍ່ແລ່ນໜຶ່ງສ່ວນສາມຂອງເວັບໄຊທັງໝົດ

Open source ໝາຍຄວາມວ່າສູດຂອງ software ນັ້ນເປີດເຜີຍຕໍ່ສາທາລະນະ — ໃຜກໍສາມາດກວດເບິ່ງ, ປັບປຸງ, ແລະ ໃຊ້ໄດ້ ໂດຍບໍ່ເສຍຄ່າ license. ນີ້ຄືເຫດຜົນທີ່ອົງກອນນ້ອຍໆທຶນຈໍາກັດ ກໍສາມາດສ້າງເວັບໄຊລະດັບໂລກໄດ້ສະເໝີ ດ້ວຍລາຄາຂອງເຄື່ອງເຊົາໜ່ວຍດຽວ.

ເສັ້ນທາງສູ່ cloud — ປະໂຫຍກທີ່ປົກປ້ອງເຫຼົ່ານີ້ຢູ່ເທິງມັນ: ຫຼາຍທົດສະວັດຜ່ານມາ ທັງໝົດນີ້ແລ່ນແບບ **on-premises** (ໃນສະຖານທີ່ຂອງຕົນເອງ): ໃນຫ້ອງ server ທີ່ອົງກອນເປັນເຈົ້າຂອງ, ຊື້ໄວ້ສໍາລັບມືຖືວຽກໜັກທີ່ສຸດເທົ່າທີ່ຈົນຕະນາການໄດ້, ແຕ່ວ່າງເປົ້າໃນທຸກມື້ອື່ນ, ແລະ ຕ້ອງເປີດແອຕະຫຼອດດ້ວຍຄ່າໃຊ້ຈ່າຍຂອງອົງກອນເອງ. ຈາກນັ້ນ Amazon ແລະ Microsoft ກໍສະເໜີໃຫ້ເຊົາກໍາລັງຄອມພິວເຕີຈາກສາງຂອງເຂົາເອງເປັນຊົ່ວໂມງ. ຕົວ

stack ບໍ່ໄດ້ປ່ຽນ — Linux, Apache, IIS, SQL Server ຊຸດດຽວກັນນັ້ນ ຍັງແລ່ນຢູ່ໃນ cloud ທຸກມື້ນີ້. ສິ່ງທີ່ປ່ຽນ ຄື ເຄື່ອງຖັງຢູ່ໃສ ແລະ ໃຜເປັນເຈົ້າຂອງມັນ. ຈິ່ງປະໂຫຍກນີ້ໄວ້ໃຫ້ໝັ້ນ: Module 1 ຂອງໄລຍະທີ 1 ເປີດສາກຢູ່ຈຸດນັ້ນພໍດີ.

ຄໍາສັບ:

ຄໍາສັບ	ຄວາມໝາຍ
Server / data center	ເຄື່ອງທີ່ເປີດຕະຫຼອດເວລາ, ແລະ ອາຄານທີ່ເຕັມໄປດ້ວຍຮູ້ rack ຂອງເຄື່ອງເຫຼົ່ານັ້ນ.
Web server (ໂປຣແກຣມ)	Software ທີ່ຕອບ HTTP request: IIS (Microsoft), Apache (open source).
Web hosting	ການເຊົາພື້ນທີ່ເທິງ server ຂອງຄົນອື່ນ ສໍາລັບເວັບໄຊຂອງທ່ານ.
N-tier architecture	ການສ້າງເປັນຊັ້ນ: front end (UI), ຊັ້ນກາງ (logic), ຊັ້ນຂັ້ນລຸ່ມ.
Database	ໂປຣແກຣມທີ່ເກັບຂໍ້ມູນ ແລະ ຕອບຄໍາຖາມ — ບ່ອນທີ່ອົງກອນອາໄສຢູ່.
SQL / query	ພາສາໃກ້ຄຽງພາສາອັງກິດ ສໍາລັບຖາມຄໍາຖາມກັບ database; query ຄືຄໍາຖາມໜຶ່ງຂໍ້.
SQL Server / MySQL	Database ແບບເສຍຄ່າ license ຂອງ Microsoft / ຕົວເລືອກຍອດນິຍົມສາຍ open source. ຫຼັງສອງເວົ້າ SQL.
Microsoft stack	Windows Server + IIS + C#/.NET + SQL Server.
LAMP stack	Linux + Apache + MySQL + PHP — open source ທັງໝົດ.
Open source	ສູດເປີດເຜີຍ: ກວດເບິ່ງ, ປັບປຸງ, ແລະ ໃຊ້ໄດ້ຟຣີ.
On-premises	ເຄື່ອງທີ່ທ່ານເປັນເຈົ້າຂອງ, ໃນຫ້ອງທີ່ທ່ານເປັນເຈົ້າຂອງ — ໂລກກ່ອນຍຸກ cloud.

Fluency drill — ເວົ້າຈົນເປັນທໍາມະຊາດ: ແຍກຄວາມໝາຍ LAMP ນັບໃສ່ສີນິວມື, ເວົ້າອອກສຽງ, ຕິດຕໍ່ກັນສາມມື້. ຈາກນັ້ນອະທິບາຍຮ້ານອາຫານ — ຫ້ອງອາຫານ, ເຮືອນຄົວ, ສາງເຄື່ອງ — ໃຫ້ຄົນທີ່ບໍ່ເຄີຍໄດ້ຍິນຄໍາວ່າ server ຝັ່ງ.

Milestone: ຜູ້ຮັບເໝົາຄົນໜຶ່ງເວົ້າວ່າ: “ພວກທ່ານຢູ່ໃນລະບົບ Microsoft ແບບ on-prem ເກົ່າ — IIS ແລະ SQL Server. ພວກເຮົາເປັນຮ້ານສາຍ LAMP, ແລະ ພວກເຮົາຈະຍ້າຍພວກທ່ານຂຶ້ນ cloud.” ທ່ານສາມາດແປທຸກຄຳໃຫ້ຄະນະບໍລິຫານຂອງທ່ານຟັງໄດ້ — ແລະ ທ່ານຮູ້ວ່າຈຸດສ່ຽງທີ່ຄວນເຈາະຖາມ ຄືການຍ້າຍ database.

ໄລຍະທີ 1 — ເວົ້າພາສານິໃຫ້ເປັນ (ອາທິດທີ 1–8)

Module 1 (ອາທິດທີ 1–2): Cloud ແທ້ຈິງແລ້ວແມ່ນຫຍັງ

ແນວຄິດຫຼັກ: Cloud (ຄລາວ — ລະບົບຄອມພິວເຕີທີ່ເຊົາຜ່ານອິນເຕີເນັດ) ຄື ຄອມພິວເຕີຂອງຄົນອື່ນ ທີ່ເຊົາເປັນຊົ່ວໂມງ ແລະ ຄຸ້ມຄອງດ້ວຍ software. ກ່ອນຍຸກ cloud, ບໍລິສັດຕ້ອງຊື້ server (ເຊັບເວີ — ຄອມພິວເຕີແມ່ຂ່າຍ) ຕົວຈິງ, ເອົາໄປວາງໄວ້ໃນຫ້ອງ, ແລະ ຈ້າງຄົນມາບິວລະບັດຮັກສາ — ຊ້າ, ແພງ, ບໍ່ຍືດຍຸ່ນ. AWS, Microsoft Azure, ແລະ Google Cloud ໄດ້ສ້າງສາງຂະໜາດໃຫຍ່ທີ່ບັນຈຸຄອມພິວເຕີຫຼາຍລ້ານໜ່ວຍ (data center — ສູນຂໍ້ມູນ) ແລ້ວເປີດໃຫ້ໃຜກໍໄດ້ເຊົາສ່ວນຍ່ອຍຂອງມັນເປັນວິນາທີ, ຈາກທຸກບ່ອນ, ຜ່ານໜ້າເວັບ ຫຼື ໂຄດແຖວດຽວ. ພຽງເທົ່ານັ້ນເອງ. ທຸກຢ່າງທີ່ເຫຼືອໃນຫຼັກສູດນີ້ ແມ່ນລາຍລະອຽດທີ່ຕໍ່ຍອດຈາກແນວຄິດດຽວນີ້.

ຜູ້ໃຫ້ບໍລິການລາຍໃຫຍ່ສາມເຈົ້າ, ນິຍາມໄວ້ດັ່ງນີ້:

ຜູ້ໃຫ້ບໍລິການ	ມັນແມ່ນຫຍັງ
AWS (Amazon Web Services)	ພະແນກ cloud ຂອງ Amazon ແລະ ເປັນຜູ້ໃຫ້ບໍລິການ cloud ໃຫຍ່ທີ່ສຸດໃນໂລກ (~30% ຂອງຕະຫຼາດໂລກ). ເລີ່ມຕົ້ນປີ 2006 ເມື່ອ Amazon ເລີ່ມເປີດໃຫ້ເຊົາລະບົບຄອມພິວເຕີທີ່ຕົນເອງສ້າງໄວ້ໃຫ້ຮ້ານຄ້າຂອງຕົນເອງ. ປັດຈຸບັນມີບໍລິການຫຼາຍກວ່າ 200 ຢ່າງ — server, ບ່ອນຈັດເກັບຂໍ້ມູນ, database, AI, ແລະ ອື່ນໆ — ເຊົາໄດ້ເປັນວິນາທີ. ເມື່ອຫຼັກສູດນີ້ເວົ້າເຖິງ “cloud,” AWS ຄືຕົວຢ່າງມາດຕະຖານ, ແລະ ມັນໄດ້ເປີດ Thailand Region (ບາງກອກ) ຂອງຕົນເອງໃນເດືອນມັງກອນ 2025.
Microsoft Azure	Cloud ຂອງ Microsoft, ອັນດັບ 2 ຂອງໂລກ ແລະ ແຂງແກ່ນທີ່ສຸດພາຍໃນວິສາຫະກິດຂະໜາດໃຫຍ່ ເພາະມັນເຊື່ອມຕໍ່ຢ່າງເປັນທຳມະຊາດກັບເຄື່ອງມື Microsoft ທີ່ບໍລິສັດຕ່າງໆໃຊ້ຢູ່ແລ້ວ (Windows, Office, Active Directory). ປັດຈຸບັນກຳລັງສ້າງ datacenter region ແຫ່ງທຳອິດຂອງຕົນໃນປະເທດໄທ.

Google Cloud (GCP)

Cloud ຂອງ Google, ອັນດັບ 3 — ແຂງແກ່ນທີ່ສຸດດ້ານການວິເຄາະຂໍ້ມູນ ແລະ ເຄື່ອງມື AI. ໄດ້ໃຫ້ຄໍາຊົມເຊີຍສັນຍາລົງທຶນ 1 ຕື້ໂດລາ ໃນ data center ຂອງໄທ ແລະ cloud region ທີ່ບາງກອກ.

ບັນຊີ AWS ຟຣີຂອງທ່ານ — ຕັ້ງມັນຂຶ້ນໃນອາທິດທີ 1. ເຂົ້າໄປທີ່ <https://aws.amazon.com/free> ແລ້ວກົດ “Create a Free Account” (ໜ້າສະໝັກໂດຍກົງແມ່ນ https://signin.aws.amazon.com/signup?request_type=register). ທ່ານຈະຕ້ອງມີອີເມວ, ເບີໂທລະສັບ, ແລະ ບັດເຄຣດິດ/ເດບິດ ເພື່ອຢັ້ງຢືນຕົວຕົນ — **Free Tier** (ຂັ້ນໃຊ້ຟຣີ) ໃຫ້ໂຄດ້າລາຍເດືອນຂອງບໍລິການພື້ນຖານໂດຍບໍ່ເສຍຄ່າ (ລວມທັງ 750 ຊົ່ວໂມງ/ເດືອນຂອງ EC2 server ຂະໜາດນ້ອຍໃນປີທໍາອິດ), ແລະ ບົດຝຶກຫັດຂອງຫຼັກສູດນິຢູພາຍໃນຂອບເຂດນັ້ນທັງໝົດ. ສອງນິໄສຄວາມປອດໄພຕັ້ງແຕ່ມີທໍາອິດ: ເປີດໃຊ້ MFA (Module 6 ຈະອະທິບາຍ) ແລະ ຕັ້ງ **billing alert** (ການແຈ້ງເຕືອນຄ່າໃຊ້ຈ່າຍ) ທີ່ \$5 (Module 7 ຈະອະທິບາຍວິທີ) ເພື່ອບໍ່ໃຫ້ຖືກເຮັດເສີຍໃຈຈັກເທື່ອ. ຫຼັງສະໝັກແລ້ວທ່ານເຂົ້າສູ່ລະບົບໄດ້ທີ່ <https://console.aws.amazon.com> — “console” ກໍຄືໜ້າເວັບຄວບຄຸມຂອງ AWS ນັ້ນເອງ.

ເປັນຫຍັງບໍລິສັດຈຶ່ງໃຊ້ cloud: ຄວາມໄວ (ໄດ້ server ໃໝ່ໃນ 60 ວິນາທີ ແທນທີ່ຈະລໍ 6 ອາທິດ), ຄວາມຍືດຍຸ່ນ (elasticity — ເຊົາ 100 server ສໍາລັບມື້ຫຼຸດລາຄາໃຫຍ່ ແລ້ວສົ່ງຄືນ 90 ໜ່ວຍໃນມື້ຖັດໄປ), ບໍ່ມີຄ່າໃຊ້ຈ່າຍລ່ວງໜ້າ (ເປັນລາຍຈ່າຍດໍາເນີນງານ ແທນລາຍຈ່າຍລົງທຶນ), ແລະ ການເຂົ້າເຖິງຫົວໂລກ (ວາງແອັບຂອງທ່ານໃກ້ລູກຄ້າຢູ່ບາງກອກ, ໂຕກຽວ, ແລະ ແຟຣງເຟີດ ໂດຍບໍ່ຕ້ອງກໍ່ສ້າງຫຍັງເລີຍ).

ສາມຊັ້ນຂອງບໍລິການ — ແບບຈໍາລອງທາງຄວາມຄິດທີ່ມີປະໂຫຍດທີ່ສຸດໃນໂລກ cloud:

ຊັ້ນ	ສິ່ງທີ່ທ່ານເຊົາ	ປຽບທຽບກັບເຮືອນຄົວ	ຕົວຢ່າງ
IaaS (Infrastructure as a Service)	ຄອມພິວເຕີຕິບ, ບ່ອນຈັດເກັບ, ເຄືອຂ່າຍ — ທ່ານຄຸ້ມຄອງທຸກຢ່າງເທິງມັນເອງ	ເຊົາເຮືອນຄົວເປົ້າ: ທ່ານເອົາພໍ່ຄົວ, ສູດອາຫານ, ວັດຖຸຕິບມາເອງ	Amazon EC2, Azure Virtual Machines
PaaS (Platform as a Service)	ແພລດຟອມທີ່ມີຄົນຄຸ້ມຄອງໃຫ້ — ທ່ານເອົາມາແຕ່ແອັບພລິເຄຊັນຂອງທ່ານ	ເຊົາເຮືອນຄົວຜ່ອມພະນັກງານ: ທ່ານເອົາມາແຕ່ສູດອາຫານ	AWS Elastic Beanstalk, Azure App Service
SaaS (Software as a Service)	Software ສໍາເລັດຮູບ	ສັງຄາຫານຈາກຮ້ານອາຫານ	Gmail, Salesforce, Canva

Region ແລະ Availability Zone: Region (ພາກພື້ນ) ຄືກຸ່ມ data center ຕາມພູມສາດ (ປະເທດໄທມີ AWS Region ຂອງຕົນເອງຕັ້ງແຕ່ເດືອນມັງກອນ 2025, ຢູ່ໃນ ແລະ ອ້ອມແອ້ມບາງກອກ). **Availability Zone (AZ)** ຄື data center ໜຶ່ງແຫ່ງ ຫຼື ຫຼາຍແຫ່ງທີ່ແຍກອິດສະຫຼະຈາກກັນ ພາຍໃນ region ດຽວກັນ. ແອັບທີ່ສຳຄັນຈະແລ່ນຢູ່ຢ່າງໜ້ອຍສອງ AZ, ເພື່ອວ່າຖ້າອາຄານໜຶ່ງລົ້ມເຫຼວ ລະບົບກໍບໍ່ລົ້ມຕາມ. ເມື່ອວິສະວະກອນເວົ້າວ່າ “we’re multi-AZ,” ເຂົາໝາຍຄວາມວ່າ “data center ໜຶ່ງແຫ່ງໄໝ້ໄດ້ ແຕ່ພວກເຮົາຍັງຢູ່ອອນລາຍ.”

ຄຳສັບ:

ຄຳສັບ	ຄວາມໝາຍ
Cloud provider / hyperscaler	ບໍລິສັດທີ່ເປັນເຈົ້າຂອງ data center ຂະໜາດມະຫິມາ ແລະ ໃຫ້ເຊົາພະລັງຄອມພິວເຕີຜ່ານອິນເຕີເນັດ (AWS, Azure, Google Cloud). “Hyperscaler” = ກຸ່ມໃຫຍ່ທີ່ສຸດຈຳນວນໜ້ອຍ ທີ່ສ້າງມາເພື່ອຂະຫຍາຍໄດ້ເກືອບບໍ່ຈຳກັດ.
Data center	ສາງທີມີການຮັກສາຄວາມປອດໄພ ເຕັມໄປດ້ວຍ server ຫຼາຍພັນໜ່ວຍ ພ້ອມລະບົບໄຟຟ້າ ແລະ ຄວາມເຢັນລະດັບອຸດສາຫະກຳ — ສະຖານທີ່ຕົວຈິງທີ່ “cloud” ອາໄສຢູ່.
Region	ກຸ່ມ data center ຕາມພູມສາດ ທີ່ສະເໜີເປັນທາງເລືອກສະຖານທີ່ດຽວ, ເຊັ່ນ “Asia Pacific (Bangkok).” ທ່ານເລືອກ region ທີ່ລະບົບຂອງທ່ານຈະແລ່ນຢູ່.
Availability Zone (AZ)	Data center ໜຶ່ງແຫ່ງ ຫຼື ຫຼາຍແຫ່ງທີ່ແຍກອິດສະຫຼະພາຍໃນ region ດຽວກັນ, ມີໄຟຟ້າ ແລະ ເຄືອຂ່າຍເປັນເອກະລາດ. ການແລ່ນຢູ່ສອງ AZ ໝາຍວ່າ ອາຄານໜຶ່ງລົ້ມເຫຼວໄດ້ ແຕ່ທ່ານຍັງອອນລາຍຢູ່.
On-premises (“on-prem”)	ວິທີເກົ່າ: server ທີ່ບໍລິສັດຂອງທ່ານເປັນເຈົ້າຂອງ, ຕັ້ງຢູ່ໃນອາຄານຂອງທ່ານເອງ. ກົງກັນຂ້າມກັບ cloud.
Migration	ໂຄງການຍ້າຍລະບົບຈາກ on-prem ຂຶ້ນສູ່ cloud.
Workload	ແອັບພລິເຄຊັນ ຫຼື ລະບົບໃດໜຶ່ງທີ່ແລ່ນຢູ່ — “workload ເງິນເດືອນ,” “workload ເວັບໄຊ.” ຄຳສັບທີ່ສະດວກ: ມັນກວມເອົາທຸກຢ່າງ.
Provision	ການສ້າງ/ຕັ້ງຄ່າຊັບພະຍາກອນ cloud (server ໜຶ່ງໜ່ວຍ, database ໜຶ່ງອັນ). “Provision a server” = ເນລະມິດ server ຂຶ້ນມາ.

ຄໍາສັບ	ຄວາມໝາຍ
Scale up / scale out	ຮອງຮັບຄວາມຕ້ອງການເພີ່ມຂຶ້ນ ໂດຍເຮັດໃຫ້ເຄື່ອງໃຫຍ່ຂຶ້ນ (up) ຫຼື ເພີ່ມຈຳນວນເຄື່ອງ (out). Cloud ນິຍົມແບບ out.
Latency	ຄວາມຊັກຊ້າກ່ອນຂໍ້ມູນມາຮອດ, ວັດເປັນ millisecond. ໄລຍະທາງສ້າງ latency — ເຫດຜົນທີ່ region ບາງກອກສໍາຄັນຕໍ່ຜູ້ໃຊ້ໄທ.
Console	ໜ້າເວັບຄວບຄຸມຂອງຜູ້ໃຫ້ບໍລິການ ບ່ອນທີ່ທ່ານເບິ່ງ ແລະ ຄຸ້ມຄອງທຸກຢ່າງທີ່ທ່ານກໍາລັງເຊົາ.

ວິດີໂອສໍາລັບ module ນີ້ (ຝຣີທັງໝົດ, ລົງກວດສອບແລ້ວ):

ວິດີໂອ	ຊ່ອງ	ຄວາມຍາວ	ລິງ
What is AWS?	Amazon Web Services (official)	~2 ນາທີ	https://www.youtube.com/watch?v=a9__D53WsUs
Top 50+ AWS Services Explained in 10 Minutes	Fireship	~10 ນາທີ	https://www.youtube.com/watch?v=JlbiYCM48to
What is Microsoft Azure? An Introduction	Eye on Tech	~3 ນາທີ	https://www.youtube.com/watch?v=I9JkLhvaKA8
ຮູ້ຈັກ AWS Cloud ຕົວອະໄຈ (Thai-language intro)	Aware Corporation	ສັ້ນ	https://www.youtube.com/watch?v=nrSpZKGxXd0

Fluency drill — ເວົ້າຈົນເປັນທໍາມະຊາດ: “Is that workload on-prem or in the cloud?” (workload ນັ້ນຢູ່ on-prem ຫຼື ຢູ່ໃນ cloud?) · “Which region are we in — and are we multi-AZ?” (ພວກເຮົາຢູ່ region ໃດ — ແລະ ເປັນ multi-AZ ບໍ່?) · “Is this an IaaS approach or is there a managed service that removes the maintenance?” (ມີແມ່ນແນວທາງ IaaS ຫຼືວ່າມີ managed service ທີ່ຕັດພາລະບໍາລຸງຮັກສາອອກໄດ້?)

ບົດຝຶກຫັດ: (1) ສ້າງບັນຊີ AWS ຝຣີຂອງທ່ານທີ່ <https://aws.amazon.com/free> — ຂັ້ນຕອນສະໝັກນັ້ນເອງສອນຄໍາສັບໃຫ້ທ່ານຫຼາຍກວ່າການອ່ານໜຶ່ງບົດເຕັມໆ. (2) ເບິ່ງວິດີໂອສີອັນໃນຕາຕະລາງຂ້າງເທິງ (ລວມກັນບໍ່ຮອດ 20 ນາທີ). (3) ໃນປຶ້ມບັນທຶກຂອງທ່ານ: ຂຽນຄໍາອະທິບາຍ cloud ແບບສັ້ນກະທັດຮັດ ທີ່ທ່ານຈະເວົ້າໃຫ້ຜູ້ອໍານວຍການໂຮງຮຽນໄທຟັງໄດ້ໃນລົມຫາຍໃຈດຽວ.

Milestone: ທ່ານສາມາດອະທິບາຍ IaaS vs PaaS vs SaaS ດ້ວຍການປຽບທຽບຂອງທ່ານເອງ, ແລະ ອະທິບາຍໄດ້ວ່າເປັນຫຍັງການທີ່ AWS ເປີດ region ບາງກອກ ຈຶ່ງສໍາຄັນຕໍ່ທະນາຄານໄທ (ຄໍາຕອບ: latency + data residency — ເບິ່ງ Module 6).

Module 2 (ອາທິດທີ 3–4): ອົງປະກອບພື້ນຖານ — compute, storage, ແລະ database

Compute — ເຄື່ອງຈັກທີ່ແລ່ນໂຄດ. Virtual machine (VM/instance) (ເຄື່ອງຈັກເສມືອນ) ຄືສ່ວນແບ່ງໜຶ່ງຂອງ server ຕົວຈິງ ທີ່ປະພຶດຕົວຄືກັບຄອມພິວເຕີເຕັມໜ່ວຍ; ທ່ານເລືອກຂະໜາດ (CPU/RAM) ແລະ ຈ່າຍຕາມວິນາທີທີ່ມັນແລ່ນ. **Container** (ຄອນເທນເນີ) ຄືວິທີຫຸ້ມຫໍ່ແອັບພລິເຄຊັນໜຶ່ງອັນ ທີ່ເບົາກວ່າ ໄວກວ່າ ເພື່ອໃຫ້ມັນແລ່ນໄດ້ຄືກັນທຸກປະການໃນທຸກບ່ອນ; **Docker** ໃຊ້ຫຸ້ມຫໍ່ພວກມັນ ແລະ **Kubernetes** (K8s) ເປັນຜູ້ຄວບຄຸມວົງດົນຕີຂອງພວກມັນເປັນຝູງ — ເມື່ອໄດ້ຍິນ “K8s,” ໃຫ້ຄິດວ່າ “ລະບົບທີ່ແລ່ນ ແລະ ຮັກສາ container ຫຼາຍຮ້ອຍອັນຂອງເຮົາໂດຍອັດຕະໂນມັດ.” **Serverless** (ເຊີເວີເລສ — ແລ່ນໂຄດໂດຍບໍ່ຕ້ອງຄຸ້ມຄອງ server) (AWS Lambda) ໝາຍວ່າທ່ານອັບໂຫຼດພຽງ function ຂອງໂຄດ; cloud ຈະແລ່ນມັນເມື່ອຖືກກະຕຸ້ນ ແລະ ທ່ານຈ່າຍຕໍ່ຄັງທີ່ເອີ້ນໃຊ້ — ບໍ່ມີ server ໃຫ້ຄຸ້ມຄອງເລີຍ. ຮູບແບບທີ່ຄວນສັງເກດ: VM → container → serverless ຄືແຖບເລື່ອນຈາກ “ຄວບຄຸມໄດ້ຫຼາຍ, ບໍາລຸງຮັກສາຫຼາຍ” ໄປສູ່ “ຄວບຄຸມໄດ້ໜ້ອຍ, ບໍາລຸງຮັກສາເກືອບສູນ.” ທີມທີ່ດີເລືອກຕາມແຕ່ລະ workload, ບໍ່ແມ່ນຕາມແຟຊັນ.

Storage — ສາມຮູບແບບ. Object storage (ບ່ອນເກັບແບບວັດຖຸ) (Amazon S3): ຖັງບັນຈຸໄຟລ໌ທີ່ບໍ່ມີກິນ — ຮູບພາບ, ວິດີໂອ, ຂໍ້ມູນສໍາຮອງ, ຊຸດຂໍ້ມູນ; ລາຄາຖືກ, ທົນທານລະດັບ eleven nines, ຄໍາຕອບມາດຕະຖານຂອງຄໍາຖາມ “ພວກເຮົາຈະເອົາໄຟລ໌ໄປໄວ້ໃສ?” **Block storage** (EBS): ຮາດດິສເສມືອນທີ່ຕິດກັບ VM. **File storage** (EFS): ໂດຣຟ໌ແບ່ງປັນທີ່ຫຼາຍເຄື່ອງ mount ພ້ອມກັນໄດ້. ຈາກນັ້ນແມ່ນ **tiers** (ຂັ້ນລາຄາ): hot (ເຂົ້າເຖິງເລື້ອຍໆ, ແພງ) vs cold/archive (Glacier — ຖືກ, ຊ້າ) — ການຍ້າຍຂໍ້ມູນເກົ່າໄປຂັ້ນ cold ແມ່ນໜຶ່ງໃນໄຊຊະນະດ້ານຕົ້ນທຶນທີ່ງ່າຍທີ່ສຸດ ທີ່ທີມໃດກໍເຮັດໄດ້.

Database — ສອງຕະກູນ. Relational/SQL (MySQL, PostgreSQL; ແບບ managed ຄື Amazon RDS/Aurora): ຂໍ້ມູນໃນຕາຕະລາງທີ່ມີໂຄງສ້າງເຄັງຄັດ; ຄໍາມາດຕະຖານສໍາລັບເງິນ, ຄໍາສັ່ງຊື້, ຜູ້ໃຊ້ — ທຸກຢ່າງທີ່ຄວາມຖືກຕ້ອງເປັນສິ່ງສັກສິດ. **NoSQL** (DynamoDB, MongoDB): ຍືດຍຸ່ນ, ຂະຫຍາຍໄດ້ມະຫາສານ; ຄໍາມາດຕະຖານສໍາລັບຂໍ້ມູນຂະໜາດໃຫຍ່ ໄວ ຮູບຮ່າງງ່າຍກວ່າ (session, catalog, feed). “Managed database” ໝາຍວ່າຜູ້ໃຫ້ບໍລິການຈັດການ backup, patch, ແລະ failover ໃຫ້ — ທີມຄວນມີເຫດຜົນໜັກແໜ້ນແທ້ໆ ຈຶ່ງຈະແລ່ນ database ຂອງຕົນເອງ.

ຄໍາສັບ:

ຄໍາສັບ	ຄວາມໝາຍ
Instance	Virtual machine (VM) ທີ່ເຊົ້າໜຶ່ງໜ່ວຍ. “Spin up an instance” = ເປີດ server ຂຶ້ນມາ.

ຄຳສັບ	ຄວາມໝາຍ
Instance type / size	ສະເປັກທີ່ທ່ານເລືອກໃຫ້ມັນ — ມີຈັກ CPU, ໜ່ວຍຄວາມຈຳເທົ່າໃດ. ຂະໜາດໃຫຍ່ຂຶ້ນ = ລາຄາຕໍ່ຊົ່ວໂມງແພງຂຶ້ນ.
Container	ຫົບຫໍ້ນ້ຳໜັກເບົາ ບັນຈຸແອັບພລິເຄຊັນໜຶ່ງອັນ ພ້ອມທຸກສິ່ງທີ່ມັນຕ້ອງການ, ເພື່ອໃຫ້ມັນແລ່ນຄືກັນທຸກປະການໃນທຸກເຄື່ອງ. ໄວກວ່າ ແລະ ຖືກກວ່າ VM ເຕັມໜ່ວຍ.
Docker	ເຄື່ອງມືມາດຕະຖານສໍາລັບສ້າງ ແລະ ແລ່ນ container.
Kubernetes (K8s)	ຕົວປະສານງານ (orchestrator) ທີ່ແລ່ນ ແລະ ຮັກສາຝູງ container ໂດຍອັດຕະໂນມັດ — ປິດເປີດອັນທີ່ລົ້ມຄືນໃໝ່, ເພີ່ມຈຳນວນເມື່ອວຽກໜັກ. “K8s” ຄືຊື່ຫຼິ້ນຂອງວົງການ.
Cluster / node	Cluster ຄືກຸ່ມເຄື່ອງທີ່ເຮັດວຽກຮ່ວມກັນເປັນລະບົບດຽວ; ແຕ່ລະເຄື່ອງໃນນັ້ນຄື node.
Serverless	ການແລ່ນໂຄດໂດຍບໍ່ຕ້ອງຄຸ້ມຄອງ server ໃດເລີຍ — cloud ແລ່ນ function ຂອງທ່ານເມື່ອຖືກກະຕຸ້ນ ແລະ ຄິດເງິນຕໍ່ການແລ່ນ.
Lambda / function	ຜະລິດຕະພັນ serverless ຂອງ AWS; “function” ຄືໂຄດຊິ້ນນ້ອຍທີ່ມັນແລ່ນ.
S3 / bucket	Object storage ຂອງ AWS ສໍາລັບໄຟລີ້; bucket ຄືພາຊະນະບັນຈຸໄຟລີ້ທີ່ມີຊື່ໜຶ່ງອັນ. ຄໍາຕອບມາດຕະຖານຂອງ “ພວກເຮົາຈະເອົາໄຟລີ້ໄປໄວ້ໃສ?”
EBS volume	ຮາດດິສເສມືອນທີ່ຕິດຢູ່ກັບ instance.
Durability	ໂອກາດທີ່ຂໍ້ມູນທີ່ເກັບໄວ້ຈະຢູ່ລອດ. “Eleven nines” ຂອງ S3 (99.999999999%) ໝາຍວ່າການສູນເສຍເກືອບຈະບໍ່ເກີດຂຶ້ນເລີຍ.
Storage tier	ຂັ້ນລາຄາ/ຄວາມໄວຂອງຂໍ້ມູນ: hot (ທັນທີ, ແພງ) → cold/archive (Glacier — ຖືກ, ໃຊ້ເວລານາທີຫາຊົ່ວໂມງໃນການດຶງຄືນ).
RDS	ບໍລິການ relational database ແບບ managed ຂອງ AWS — AWS ຈັດການ backup, patch, ແລະ failover

ຄໍາສັບ	ຄວາມໝາຍ
	ໃຫ້ທ່ານ.
SQL vs NoSQL	SQL: ຕາຕະລາງເຄັງຄັດ, ເໝາະສົມທີ່ສຸດສໍາລັບເງິນ ແລະ ຄໍາສັ່ງຊື້. NoSQL: ຍືດຢູ່ນ ແລະ ຂະຫຍາຍໄດ້ ມະຫາສານ, ສໍາລັບ session, catalog, feed.
Backup / snapshot	ສໍາເນົາຂໍ້ມູນທີ່ບັນທຶກໄວ້ (snapshot ຄືສໍາເນົາຂອງຈຸດ ເວລາໃດໜຶ່ງ ຂອງດິສ ຫຼື database) ທີ່ທ່ານກູ້ຄືນຈາກ ມັນໄດ້.
Failover	ການສະຫຼັບໄປໃຊ້ສໍາເນົາສໍາຮອງໂດຍອັດຕະໂນມັດ ເມື່ອ ຕົວຫຼັກລົ້ມເຫຼວ — ເຫດຜົນທີ່ managed database ຜ່ານຄືນຮ້າຍໆໄປໄດ້.

ວິດີໂອສໍາລັບ module ນີ້:

ວິດີໂອ	ຊ່ອງ	ຄວາມຍາວ	ລິງ
Getting Started with EC2	AWS Developers (official)	27 ນາທີ	https://www.youtube.com/watch?v=nJ-djerESW0
Introduction to Amazon S3	Amazon Web Services (official)	~5 ນາທີ	https://www.youtube.com/watch?v=ecv-19sYL3w
Docker in 100 Seconds	Fireship	2 ນາທີ	https://www.youtube.com/watch?v=Gjnup-PuquQ
Kubernetes explained in 15 mins	TechWorld with Nana	~16 ນາທີ	https://www.youtube.com/watch?v=VnvRFRk_51k
Serverless Computing in 100 Seconds	Fireship	~2 ນາທີ	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?” (ອັນນີ້ຄວນແລ່ນເທິງ VM, container, ຫຼື serverless — ແລະ ຕົ້ນທຶນດ້ານປະຕິບັດງານ ຂອງແຕ່ລະທາງເລືອກແມ່ນເທົ່າໃດ?) · “Is that data hot or can it go to a cheaper tier?” (ຂໍ້ມູນນັ້ນເປັນ hot

ຫຼືວ່າຍ້າຍໄປ tier ທີ່ຖືກກວ່າໄດ້?) · “Why are we self-managing that database instead of using RDS?” (ເປັນຫຍັງພວກເຮົາຈຶ່ງຖືກຮຸ້ມຄອງ database ນັ້ນເອງ ແທນທີ່ຈະໃຊ້ RDS?)

ບົດຝຶກຫັດ: (1) ໃນບັນຊີຟຣີຂອງທ່ານ: ເປີດ EC2 instance ຂະໜາດນ້ອຍທີ່ສຸດ, ແລ້ວປິດຖິ້ມ (terminate). ຮັບໂຫຼດໄຟລ໌ໜຶ່ງໄຟລ໌ຂຶ້ນ S3. ດຽວນີ້ທ່ານໄດ້ໃຊ້ IaaS ດ້ວຍຕົນເອງແລ້ວ. (2) ຂໍໃຫ້ AI assistant ທົດສອບທ່ານ: “Give me 10 scenarios; I’ll answer VM, container, or serverless, and you grade me.” (ໃຫ້ 10 ສະຖານະການ; ຂ້ອຍຈະຕອບ VM, container, ຫຼື serverless, ແລ້ວເຈົ້າໃຫ້ຄະແນນຂ້ອຍ.)

Milestone: ເມື່ອໄດ້ຍິນແອັບປະກອນອັນໃດກໍຕາມ ທີ່ອະທິບາຍໃນປະໂຫຍກດຽວ (“ເວັບໄຊທີ່ມີກະຽນສັງກະຫານທ່ຽງ”), ທ່ານສາມາດບອກຊື່ສ່ວນປະກອບ compute, storage, ແລະ database ຂອງມັນອອກສຽງໄດ້ພາຍໃນ 60 ວິນາທີ.

Module 3 (ອາທິດທີ 5–6): ເຄືອຂ່າຍ, ແລະ ວິທີທີ່ຊື່ນສ່ວນຕ່າງໆ ສື່ສານກັນ

VPC — ບ້ານສ່ວນຕົວຂອງທ່ານ. Virtual Private Cloud (ເຄືອຂ່າຍສ່ວນຕົວເສມືອນໃນ cloud) ຄືເຂດຮີວກັນຂອງທ່ານເອງ ພາຍໃນເຄືອຂ່າຍຂອງຜູ້ໃຫ້ບໍລິການ. ພາຍໃນມັນມີ **subnet** (ເຄືອຂ່າຍຍ່ອຍ) — ແບບ public (ເຂົ້າເຖິງໄດ້ຈາກອິນເຕີເນັດ, ເຊັ່ນ web server) ແລະ ແບບ private (ເຂົ້າເຖິງບໍ່ໄດ້, ເຊັ່ນ database). ປະໂຫຍກຄວາມປອດໄພທີ່ໄດ້ຍິນເລື້ອຍທີ່ສຸດ: “the database sits in a private subnet.” (database ຢູ່ໃນ private subnet.) ຖ້າທ່ານເຂົ້າໃຈວ່າເປັນຫຍັງ — ຜູ້ໃຈມຕິແຕະຕ້ອງສິ່ງທີ່ບໍ່ມີເສັ້ນທາງຈາກອິນເຕີເນັດບໍ່ໄດ້ — ທ່ານກໍເຂົ້າໃຈເຄິ່ງໜຶ່ງຂອງຄວາມປອດໄພເຄືອຂ່າຍແລ້ວ.

ຊັ້ນຈະລາຈອນ: load balancer (ຕົວແບ່ງເບົາການຈະລາຈອນ) ກະຈາຍ traffic ຂາເຂົ້າໄປຫາຫຼາຍ server (ແລະ ຄ່ອຍໆເອົາເຄື່ອງທີ່ບໍ່ສະບາຍອອກ); **DNS** (Route 53) ແປຊີຢ່າງ yourcompany.com ໃຫ້ເປັນທີ່ຢູ່; **CDN** (CloudFront) ເກັບສຳເນົາເນື້ອຫາຂອງທ່ານໄວ້ໃນຫຼາຍຮ້ອຍເມືອງ ເພື່ອໃຫ້ໂຫຼດໄວທຸກບ່ອນ; **API** ຄືປະຕູທີ່ software ອັນໜຶ່ງເປີດໃຫ້ software ອື່ນ — ເມື່ອວິສະວະກອນເວົ້າວ່າ “we’ll expose an API,” ເຂົາໝາຍວ່າ “ພວກເຮົາຈະໃຫ້ software ອື່ນມີວິທີໃຊ້ຂອງເຮົາແບບຄວບຄຸມໄດ້”; **API gateway** ຄືໂຕະຕ້ອນຮັບທີ່ຖືກຮຸ້ມຄອງປະຕູເຫຼົ່ານັ້ນ.

ການເຊື່ອມສອງໂລກ: VPN (ອຸໂມງເຂົ້າລະຫັດຜ່ານອິນເຕີເນັດ) ຫຼື **Direct Connect** (ສາຍສ່ວນຕົວທາງກາຍະພາບ) ເຊື່ອມຕ້ອງການ/ລະບົບ on-prem ຂອງບໍລິສັດ ເຂົ້າກັບ cloud ຂອງມັນ. **Hybrid cloud** = ແລ່ນທັງ on-prem ແລະ cloud ພ້ອມກັນ (ວິສາຫະກິດໄຫລສ່ວນໃຫຍ່ເປັນແບບນີ້); **multi-cloud** = ໃຊ້ຜູ້ໃຫ້ບໍລິການຫຼາຍກວ່າໜຶ່ງລາຍ.

ຄຳສັບ:

ຄຳສັບ	ຄວາມໝາຍ
VPC (Virtual Private Cloud)	ເຂດສ່ວນຕົວ ມີຮີວກັນ ຂອງທ່ານເອງ ພາຍໃນເຄືອຂ່າຍຂອງຜູ້ໃຫ້ບໍລິການ ບ່ອນທີ່ລະບົບຂອງທ່ານອາໄສຢູ່.

ຄຳສັບ	ຄວາມໝາຍ
Subnet (public / private)	ສ່ວນຍ່ອຍຂອງ VPC. Public subnet ເຂົ້າເຖິງໄດ້ຈາກອິນເຕີເນັດ (web server); private subnet ເຂົ້າເຖິງບໍ່ໄດ້ (database).
IP address	ທີ່ຢູ່ຕົວເລກຂອງເຄື່ອງໃນເຄືອຂ່າຍ — ວິທີທີ່ຄອມພິວເຕີຊອກຫາກັນ.
Firewall / security group	ບັນຊີກົດລະບຽບທີ່ບອກວ່າ traffic ໃດເຂົ້າຫາເຄື່ອງໄດ້ (“web traffic ເຂົ້າໄດ້, ອື່ນໆບໍ່ໄດ້”). Security group ຄື firewall ລະດັບ server ຂອງ AWS.
Load balancer	ຜູ້ຄວບຄຸມຈະລາຈອນ ທີ່ກະຈາຍຄຳຮ້ອງຂໍຂາເຂົ້າໄປຫາຫຼາຍ server ແລະ ຢຸດສົ່ງໄປຫາເຄື່ອງທີ່ບໍ່ແຂງແຮງ.
DNS	ປຶ້ມໂທລະສັບຂອງອິນເຕີເນັດ — ແປ yourcompany.com ໃຫ້ເປັນ IP address. ບໍລິການ DNS ຂອງ AWS ຄື Route 53.
CDN (Content Delivery Network)	ສຳເນົາເນື້ອຫາຂອງທ່ານ ເກັບໄວ້ໃນຫຼາຍຮ້ອຍເມືອງ ເພື່ອໃຫ້ໂຫຼດໄວທຸກບ່ອນ. ຂອງ AWS ຄື CloudFront.
Edge location	ຈຸດ CDN ລະດັບເມືອງເຫຼົ່ານັ້ນຈຸດໜຶ່ງ — “the edge” ໝາຍເຖິງໃກ້ຜູ້ໃຊ້.
API	ປະຕູທີ່ software ອື່ນໜຶ່ງເປີດໃຫ້ software ອື່ນ. “We’ll expose an API” = ພວກເຮົາຈະໃຫ້ software ອື່ນມີວິທີໃຊ້ຂອງເຮົາແບບຄວບຄຸມໄດ້.
API gateway	ໂຕະຕ້ອນຮັບທີ່ຖືກຄອງປະຕູເຫຼົ່ານັ້ນ — ການຢັ້ງຢືນຕົວຕົນ, ຈຳກັດອັດຕາ, ບັນທຶກ log.
VPN	ອຸໂມງເຂົ້າລະຫັດຜ່ານອິນເຕີເນັດສາທາລະນະ ເຊື່ອມສອງເຄືອຂ່າຍ (ເຊັ່ນ ຫ້ອງການຂອງທ່ານ ກັບ VPC ຂອງທ່ານ).
Direct Connect	ສາຍສ່ວນຕົວທາງກາຍະພາບໄປສູ່ cloud — ໄວກວ່າ ແລະ ນຶ່ງກວ່າ VPN, ແຕ່ມີລາຄາ.
Hybrid / multi-cloud	Hybrid: ແລ່ນ on-prem ແລະ cloud ຮ່ວມກັນ (ວິສາຫະກິດໄຫສ່ວນໃຫຍ່). Multi-cloud: ໃຊ້ຜູ້ໃຫ້ບໍລິການຫຼາຍກວ່າໜຶ່ງລາຍ.

ຄຳສັບ	ຄວາມໝາຍ
Ingress / egress	Traffic ທີ່ເຂົ້າ / ອອກຈາກ cloud. Egress ເສຍເງິນ — ຈິດຄຳນີໄວ້ສຳລັບ Module 7.

ວິດີໂອສຳລັບ module ນີ້: AWS Networking Basics — VPC & Subnets for Beginners — KodeKloud — https://www.youtube.com/watch?v=QM63dyA_4Pc (ເບິ່ງເຄິ່ງທຳອິດຕອນນີ້; ກັບມາເບິ່ງສ່ວນທີ່ເຫຼືອຫຼັງ Module 5).

Fluency drill: “Is the database in a private subnet?” (database ຢູ່ໃນ private subnet ບໍ່?) · “What happens when one web server dies — is the load balancer health-checking?” (ຈະເກີດຫຍັງຂຶ້ນເມື່ອ web server ໜຶ່ງໝວຍຕາຍ — load balancer ມີ health check ຢູ່ບໍ່?) · “Are we exposing that as an API or is it internal only?” (ພວກເຮົາເປີດອັນນັ້ນເປັນ API ຫຼືວ່າໃຊ້ພາຍໃນເທົ່ານັ້ນ?)

ບົດຝຶກຫັດ: (1) ໃຫ້ AI ນຳພາທ່ານແຕ້ມ, ໃສ່ເຈ້ຍ, ເຄືອຂ່າຍຂອງແອັບສົງຄາຫານ: ຜູ້ໃຊ້ → CDN → load balancer → web server (public subnet) → database (private subnet). ແຕ້ມມັນສາມເທື່ອ ຈົນທ່ານແຕ້ມໄດ້ຈາກຄວາມຈຳ. (2) ຊອກຫາແຜນວາດສະຖາປັດຕະຍະກຳຂອງບໍລິສັດທ່ານ ຫຼື ຕົວຢ່າງໃດກໍໄດ້ ແລ້ວວິນິດໄສທຸກກ່ອງທີ່ຕອນນີ້ທ່ານບອກຊື່ໄດ້.

Milestone: ທ່ານສາມາດແຕ້ມແຜນວາດສາມຊັ້ນ (three-tier) ມາດຕະຖານນັ້ນເທິງກະດານຂາວ ແລະ ບັນຍາຍເສັ້ນທາງການຄລິກຂອງລູກຄ້າຄົນໜຶ່ງ ຜ່ານມັນໄດ້.

Module 4 (ອາທິດທີ 7–8): ທີ່ມ cloud ເຮັດວຽກແນວໃດ — DevOps, IaC, ແລະ ການອ່ານບັນຍາກາດຫ້ອງປະຊຸມ

DevOps (ວັດທະນະທຳການເຮັດວຽກຮ່ວມກັນລະຫວ່າງຝ່າຍພັດທະນາ ແລະ ຝ່າຍປະຕິບັດງານ) ຄື ວັດທະນະທຳຂອງການລວມ “ຄົນຂຽນ software” (Dev) ແລະ “ຄົນແລ່ນມັນ” (Ops) ເຂົ້າເປັນທີມດຽວ ທີ່ສົ່ງມອບການປ່ຽນແປງນ້ອຍໆ ເລື້ອຍໆ ຢ່າງປອດໄພ, ໂດຍມີລະບົບອັດຕະໂນມັດເປັນຜູ້ຮັບພາລະໜັກ. ຫົວໃຈເຕັ້ນຂອງມັນຄື **CI/CD pipeline** (ສາຍພານອັດຕະໂນມັດສຳລັບສ້າງ-ທົດສອບ-ນຳໃຊ້ໂຄດ): ທຸກການປ່ຽນແປງໂຄດ ຖືກ build, ທົດສອບ, ແລະ deploy ໂດຍອັດຕະໂນມັດ (Continuous Integration / Continuous Delivery). ເມື່ອທີ່ມເວົ້າວ່າ “it’s in the pipeline,” ເຂົາໝາຍວ່າ ຫຸ່ນຍົນກຳລັງທົດສອບ ແລະ ສົ່ງມອບມັນຢູ່.

Infrastructure as Code (IaC) (ໂຄງລ່າງເປັນໂຄດ) — ແນວຄິດທີ່ປ່ຽນທຸກຢ່າງ: ແທນທີ່ຈະຄລິກໄປມາໃນ console ເພື່ອສ້າງ server, ວິສະວະກອນຂຽນ *ໄຟລ໌ຂໍ້ຄວາມ* ທີ່ປະກາດໂຄງລ່າງ (“server ສອງໜ່ວຍ, load balancer ໜຶ່ງອັນ, database ໜຶ່ງອັນ, ກົດ firewall ໜຶ່ງອັນ”), ແລ້ວເຄື່ອງມື (**Terraform**, CloudFormation) ເຮັດໃຫ້ຄວາມເປັນຈິງກົງກັບໄຟລ໌. ເຫດຜົນທີ່ຜູ້ນຳຕ້ອງສົນໃຈ: ໄຟລ໌ເຫຼົ່ານັ້ນຢູ່ໃນ **Git** (ລະບົບຄວບຄຸມເວີຊັນ), ດັ່ງນັ້ນທຸກການປ່ຽນແປງໂຄງລ່າງ ຖືກກວດທານ, ຍ້ອນກັບໄດ້, ແລະ ກວດສອບຄືນໄດ້ — ຄວາມແຕກຕ່າງລະຫວ່າງໂຮງຊ່າງນ້ອຍ ກັບ ໂຮງງານ.

ພິທີກຳທີ່ທ່ານຈະໄດ້ນັ່ງຮ່ວມ: **stand-up** (ປະຊຸມຍືນປະຈຳວັນ 15 ນາທີ: ເຮັດຫຍັງແລ້ວ, ຈະເຮັດຫຍັງຕໍ່, ຕິດຫຍັງຢູ່ — ຝັ່ງຫາສິ່ງກົດຂວາງ (blocker); ການເອົາມັນອອກຄືວຽກຂອງທ່ານ), **sprint** (ໝວຍວຽກຫຼວງແຜນໄວ້ 1–2 ອາທິດ), **retro** (ປະຊຸມທົບທວນວ່າຈະປັບປຸງຫຍັງ), **post-mortem/incident review** (ຫຼັງລະບົບລົ້ມ: ການວິເຄາະແບບ ບໍ່ຫາຄົນຜິດ ວ່າຫຍັງຝັ່ງ ແລະ ຫຍັງຈະປ້ອງກັນມັນ — ສຸຂະພາບຂອງທີມສະແດງອອກຢູ່ບ່ອນວ່າ ການທົບທວນເຫຼົ່ານີ້ຊີ້ສັດພຽງໃດ), **on-call** (ຮອບວຽນຂອງຄົນທີ່ຖືກປຸກຕອນຕີ 3; ຖ້າ on-call ທໍລະມານ, ຄົນເກັ່ງທີ່ສຸດຂອງທ່ານຈະລາອອກ — ຖາມເລື່ອງນີ້ທຸກເດືອນ).

ຕົວຊີ້ວັດສຸຂະພາບທີ່ສໍາຄັນ: **uptime/availability** (“three nines” = 99.9% ≈ ລົ້ມ 8.8 ຊົ່ວໂມງ/ປີ; ແຕ່ລະ nine ທີ່ເພີ່ມ ຄູນຕົ້ນທຶນຂຶ້ນ), **SLA** (ຄໍາສັນຍາຕໍ່ລູກຄ້າ, ຜ່ອມຄ່າປັບ), **SLO** (ເປົ້າໝາຍພາຍໃນ), **MTTR** (ຄວາມໄວໃນການກູ້ຄືນ — ທີມທີ່ເຕີບໃຫຍ່ແລ້ວ ປັບປຸງການກູ້ຄືນ, ບໍ່ແມ່ນຄວາມຝັນລົມໆແລ້ງໆເລື່ອງບໍ່ມີວັນລົມ), **deployment frequency** (ທີມສຸຂະພາບດີ ສົ່ງມອບການປ່ຽນແປງນ້ອຍໆເລື້ອຍໆ; ຄວາມຢ້ານທີ່ຈະ deploy ຄື ສັນຍານບໍ່ດີ).

ຄໍາສັບ:

ຄໍາສັບ	ຄວາມໝາຍ
DevOps	ວັດທະນະທຳຂອງທີມດຽວ ທີ່ທັງສ້າງ ແລະ ແລ່ນ software ຂອງຕົນ, ສົ່ງມອບການປ່ຽນແປງນ້ອຍໆເລື້ອຍໆ ດ້ວຍລະບົບອັດຕະໂນມັດ.
CI/CD pipeline	ສາຍພານອັດຕະໂນມັດທີ່ build, ທົດສອບ, ແລະ deploy ທຸກການປ່ຽນແປງໂຄດ (Continuous Integration / Continuous Delivery).
Deploy / rollback	Deploy: ປ່ອຍການປ່ຽນແປງເຂົ້າສູ່ລະບົບຈິງ. Rollback: ຍົກເລີກມັນຢ່າງໄວ ເມື່ອມັນເຮັດວຽກຜິດປົກກະຕິ.
Git / repository / pull request	Git: ລະບົບຄວບຄຸມເວີຊັນ ບັນທຶກທຸກການປ່ຽນແປງ. Repository (“repo”): ບ້ານຂອງໂຄດໂຄງການໜຶ່ງ. Pull request (PR): ການປ່ຽນແປງທີ່ສະເໜີ ໃຫ້ວິສະວະກອນອື່ນກວດຫານກ່ອນຮວມເຂົ້າ.
IaC (Infrastructure as Code)	ການປະກາດໂຄງລ່າງໃນໄຟລີ້ຂໍ້ຄວາມ ທີ່ເຄື່ອງມືປ່ຽນເປັນຄວາມຈິງ — ກວດຫານໄດ້, ຍ້ອນກັບໄດ້, ກວດສອບຄືນໄດ້.
Terraform	ເຄື່ອງມື IaC ທີ່ນິຍົມທີ່ສຸດ (ໃຊ້ໄດ້ທຸກ cloud). ຂອງ AWS ເອີ້ນ CloudFormation.

ຄຳສັບ	ຄວາມໝາຍ
Staging vs production (“prod”)	Staging: ສະບັບຊ້ອມໃຫຍ່ຂອງລະບົບ. Prod: ອັນຈິງທີ່ລູກຄ້າສຳຜັດ. “Broke prod” = ມືຮ້າຍ.
Stand-up	ປະຊຸມສັ້ນ 15 ນາທີປະຈຳວັນ: ເຮັດແລ້ວ / ຈະເຮັດ / ຕິດຢູ່. ຝັ່ງຫາ blocker — ການເອົາມັນອອກຄິວຮອງທ່ານ.
Sprint / backlog	Sprint: ໜ່ວຍວຽກຫຼວງແຜນໄວ້ 1–2 ອາທິດ. Backlog: ບັນຊີວຽກຮຽງລຳດັບ ທີ່ປ່ອນເຂົ້າ sprint.
Retro	ປະຊຸມທ້າຍ sprint ວ່າຈະເຮັດວຽກໃຫ້ດີຂຶ້ນແນວໃດເທື່ອໜ້າ.
Post-mortem	ການທົບທວນແບບບໍ່ຫາຄົນຜິດ ຫຼັງລະບົບລົ້ມ: ຫຍັງພັງ, ເປັນຫຍັງ, ຫຍັງຈະປ້ອງກັນບໍ່ໃຫ້ຊ້າ.
On-call	ຮອບວຽນຂອງຄົນທີ່ຕອບສັ່ນຍານເຕືອນຕອນທີ 3. ຖ້າ on-call ທໍລະມານ, ຄົນເກັ່ງທີ່ສຸດຂອງທ່ານຈະໄປ.
Incident / sev-1	ການຂັດຂ້ອງທີ່ບໍ່ໄດ້ວາງແຜນ; “sev-1” (severity one) = ຮ້າຍແຮງທີ່ສຸດ, ທຸກຄົນລົງມືຊ່ວຍ.
SLA / SLO	SLA: ຄຳສັນຍາຕໍ່ລູກຄ້າຜ່ານຄ່າປັບ (ເຊັ່ນ uptime 99.9%). SLO: ເປົ້າໝາຍພາຍໃນທີ່ເຄັ່ງກວ່າ ເພື່ອປົກປ້ອງ SLA.
MTTR	Mean Time To Recovery — ຄວາມໄວທີ່ທ່ານກັບມາອອນລາຍຫຼັງລົ້ມເຫຼວ. ທີ່ມີເຕີບໃຫຍ່ແລ້ວ ປັບປຸງອັນນີ້, ບໍ່ແມ່ນຄວາມຝັນວ່າຈະບໍລິມະຈິກເທື່ອ.
Monitoring / observability	ການເຝົ້າເບິ່ງສຸຂະພາບແບບສົດ ຜ່ານ log (ບັນທຶກເຫດການ), metric (ຕົວເລກຕາມເວລາ), ແລະ alert (ການແຈ້ງເຕືອນອັດຕະໂນມັດ ເມື່ອເກີນຂີດກຳນົດ).

ວິດີໂອສຳລັບ module ນີ້:

ວິດີໂອ	ຊ່ອງ	ຄວາມຍາວ	ລິງ
What is DevOps? REALLY understand	TechWorld with Nana	~15 ນາທີ	https://www.youtube.com/watch?v=0yWAtQ6wYNM

ວິດີໂອ	ຊ່ອງ	ຄວາມຍາວ	ລິງ
it			
DevOps CI/CD Explained in 100 Seconds	Fireship	2 ນາທີ	https://www.youtube.com/watch?v=scEDHsr3APg
Terraform explained in 15 mins	TechWorld with Nana	18 ນາທີ	https://www.youtube.com/watch?v=I5k1ai_GBDE

Fluency drill: “Is that change through the pipeline or was it manual?” (ການປ່ຽນແປງນັ້ນຜ່ານ pipeline ຫຼືເຮັດດ້ວຍມື?) · “What did the post-mortem conclude, and what’s the prevention item?” (post-mortem ສະຫຼຸບວ່າແນວໃດ, ແລະ ມາດຕະການປ້ອງກັນແມ່ນຫຍັງ?) · “What’s our MTTR trending like?” (ແນວໂນ້ມ MTTR ຂອງເຮົາເປັນແນວໃດ?) · “Is this in Terraform, or did someone click it into existence?” (ອັນນີ້ຢູ່ໃນ Terraform ຫຼືວ່າມີຄົນຄລິກສ້າງມັນຂຶ້ນມາ?) (ແບບຫຼັງເອີ້ນວ່າ “ClickOps,” ເວົ້າຜ່ອມສີໝ້າບໍ່ພໍໃຈ).

ບົດຝຶກຫັດ: (1) ເບິ່ງການສືບທະນາບົດລາຍງານ post-mortem ຂອງເຫດການຈິງໜຶ່ງອັນ (ຫຼາຍອັນເປັນສາທາລະນະ — ບົດລາຍງານການລົ້ມຂອງ Cloudflare ແລະ AWS ມີຊື່ສຽງ) ແລ້ວສະຫຼຸບໃສ່ປຶ້ມບັນທຶກຂອງທ່ານໃນຫ້າປະໂຫຍກ. (2) ນັ່ງຮ່ວມ (ຫຼືເບິ່ງບັນທຶກຂອງ) stand-up ໃດກໍໄດ້ ແລ້ວຂຽນ blocker ສາມອັນທີ່ທ່ານໄດ້ຍິນ.

Milestone — ຈົບໄລຍະທີ 1: ທ່ານສາມາດນັ່ງຝັ່ງກອງປະຊຸມວາງແຜນດ້ານເຕັກນິກ 30 ນາທີ ແລະ ຕິດຕາມໄດ້ $\geq 80\%$ ຂອງມັນ, ແລະ ປຶ້ມບັນທຶກຂອງທ່ານພິສູດມັນ: ບັນທຶກຈາກກອງປະຊຸມຈິງ ຫຼື ຈຳລອງໜຶ່ງຄັ້ງ ໂດຍທຸກຄຳຫຍໍ້ຖືກຂະຫຍາຍຄວາມຢ່າງຖືກຕ້ອງ. **ນິຍົງເປັນເວລາທີ່ທ່ານຄວນນັດສອບເສັງ AWS Cloud Practitioner (CLF-C02)** — ການກຽມຕົວແບບຕັ້ງໃຈ 2–4 ອາທິດ ຕໍ່ຍອດຈາກ module ເຫຼົ່ານີ້ ຄືມາດຕະຖານທີ່ເຜີຍແຜ່ກັນທົ່ວໄປ, ແລະ ການສອບຜ່ານມັນຄືຫຼັກຖານພາຍນອກອັນທຳອິດຂອງທ່ານ.

ໄລຍະທີ 2 — ຄິດຄືກັບຄືນໃນຫ້ອງປະຊຸມ (ອາທິດທີ 9–16)

Module 5 (ອາທິດທີ 9–10): ສະຖາປັດຕະຍະກຳ — ຕັດສິນແບບອອກແບບ ໂດຍບໍ່ຕ້ອງແຕ້ມມັນເອງ

ບົດບາດຂອງທ່ານໃນ design review ບໍ່ແມ່ນການອອກແບບ — ແມ່ນການສອບຖາມ. ກອບແນວຄິດທີ່ທັງອຸດສາຫະກຳໃຊ້ ຄື **Well-Architected Framework** ຂອງ AWS, ຫົກເລື່ອງທີ່ທຸກການອອກແບບຄວນຕອບໄດ້: operational excellence, **security**, **reliability**, performance efficiency, **cost optimization**, sustainability. ຮຽນສາມອັນທີ່ເປັນຕົວໜ້າໃຫ້ເລິກ; ເງິນ ແລະ ຄວາມສ່ຽງອາໄສຢູ່ບ່ອນນັ້ນ.

Reliability (ຄວາມເຊື່ອຖືໄດ້), ເວົ້າແບບງ່າຍໆ: ທຸກຢ່າງລົ້ມເຫຼວໃນທີ່ສຸດ, ດັ່ງນັ້ນລະບົບທີ່ດີຈຶ່ງສົມມຸດໄວ້ກ່ອນ.
Redundancy (ຄວາມຊ້ຳຊ້ອນ — ບໍ່ມີຈຸດລົ້ມເຫຼວດ່ຽວ; ຂອງສຳຄັນມີສອງຊຸດ), multi-AZ (ຜ່ານການສູນເສຍ data center ໄປໄດ້), auto-scaling (ເພີ່ມ/ຫຼຸດເຄື່ອງອັດຕະໂນມັດຕາມຄວາມຕ້ອງການ), backup ທີ່ຖືກທົດສອບແລ້ວ (backup ທີ່ບໍ່ເຄີຍທົດສອບ ຄືຄວາມຫວັງ, ບໍ່ແມ່ນແຜນ), RTO/RPO (Recovery Time Objective: ເຮົາລົ້ມໄດ້ດົນປານໃດ; Recovery Point Objective: ເຮົາເສຍຂໍ້ມູນໄດ້ຫຼາຍປານໃດ — ສອງຕົວເລກນີ້ ຄືບົດສົນທະນາເລື່ອງ disaster recovery, ແລະ ພວກມັນເປັນການຕັດສິນໃຈທາງທຸລະກິດ, ໝາຍຄວາມວ່າ ເປັນຂອງທ່ານ).

ສາມຫຼ່ຽມການແລກປ່ຽນ: ໄວ, ຖືກ, ທົນທານ — ເລືອກໄດ້ສອງ. ທຸກການໂຕ້ຖຽງດ້ານສະຖາປັດຕະຍະກຳທີ່ທ່ານຈະໄດ້ເປັນກຳມະການ ຫຍໍ້ລົງມາເປັນຄຳຖາມວ່າ ທຸລະກິດຕ້ອງການຢູ່ຈຸດໃດຂອງສາມຫຼ່ຽມນັ້ນ ສຳລັບ workload ນີ້. ລະບົບຊຳລະເງິນ ກັບ ເວັບການຕະຫຼາດ ບໍ່ສົມຄວນໄດ້ຄຳຕອບດຽວກັນ.

ເຈັດຄຳຖາມຂອງຜູ້ນຳ — ທ່ອງຈຳໄວ້; ພວກມັນເຮັດໃຫ້ທ່ານໝ້າຍຢ່າເກງໃນທຸກ design review:

1. “What happens when this component fails?” (ຈະເກີດຫຍັງຂຶ້ນເມື່ອສ່ວນປະກອບນີ້ລົ້ມເຫຼວ?) (ຄຳວ່າ “ເມື່ອ” ມີສະເໝີ)
2. “What are the RTO and RPO, and who signed off on them?” (RTO ແລະ RPO ແມ່ນເທົ່າໃດ, ແລະ ໃຜເປັນຜູ້ອະນຸມັດ?)
3. “Where does this run — single AZ, multi-AZ, multi-region — and why?” (ອັນນີ້ແລ່ນຢູ່ໃສ — single AZ, multi-AZ, multi-region — ແລະ ເປັນຫຍັງ?)
4. “What will this cost per month at 10× today’s load?” (ອັນນີ້ຈະເສຍເດືອນລະເທົ່າໃດ ຖ້າວຽກໜັກ 10 ເທົ່າຂອງມື້ນີ້?)
5. “What’s the simplest version that meets the requirement?” (ສະບັບທີ່ງ່າຍທີ່ສຸດ ທີ່ຕອບສະໜອງເງື່ອນໄຂ ແມ່ນຫຍັງ?) (over-engineering ຄືພະຍາດຂອງມືໃໝ່; ຄວາມງ່າຍຄືຄຸນນະທຳຂອງມືອາວຸໂສ)
6. “What are we locked into, and what would leaving cost?” (ພວກເຮົາຖືກຜູກຕິດກັບຫຍັງ, ແລະ ການອອກຈາກມັນຈະເສຍເທົ່າໃດ?) (vendor lock-in ຄືລາຄາອັນໜຶ່ງ, ບາງເທື່ອກໍ່ຄຸ້ມທີ່ຈະຈ່າຍ — ແຕ່ຕ້ອງຮູ້ຕົວ)
7. “Who else has built this before — are we inventing or assembling?” (ໃຜເຄີຍສ້າງອັນນີ້ມາກ່ອນແລ້ວ — ພວກເຮົາກຳລັງປະດິດ ຫຼື ປະກອບ?) (ນິຍົມຮູບແບບທີ່ໜ້າເບື້ອແຕ່ພິສູດແລ້ວ)

Fluency drill: ຝຶກຖາມຄຳຖາມ 1, 4, ແລະ 5 ດ້ວຍນ້ຳສຽງອົບອຸ່ນ — ພວກມັນຈະຖືກຮັບຮູ້ວ່າເປັນປັນຍາ ຫຼື ເປັນການໂຈມຕີ ຂຶ້ນກັບວິທີເວົ້າລ້ວນໆ. “Help me understand what happens if the cache goes down” (ຊ່ວຍໃຫ້ຂ້ອຍເຂົ້າໃຈແດ່ ວ່າຈະເກີດຫຍັງຂຶ້ນຖ້າ cache ລົ້ມ) ດຶກວ່າ “did you think about failure?” (ເຈົ້າໄດ້ຄິດເລື່ອງຄວາມລົ້ມເຫຼວບໍ່?)

ບົດຝຶກຫັດ: (1) ເອົາສະຖາປັດຕະຍະກຳຕົວຢ່າງສາມອັນ (ຂໍໃຫ້ AI ສ້າງ: ເວັບ e-commerce, backend ຂອງ ແອັບມືຖື, ແຜນລາຍງານວິເຄາະຂໍ້ມູນ) ແລ້ວໃຊ້ເຈັດຄຳຖາມທັງໝົດກັບແຕ່ລະອັນ, ຂຽນຄຳຕອບທີ່ທ່ານຄາດຫວັງ. (2) ອ່ານບົດສະຫຼຸບໜຶ່ງໜ້າຂອງເສົາຫຼັກ Well-Architected ທີ່ <https://aws.amazon.com/architecture/well-architected/>.

ວິດີໂອສຳລັບ module ນີ້:

ວິດີໂອ	ຊ່ອງ	ຄວາມຍາວ	ລິງ
The Five Pillars of the AWS Well-Architected Framework	Amazon Web Services (official; ເສົາຫຼັກທີ່ຫົກ, Sustainability, ຖືກ ເພີ່ມພາຍຫຼັງ)	~4 ນາທີ	https://www.youtube.com/watch?v=KvEDbPmha6o
What is the AWS Well-Architected Framework?	Tech With Lucy (ex-AWS)	~10 ນາທີ	https://www.youtube.com/watch?v=MpDJ6TCWKjk

Milestone: ໃນ design review ຈຳລອງ (ເຮັດກັບ AI ທີ່ສະແດງບົດເປັນວິສະວະກອນ), ທ່ານຖາມຄຳຖາມທີ່ມີເນື້ອໃນຫ້າຂໍ້ ແລະ ສະຫຼຸບຈຸດອ່ອນທີ່ສຸດຂອງການອອກແບບໄດ້ຢ່າງຖືກຕ້ອງໃນຕອນທ້າຍ.

Module 6 (ອາທິດທີ 11–12): ຄວາມປອດໄພ ແລະ ການປະຕິບັດຕາມກົດໝາຍ — ບົດສົນທະນາທີ່ອາດເຮັດໃຫ້ບໍລິສັດຈົບຊີວິດ

Shared responsibility model (ຮູບແບບຄວາມຮັບຜິດຊອບຮ່ວມ) — ສິ່ງທຳອິດທີ່ຕ້ອງເຂົ້າໃຈ: ຜູ້ໃຫ້ບໍລິການຮັກສາຄວາມປອດໄພຂອງຕົວ cloud ເອງ (ອາຄານ, ຮາດແວ, hypervisor); **ທ່ານ** ຮັກສາຄວາມປອດໄພຂອງສິ່ງທີ່ທ່ານເອົາໄປໃສ່ ໃນ ມັນ (ຂໍ້ມູນຂອງທ່ານ, ກົດການເຂົ້າເຖິງ, ການຕັ້ງຄ່າ). ການຮົວໄຫຼສ່ວນໃຫຍ່ ແມ່ນການຕັ້ງຄ່າຜິດຂອງລູກຄ້າ — S3 bucket ທີ່ຖືກປະໄວ້ເປັນ public, access key ທີ່ຮົວ — ບໍ່ແມ່ນຄວາມລົ້ມເຫຼວຂອງຜູ້ໃຫ້ບໍລິການ. ດັ່ງນັ້ນ “AWS ປອດໄພ” ກັບ “ພວກເຮົາປອດໄພເທິງ AWS” ຄືສອງປະໂຫຍກທີ່ຕ່າງກັນ.

ຄຳສັບຂອງການປ້ອງກັນ: IAM (Identity and Access Management — ໃຜເຮັດຫຍັງໄດ້; ສິ່ງທີ່ຖືກກວດສອບຫຼາຍທີ່ສຸດໃນ cloud), **least privilege** (ທຸກຄົນໄດ້ສິດເຂົ້າເຖິງຂັ້ນຕ່ຳສຸດທີ່ຈຳເປັນ — ກົດທອງ), **MFA** (ການຢັ້ງຢືນຕົວຕົນຫຼາຍປັດໄຈ — ປັດໄຈທີ່ສອງໃນທຸກການເຂົ້າລະບົບຂອງມະນຸດ — ຕໍ່ລອງບໍ່ໄດ້), **encryption at rest and in transit** (ການເຂົ້າລະຫັດຂໍ້ມູນທັງຕອນເກັບ ແລະ ຕອນເດີນທາງ — ມາດຕະຖານຂັ້ນຕ່ຳ, ເປີດສະເໝີ), **root account** (ກະແຈແຜ່ — ເກັບມ້ຽນໄວ້, ບໍ່ໃຊ້ປະຈຳວັນ), **secrets management** (ລະຫັດຜ່ານ/key ເກັບໃນຜູ້ນຳລະໄພ, ບໍ່ຂຽນໃສ່ໂຄດເດັດຂາດ), **zero trust** (ກວດສອບທຸກຢ່າງ, ບໍ່ໄວ້ໃຈຕໍ່ແໜ້ງເຄືອຂ່າຍໃດໂດຍປະລິຍາຍ), **penetration test** (ຈ້າງນັກໂຈມຕີມາພິສູດແນວປ້ອງກັນຂອງທ່ານ), **ransomware**

(ເຫດຜົນທີ່ backup ແບບ offline ທີ່ຖືກທົດສອບແລ້ວ ເປັນມາດຕະການຄວາມປອດໄພ, ບໍ່ແມ່ນພຽງມາດຕະການ ops).

PDPA — ກົດໝາຍຂໍ້ມູນຂອງໄທ, ແລະ ຂໍ້ໄດ້ປຽບທາງຕະຫຼາດຂອງທ່ານ. Personal Data Protection Act ຄື GDPR ຂອງໄທ: ຕ້ອງມີການຍິນຍອມກ່ອນເກັບຂໍ້ມູນສ່ວນບຸກຄົນ, ໜ້າທີ່ ແຈ້ງເຫດຮົ່ວໄຫຼພາຍໃນ 72 ຊົ່ວໂມງ, ຕ້ອງມີ Data Protection Officer ສໍາລັບຜູ້ປະມວນຜົນຂະໜາດໃຫຍ່, ແລະ ກົດເລື່ອງການສົ່ງຂໍ້ມູນຂ້າມຊາຍແດນ. ການປັບຄືນໃຊ້ກາຍເປັນເລື່ອງຈິງໃນປີ 2025 — ຄ່າປັບຄືນທໍາອິດເກີນ THB 21.5M, ລວມທັງໂຮງໝໍທີ່ຖືກປັບ ຍ້ອນຄຸ້ມຄອງ vendor ບໍ່ດີ. ສອງຜົນຕາມມາສໍາລັບທ່ານ: (1) ທຸກວິສາຫະກິດໄທ ຕອນນີ້ມີບັນຫາການປະຕິບັດຕາມກົດໝາຍ ທີ່ບໍ່ສິດຂອງທ່ານສາມາດຮັບເງິນເພື່ອຈັດການໃຫ້; (2) **data residency** (ການເກັບຂໍ້ມູນໄວ້ໃນປະເທດ) — ການຮັກສາຂໍ້ມູນໄທໄວ້ໃນແຜ່ນດິນໄທ — ຄືແຮງຂັບຕົວຈິງ ທີ່ດັນ workload ເຂົ້າສູ່ region ບາງກອກ. ວັກນິຄົເຄິ່ງໜຶ່ງຂອງບົດຂາຍຂອງທ່ານໃນປະເທດໄທ; ຈິມັນໃຫ້ຂຶ້ນໃຈ.

ຄໍາຖາມດ້ານຄວາມປອດໄພຂອງຜູ້ນໍາ: “Who has access to production, and when did we last review the list?” (ໃຜເຂົ້າເຖິງ production ໄດ້, ແລະ ພວກເຮົາທົບທວນບັນຊີນັ້ນຄັ້ງສຸດທ້າຍເມື່ອໃດ?) · “Are we alerted on unusual access, or would we find out from the news?” (ພວກເຮົາໄດ້ຮັບການແຈ້ງເຕືອນເມື່ອມີການເຂົ້າເຖິງຜິດປົກກະຕິ ຫຼືວ່າຈະຮູ້ຈາກຂ່າວ?) · “When did we last restore a backup?” (ພວກເຮົາ ກູ້ຄືນ backup ຄັ້ງສຸດທ້າຍເມື່ອໃດ?) (ບໍ່ແມ່ນ “ພວກເຮົາມີ backup ບໍ່”) · “If we lost this dataset, is it a PDPA-notifiable breach — and could we notify within 72 hours?” (ຖ້າພວກເຮົາເສຍຊຸດຂໍ້ມູນນີ້, ມັນເປັນເຫດຮົ່ວໄຫຼທີ່ຕ້ອງແຈ້ງຕາມ PDPA ບໍ່ — ແລະ ພວກເຮົາແຈ້ງພາຍໃນ 72 ຊົ່ວໂມງໄດ້ບໍ່?) · “What did the last pen test find, and what’s still open?” (pen test ຄັ້ງລ່າສຸດພົບຫຍັງ, ແລະ ຫຍັງຍັງບໍ່ຖືກແກ້?)

ຄໍາສັບ:

ຄໍາສັບ	ຄວາມໝາຍ
IAM (Identity and Access Management)	ລະບົບຄວບຄຸມວ່າໃຜ (ທັງຄົນ ແລະ software) ເຮັດຫຍັງໄດ້ໃນ cloud ຂອງທ່ານ — ສິ່ງທີ່ຖືກກວດສອບຫຼາຍທີ່ສຸດໃນນັ້ນ.
Role / policy	Policy ຄືຊຸດສິດອະນຸຍາດທີ່ຂຽນໄວ້; role ຄືມັດຂອງ policy ທີ່ຄົນ ຫຼື ໂປຣແກຣມສວມໃສ່ (assume) ໄດ້.
Least privilege	ກົດທອງ: ທຸກຄົນໄດ້ສິດເຂົ້າເຖິງຂັ້ນຕໍ່າສຸດທີ່ວຽກຂອງເຂົາຕ້ອງການ, ບໍ່ຫຼາຍກວ່ານັ້ນ.
MFA (multi-factor authentication)	ຫຼັກຖານທີ່ສອງ (ລະຫັດຈາກໂທລະສັບ, hardware key) ນອກເໜືອຈາກລະຫັດຜ່ານ. ຕໍ່ລອງບໍ່ໄດ້ ໃນທຸກການເຂົ້າລະບົບຂອງມະນຸດ.

ຄຳສັບ	ຄວາມໝາຍ
Encryption at rest / in transit	ຂໍ້ມູນຖືກເຂົ້າລະຫັດ ຕອນເກັບໄວ້ / ຕອນເດີນທາງໃນເຄືອຂ່າຍ. ທັງສອງເປີດສະເໝີ; ມາດຕະຖານຂັ້ນຕ່ຳ.
KMS (key management)	ບໍລິການທີ່ຖື ແລະ ໝູນວຽນກະແຈເຂົ້າລະຫັດ.
Secrets	ລະຫັດຜ່ານ, API key, token — ເກັບໃນຖັງລະໂພ, ບໍ່ຂຽນລົງໃນໂຄດເດັດຂາດ.
Root account	ກະແຈແມ່ຂອງບັນຊີ cloud ທັງໝົດ. ເກັບມ້ຽນຜ່ານ MFA, ບໍ່ໃຊ້ວຽກປະຈຳວັນ.
Zero trust	ທ່າທີຄວາມປອດໄພ ທີ່ກວດສອບທຸກຄຳຮ້ອງຂໍ ແລະ ບໍ່ໄວ້ໃຈຕໍ່ແໜ້ງເຄືອຂ່າຍໃດໂດຍປະລິຍາຍ.
Vulnerability / patching	ຈຸດອ່ອນທີ່ຮູ້ຈັກແລ້ວໃນ software / ການຕິດຕັ້ງຕົວແກ້ໄຂ. ລະບົບທີ່ບໍ່ໄດ້ patch ຄືທາງເຂົ້າຂອງການເຈາະລະບົບສ່ວນໃຫຍ່.
Pen test (penetration test)	ນັກໂຈມຕີມີຈັນຍາບັນທຶກຖືກຈ້າງ ມາພິສູດວ່າແນວປ້ອງກັນຂອງທ່ານພົບບ່ອນໃດ ກ່ອນນັກໂຈມຕີຕົວຈິງຈະພົບ.
Ransomware	ການໂຈມຕີທີ່ເຂົ້າລະຫັດຂໍ້ມູນຂອງທ່ານເພື່ອຮຽກຄ່າໄຖ່ — ເຫດຜົນທີ່ backup ແບບ offline ທີ່ຖືກທົດສອບແລ້ວເປັນມາດຕະການ ຄວາມປອດໄພ.
SOC 2 / ISO 27001	ກາໝາຍການກວດສອບຄວາມປອດໄພຈາກພາກສ່ວນອິດສະຫຼະ ທີ່ລູກຄ້າວິສາຫະກິດຮຽກຮ້ອງກ່ອນເຊັນສັນຍາ.
PDPA	Personal Data Protection Act ຂອງໄທ — ການຍິນຍອມ, ການແຈ້ງເຫດຮົ່ວໄຫຼພາຍໃນ 72 ຊົ່ວໂມງ, ກົດການໂອນຂໍ້ມູນຂ້າມຊາຍແດນ. ບັງຄັບໃຊ້ດ້ວຍຄ່າປັບຈິງຕັ້ງແຕ່ປີ 2025.
Data residency	ການຮັກສາຂໍ້ມູນໄວ້ທາງກາຍະພາບພາຍໃນຊາຍແດນຂອງປະເທດ — ເຫດຜົນຫຼັກທີ່ workload ຂອງໄທຍ້າຍໄປ region ບາງກອກ.
DPO / breach notification	Data Protection Officer (ບັງຄັບສຳລັບຜູ້ປະມວນຜົນຂະໜາດໃຫຍ່) / ໜ້າທີ່ລາຍງານເຫດຂໍ້ມູນສ່ວນບຸກຄົນຮົ່ວໄຫຼ ພາຍໃນ 72 ຊົ່ວໂມງ.

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

Fluency drill: “Is that bucket public? Why?” (bucket ນັ້ນເປັນ public ບໍ່? ເປັນຫຍັງ?) · “Least privilege — does the intern really need prod access?” (least privilege — ນັກຝຶກງານຈໍາເປັນຕ້ອງເຂົ້າ prod ແທ້ ບໍ່?) · “Where does the PDPA line sit in this design — what’s personal data here and where does it physically live?” (ເສັ້ນ PDPA ຢູ່ບ່ອນໃດໃນການອອກແບບນີ້ — ຫຍັງແມ່ນຂໍ້ມູນສ່ວນບຸກຄົນຢູ່ນີ້ ແລະ ມັນຢູ່ ບ່ອນໃດທາງກາຍະພາບ?)

ບົດຝຶກຫັດ: (1) ອ່ານເລື່ອງການຮົວໄຫຼຍ້ອນຕັ້ງຄ່າ cloud ຜິດ ທີ່ມີຊື່ສຽງໜຶ່ງເລື່ອງ (Capital One ປີ 2019 ຄືກໍາລະນີຄລາສສິກ) ແລ້ວຂຽນສະບັບສາມປະໂຫຍກໃສ່ປຶ້ມບັນທຶກຂອງທ່ານ. (2) ໃຫ້ AI ຈຳລອງລູກຄ້າຖາມວ່າ “why should I trust you with my data?” (ເປັນຫຍັງຂ້ອຍຄວນໄວ້ໃຈເຈົ້າກັບຂໍ້ມູນຂອງຂ້ອຍ?) ແລ້ວຝຶກຕອບ ໂດຍໃຊ້ພາສາ shared responsibility + PDPA.

Milestone: ທ່ານສາມາດອະທິບາຍ shared responsibility model ແລະ ຄວາມໝາຍພາກປະຕິບັດຂອງ PDPA ໃຫ້ເຈົ້າຂອງທຸລະກິດໄທທີ່ບໍ່ແມ່ນສາຍເຕັກນິກຟັງໄດ້ ພາຍໃນສາມນາທີ — ເພາະຄໍາອະທິບາຍນັ້ນ ຄື ກອງປະຊຸມຂາຍຂອງ MSP ນັ້ນເອງ.

Module 7 (ອາທິດທີ 13–14): ເລື່ອງເງິນ — ເສດຖະສາດ cloud ແລະ FinOps

ນິຕິບັນຍັດທີ່ຜູ້ນຳທີ່ບໍ່ແມ່ນວິສະວະກອນ ສ້າງຄຸນຄ່າໄດ້ໄວທີ່ສຸດ, ເພາະວິສະວະກອນສ່ວນໃຫຍ່ບໍ່ເຄີຍຖືກສອນເລື່ອງ ນີ້ ແລະ ການສູນເສຍນັ້ນມະຫາສານ — ການສຳຫຼວດພົບຢ່າງສະໝໍ່າສະເໝີວ່າ ປະມານໜຶ່ງສ່ວນສາມຂອງ ລາຍຈ່າຍ cloud ຖືກເສຍໄປລ້າງ.

ມິເຕີແລ່ນແນວໃດ: compute ຄິດເງິນຕໍ່ວິນາທີທີ່ມັນ *ເປີດຢູ່* (ບໍ່ແມ່ນຕໍ່ການໃຊ້ — server ທີ່ວາງເປົ້າກໍຄິດເງິນເຕັມ; “ພວກເຮົາລືມປິດມັນ” ຄືການສູນເສຍແບບຄລາສສິກ), storage ຄິດຕໍ່ GB-ເດືອນ, ແລະ — ກັບດັກທີ່ໂດ່ງດັງ — egress (ຂໍ້ມູນຂາອອກ): ຂໍ້ມູນທີ່ໄຫຼ *ອອກ* ຈາກ cloud ເສຍເງິນ ໃນຂະນະທີ່ຂໍ້ມູນຂາເຂົ້າຟຣີ. ໃບບິນ egress ກ້ອນໃຫຍ່ ເຮັດເສີຍໃຈທຸກຄົນເທື່ອໜຶ່ງ; ຫຼັງ module ນີ້, ຈະບໍ່ແມ່ນທ່ານ.

ເມນູລາຄາ: on-demand (ລາຄາເຕັມ, ຍືດຍຸ່ນເຕັມທີ່) · reserved instances / savings plans (ຜູກມັດ 1–3 ປີ ແລກກັບສ່ວນຫຼຸດ 30–70% — ຄັນໂຍກໃຫຍ່ທີ່ສຸດອັນດຽວ ສໍາລັບ workload ທີ່ນຶງ) · spot instances (ຫຼຸດເຖິງ 90% ສໍາລັບວຽກທີ່ຂັດຈັງຫວະໄດ້ ເຊັ່ນ batch job) · rightsizing (server ສ່ວນໃຫຍ່ຖືກຈັດຂະໜາດເກີນຕົວ; ການຫົດມັນລົງຄືເງິນຟຣີ) · storage tiering (Module 2) · auto-scaling ໃນຖານະເຄື່ອງມືປະຢັດ (ຈະຈ່າຍຄ່າຄວາມສາມາດຕອນທ່ຽງຄືນ ໃນລາຄາຕອນທ່ຽງວັນເຮັດຫຍັງ?).

FinOps (ວິໄນການບໍລິຫານການເງິນ cloud) ຄືການປະຕິບັດເພື່ອເຮັດໃຫ້ລາຍຈ່າຍ cloud ເບິ່ງເຫັນໄດ້, ຈັດສັນໄດ້, ແລະ ຖືກປັບປຸງຢ່າງຕໍ່ເນື່ອງ: tagging ທຸກຊັບພະຍາກອນ ດ້ວຍເຈົ້າຂອງ ແລະ ໂຄງການຂອງມັນ (ລາຍຈ່າຍທີ່ບໍ່ມີ tag ຄືລາຍຈ່າຍທີ່ບໍ່ມີໃຜຮັບຜິດຊອບ), showback/chargeback (ສະແດງໃບບິນຂອງແຕ່ລະທີມໃຫ້ເຂົາເຫັນ — ພຶດຕິກຳປ່ຽນຫັນທີ), ງົບປະມານຜ່ອມ alert (ຢ່າຄືນພົບການໃຊ້ເກີນຈາກໃບແຈ້ງໜີເດັດຂາດ), ແລະ

unit economics (ເສດຖະສາດຕໍ່ຫົວໜ່ວຍ) — ຕົວຊີ້ວັດຂອງຜູ້ນຳ: ບໍ່ແມ່ນ “ໃບບິນຂອງເຮົາ \$800k/ເດືອນ” ແຕ່ແມ່ນ “ຕື້ນທຶນ ຕໍ່ທຸລະກຳຂອງລູກຄ້າ ຂອງເຮົາກຳລັງຫຼຸດລົງ.” ໃບຢັ້ງຢືນ FinOps Certified Practitioner ໃຊ້ເວລາປະມານສອງອາທິດ ແລະ ສຳລັບຜູ້ນຳສາຍທຸລະກິດ ມັນຄືໃບຢັ້ງຢືນທີ່ໃຫ້ຄວາມໝ້າເຊື່ອຖືສູງສຸດຕໍ່ຊົ່ວໂມງໃນໂລກ cloud ທັງໝົດ. ໄປເຮົາມາ.

ຄຳຖາມເລື່ອງເງິນຂອງຜູ້ນຳ: “What’s our cost per [customer/transaction/tenant], and which direction is it moving?” (ຕື້ນທຶນຕໍ່ [ລູກຄ້າ/ທຸລະກຳ/tenant] ຂອງເຮົາແມ່ນເທົ່າໃດ, ແລະ ມັນກຳລັງເຄື່ອນໄປທາງໃດ?) · “What percentage of our steady workload is on reservations?” (ຈັກເປີເຊັນຂອງ workload ນຶ່ງໆຂອງເຮົາ ຢູ່ໃນ reservation?) · “What’s untagged?” (ຫຍັງແຕ່ບໍ່ມີ tag?) · “What died but is still billing?” (ຫຍັງຕາຍແລ້ວແຕ່ຍັງຄິດເງິນຢູ່?) (ດິສກຳຜ້າ ແລະ IP ວ່າງເປົ້າ — ທຸກບັນຊີມີ) · “What would this bill look like at 10× growth — does our architecture get cheaper or more expensive per unit?” (ໃບບິນນີ້ຈະເປັນແນວໃດ ຖ້າເຕີບໂຕ 10 ເທົ່າ — ສະຖາປັດຕະຍະກຳຂອງເຮົາ ຖືກລົງ ຫຼື ແພງຂຶ້ນຕໍ່ຫົວໜ່ວຍ?)

ຄຳສັບ:

ຄຳສັບ	ຄວາມໝາຍ
On-demand	ລາຄາຈ່າຍຕາມໃຊ້: ລາຄາເຕັມ, ຍົກເລີກໄດ້ທຸກເວລາ. ຄຳມາດຕະຖານ, ແລະ ເປັນວິທີແພງທີ່ສຸດໃນການແລ່ນ workload ນຶ່ງໆ.
Reserved / savings plan	ການຜູກມັດ 1–3 ປີ ຕໍ່ການໃຊ້ງານນຶ່ງໆ ແລກກັບສ່ວນຫຼຸດ 30–70% — ຄັນໂຍກຕື້ນທຶນໃຫຍ່ທີ່ສຸດອັນດຽວ.
Spot	ຄວາມສາມາດເຫຼືອໃຊ້ ຫຼຸດເຖິງ 90% ທີ່ຜູ້ໃຫ້ບໍລິການດຶງຄືນໄດ້ ໂດຍແຈ້ງລ່ວງໜ້າພຽງນາທີ — ເໝາະທີ່ສຸດສຳລັບວຽກ batch ທີ່ຂັດຈັ່ງຫວະໄດ້.
Rightsizing	ການຫົດ server ທີ່ຈັດຂະໜາດເກີນຕົວ ລົງມາເທົ່າທີ່ມັນໃຊ້ຈິງ. ເງິນຟຣີໃນເກືອບທຸກບັນຊີ.
Egress	ຂໍ້ມູນທີ່ອອກຈາກ cloud — ຄິດເງິນຕໍ່ GB ໃນຂະນະທີ່ຂໍ້ມູນຂາເຂົ້າຟຣີ. ຄວາມແປກໃຈອັນໂດ່ງດັງໃນໃບແຈ້ງໜີ້.
Tagging	ການຕິດປ້າຍທຸກຊັບພະຍາກອນ ດ້ວຍເຈົ້າຂອງ/ໂຄງການ/ສະພາບແວດລ້ອມ ເພື່ອໃຫ້ທຸກບາດຂອງລາຍຈ່າຍ ຊິຕົວໄດ້. ລາຍຈ່າຍບໍ່ມີ tag = ລາຍຈ່າຍບໍ່ມີໃຜຮັບຜິດຊອບ.
Showback / chargeback	ການສະແດງໃບບິນ cloud ຂອງແຕ່ລະທີມໃຫ້ທີມນັ້ນເຫັນ (showback) ຫຼື ຄິດເງິນພາຍໃນຕົວຈິງ (chargeback). ພຶດຕິກຳປ່ຽນທັນທີ.

ຄໍາສັບ	ຄວາມໝາຍ
Budget alert	ການເຕືອນອັດຕະໂນມັດ ເມື່ອຮອດຂີດກຳນົດລາຍຈ່າຍ — ເພື່ອບໍ່ໃຫ້ທ່ານຮູ້ເລື່ອງໃຊ້ເກີນຈາກໃບແຈ້ງໜີ້ຈັກເທື່ອ.
Unit economics	ຕົ້ນທຶນຕໍ່ຫົວໜ່ວຍທຸລະກິດ — ຕໍ່ລູກຄ້າ, ຕໍ່ທຸລະກຳ — ຕົວຊີວັດຂອງຜູ້ນຳ, ມີຄວາມໝາຍຫຼາຍກວ່າຍອດໃບບິນລວມ.
TCO (total cost of ownership)	ຕົ້ນທຶນເຕັມຂອງທາງເລືອກໜຶ່ງ ຕະຫຼອດອາຍຸຂອງມັນ: ໃບອະນຸຍາດ, ຄົ້ນ, ໄຟຟ້າ, ການ migrate — ບໍ່ແມ່ນພຽງລາຄາປ່າຍ.
FinOps	ວິໄນ (ແລະ ວັດທະນະທຳທີມ) ຂອງການເຮັດໃຫ້ລາຍຈ່າຍ cloud ເບິ່ງເຫັນໄດ້, ຈັດສັນໄດ້, ແລະ ຖືກປັບປຸງຢ່າງຕໍ່ເນື່ອງ.

ວິດີໂອສໍາລັບ module ນີ້: What is FinOps? — FinOps Foundation (official) — 2 ນາທີ —

https://www.youtube.com/watch?v=Y-c_xw9bHFW · ຈາກນັ້ນຮຽນຄອສະຟີ “Introduction to FinOps” ທີ່ <https://learn.finops.org>.

ບົດຝຶກຫັດ: (1) ເປີດ AWS pricing calculator ແລ້ວຄິດໄລ່ລາຄາລະບົບຈິງຂະໜາດນ້ອຍ (server ສອງໜ່ວຍ, database ໜຶ່ງອັນ, storage 500GB, egress 1TB) — ການໄດ້ເຮັດມັນເທື່ອດຽວ ຈະປິດຄວາມລຶກລັບຂອງທຸກບົດສົນທະນາເລື່ອງຕົ້ນທຶນໃນອະນາຄົດ. (2) ຂໍໃຫ້ AI ສະແດງບົດເປັນວິສະວະກອນ ທີ່ປົກປ້ອງ server ຂະໜາດເກີນຕົວ; ຝຶກເຈລະຈາການ rightsizing ຢ່າງອ່ອນໂຍນ.

Milestone: ສອບເສັງ (ຫຼືນັດສອບ) FinOps Practitioner, ແລະ ໃນການທົບທວນຈຳລອງ, ຊອກຫາບັນຫາຕົ້ນທຶນສີອັນ ໃນໃບບິນຕົວຢ່າງທີ່ AI ສ້າງໃຫ້ທ່ານ.

Module 8 (ອາທິດທີ 15–16): ຄ້າຍຝຶກຄວາມຄ່ອງແຄ້ວ — ປະກອບພາສາເຂົ້າກັນ

ສອງອາທິດນີ້ຄືການບູລະນາການລ້ວນໆ — ຄວາມແຕກຕ່າງລະຫວ່າງການຮູ້ຄໍາສັບ ກັບ ການຄ່ອງແຄ້ວໃນຫ້ອງປະຊຸມ.

ບົດຝຶກປະຈຳວັນ (30 ນາທີ): ໃຫ້ AI ສ້າງບົດບັນທຶກກອງປະຊຸມທີ່ສົມຈິງ (design review, incident retro, ຫຼື cost review), ໂດຍຝັ່ງຂໍ້ຜິດພາດທາງເຕັກນິກສອງອັນໄວ້ໂດຍເຈດຕະນາ. ວຽກຂອງທ່ານ: ສະຫຼຸບກອງປະຊຸມໃນຫ້າປະໂຫຍກ, ຈັບຂໍ້ຜິດພາດ, ແລະ ຂຽນສາມຄໍາຖາມທີ່ທ່ານຄົງຈະໄດ້ຖາມ. ສະຫຼຸບປະເພດກອງປະຊຸມທຸກມື້.

ກ້າມການແປຄວາມ (15 ນາທີ): ເອົາຄໍາເວົ້າທາງເຕັກນິກມື້ລະໜຶ່ງອັນ ແລ້ວແປໃຫ້ສາມກຸ່ມຜູ້ຝັ່ງ — CFO (ເງິນ), ລູກຄ້າ (ຄວາມສ່ຽງ/ຜົນປະໂຫຍດ), ແລະ ວິສະວະກອນຈູເນຍໃໝ່ (ການສອນ). ຕົວຢ່າງ: “ພວກເຮົາກຳລັງຍ້າຍ

session store ຈາກ database ໄປ Redis” → CFO: “ທູດພາລະ database ຈຶງເລືອນການອັບເກດມູນຄ່າ \$2M ອອກໄປໄດ້” → ລູກຄ້າ: “ໜ້າເວັບໂຟດໄວຂຶ້ນຕອນຄືນຫຼາຍ” → ຈູເນຍ: “Redis ເກັບຂໍ້ມູນ hot ໄວ້ໃນ memory, ພວກເຮົາຈຶງເຊົາທູບ database ໃນທຸກຄລິກ.”

ຝຶກອ່ານ (15 ນາທີ): ບົດຄວາມ engineering blog ຈຶງມືລະບົດ (AWS Architecture Blog, ຫຼື blog ວິສະວະກຳຂອງ Netflix/Grab — Grab ກ່ຽວຂ້ອງເປັນພິເສດ: ຂະໜາດລະດັບອາຊີຕາເວັນອອກສຽງໃຕ້, ຕະຫຼາດໃກ້ຄຽງໄທ). ຕອນນີ້ທ່ານຈະເຂົ້າໃຈ 70–80% ຂອງພວກມັນ. ຄົ້ນຫາສ່ວນທີ່ເຫຼືອ.

ຄູ່ຮ່ວມກຽມສອບຂອງໄລຍະນີ້: ຄອສເຕັມຟຣີທີ່ໂດ່ງດັງ — AWS Certified Cloud Practitioner (CLF-C02) 2026 ໂດຍ Andrew Brown ທາງ freeCodeCamp — <https://www.youtube.com/watch?v=7HKot-brXFE> (ສະບັບກ່ອນໜ້າຄວາມຍາວ 14 ຊົ່ວໂມງ, ໃຊ້ໄດ້ກັບລະຫັດສອບດຽວກັນ, ຢູ່ທີ່ <https://www.youtube.com/watch?v=NhDYbskXRgc>). ເບິ່ງທີ່ຄວາມໄວ 1.25× ຕະຫຼອດອາທິດທີ 13–16; ຫຼັງຈາກ Module 1–7 ສ່ວນໃຫຍ່ຂອງມັນຈະຮູ້ສຶກຄືການທວນຄືນ, ຊຶ່ງນັ້ນແຫຼະຄືສັນຍານວ່າທ່ານພ້ອມແລ້ວ.

Milestone — ຈົບໄລຍະທີ 2, ການສອບຈົບຊັ້ນຂອງທ່ານ: (1) ສອບຜ່ານ AWS Cloud Practitioner ຖ້າຍັງບໍ່ທັນຜ່ານ. (2) ດ່ານທິດສອບຈຳລອງ: AI ສະແດງບົດເປັນວິສະວະກອນອາວຸໂສ ພາທ່ານຍ່າງຜ່ານສະຖາປັດຕະຍະກຳທີ່ມີຂັ້ນສູງສຳລັບລູກຄ້າ e-commerce ໄທ; ທ່ານຕ້ອງພົບ database ທີ່ເປັນ single-AZ, S3 bucket ທີ່ເປັນ public, ການປະເມີນ egress ທີ່ຂາດຫາຍ, ແລະ ການພິຈາລະນາ PDPA ທີ່ບໍ່ມີ — ແລ້ວສົ່ງມອບຄຳຕິຊົມ ດ້ວຍນ້ຳສຽງທີ່ເຮັດໃຫ້ “ວິສະວະກອນ” ຮູ້ສຶກວ່າຖືກຊ່ວຍ, ບໍ່ແມ່ນຖືກຈັບຜິດ. ເມື່ອທ່ານເຮັດໄດ້, ທ່ານກໍ່ເປັນຄົນທີ່ໜ້າຢາກເຖິງໃນການສື່ສານທະນາແລ້ວ, ພຽງສິບຫົກອາທິດເທົ່ານັ້ນ.

ໄລຍະທີ 3 — ນຳພາ (ເດືອນທີ 5–24)

Module 9 (ເດືອນທີ 5–8): ໃບຢັ້ງຢືນຂັ້ນເລິກ ແລະ ພື້ນຖານທາງເຕັກນິກ

ຮຽນເນື້ອຫາ AWS Solutions Architect Associate (SAA-C03) — 2–3 ເດືອນ ຕາມຈັງຫວະໜຶ່ງຊົ່ວໂມງຕໍ່ມື້ຂອງທ່ານ. ທ່ານອາດຈະສອບ ຫຼື ບໍ່ສອບກໍ່ໄດ້ (ໃນຖານະຜູ້ນຳ, ເນື້ອຫາ ຄືຄຸນຄ່າ; ກາໝາຍເປັນພຽງລະຄອນເສີມ), ແຕ່ມີຄືບ່ອນທີ່ region, VPC, IAM, ແລະ ລາຄາ ຢຸດເປັນພຽງຄຳສັບ ແລ້ວກາຍເປັນລະບົບທີ່ເຊື່ອມໂຍງກັນໃນຫົວຂອງທ່ານ. ຄຽງຄູ່ກັນ, ຮັກສາບົດຝຶກປະຈຳວັນຂອງ Module 8 ໄວ້ເຄິ່ງໂດສ. ນີ້ຍັງເປັນຊ່ວງເວລາທີ່ຈະເພີ່ມ Azure fundamentals (ລະດັບ AZ-900) — ປະເທດໄທເປັນຕະຫຼາດວິສາຫະກິດທີ່ໜັກໄປທາງ Microsoft, ແລະ ຄວາມຮູ້ສອງພາສາ (AWS+Azure) ຂະຫຍາຍວົງສົນທະນາກັບລູກຄ້າຂອງທ່ານ.

Module 10 (ເດືອນທີ 5–12, ດຳເນີນຕໍ່ເນື່ອງ): ການຈ້າງ ແລະ ຄຸ້ມຄອງວິສະວະກອນ

ການ້າງເມືອທ່ານຕັດສິນທັກສະເອບໂຕ້ຕິມທິ: ໂຄງສ້າງຊະນະສັນຊາດຕະຍານ. ໃຊ້ວົງຈອນທີ່ຄົງເສັ້ນຄົງວາ: ການສິນທະນາຄັດກອງທີ່ທ່ານນຳເອງ (ແຮງຈູງໃຈ, ການສື່ສານ, ວິທີເຂົ້າອະທິບາຍໂຄງການເກົ່າ ໃຫ້ຄົນທີ່ບໍ່ແມ່ນ ວິສະວະກອນຝັ່ງ — ຖ້າເຂົາເຈັດບໍ່ໄດ້, ເຂົາກໍຈະລົ້ມເຫຼວກັບລູກຄ້າຂອງທ່ານເຊັ່ນກັນ) + ການສຳພາດເຕັກນິກ ດຳເນີນໂດຍ bar-raiser ຂອງທ່ານ (ຜູ້ຊ່ວຍເຕັກນິກໝາຍເລກສອງຂອງທ່ານ ຫຼື ຜູ້ຮັບເໝົາອາວຸໂສທີ່້າງມາ) + ການໂທຫາຜູ້ອ້າງອີງ ບ່ອນທີ່ທ່ານຖາມພຽງໜຶ່ງຄຳຖາມທີ່ສຳຄັນແທ້: “would you hire this person again for this role?” (ທ່ານຈະ້າງຄົນນີ້ອີກຄັ້ງ ສຳລັບຕຳແໜ່ງນີ້ບໍ່?) ລະວັງສອງແບບຢ່າງແຫ່ງຄວາມລົ້ມເຫຼວ: ນັກເວົ້າ ຄ່ອງທີ່ບໍ່ມີຄວາມເລິກ (bar-raiser ຂອງທ່ານຈະຈັບໄດ້) ແລະ ຜູ້ຊ່ຽວຊານເລິກແຕ່ມິດງຽບ (ມັກເປັນຄຳໃນທີມໂທ, ບ່ອນທີ່ຄວາມຖ່ອມຕົນເປັນວັດທະນະທຳ — ຢ່າໃຫ້ຄວາມມັນວາວໃນການສຳພາດ ມີນ້ຳໜັກກວ່າຫຼັກຖານຂອງ ຜົນງານຈິງ).

ຜູ້ຊ່ວຍເຕັກນິກໝາຍເລກສອງ — ການຕັດສິນໃຈສຳຄັນທີ່ສຸດຂອງກິດຈະການ: ັ້າງເຂົ້າກ່ອນໝູ່, ຈ່າຍດ້ວຍທຸ້ນທີ່ ມີຄວາມໝາຍ (10–20% ຖ້າເປັນ co-founder ແທ້ໆ), ແລະ ກຳນົດຂໍ້ຕົກລົງຢ່າງຈະແຈ້ງ: ເຂົາຖືມາດຕະຖານ ເຕັກນິກ ແລະ ເປັນເຈົ້າຂອງການຕັດສິນໃຈດ້ານສະຖາປັດຕະຍະກຳ; ທ່ານເປັນເຈົ້າຂອງລູກຄ້າ, ເງິນ, ບຸລິມະສິດ, ແລະ ຄົນ; ຄວາມເຫັນຕ່າງລະຫວ່າງທ່ານທັງສອງ ເກີດຂຶ້ນເປັນການສ່ວນຕົວ ແລະ ແກ້ໄຂໃຫ້ຈົບກ່ອນທີ່ມຈະເຫັນ.

ພິທີກຳທີ່ເຮັດໃຫ້ວິສະວະກອນຢູ່ຕໍ່: 1:1 ທຸກອາທິດ ຫຼື ສອງອາທິດຕໍ່ຄັ້ງ ທີ່ເປັນເລື່ອງຂອງ ເຂົາ (ອາຊີບ, ຄວາມຝຶດ, ພະລັງງານ — ບໍ່ແມ່ນລາຍງານສະຖານະ); ເສັ້ນທາງເຕີບໂຕເປັນລາຍລັກອັກສອນ ຕໍ່ຄົນ (ໃນຕະຫຼາດຂາດແຄນ ຄົນເກັ່ງ 70k/ປີ ຂອງໂທ, ງົບການເຕີບໂຕ ແລະ certification ຮັກສາຄົນໄດ້ດີກວ່າເງິນເດືອນຢ່າງດຽວ — ຈ່າຍໃຫ້ ທຸກ cert, ພ້ອມໂບນັດ vesting 12 ເດືອນ ເມື່ອສຳເລັດ); ຍ້ອງຍໍໃນທີ່ແຈ້ງ, ຕັກເຕືອນໃນທີ່ລັບ; ແລະ ປົກປ້ອງເວລາ ສະມາທິຂອງເຂົາຢ່າງເດັດດ່ຽວ — ຜູ້ນຳທີ່ຍົກເລີກກອງປະຊຸມກາງ sprint ຄືວິລະບຸລຸດ.

ເວທີຕັດສິນໃຈ — ວິທີທີ່ທ່ານຕັດສິນເລື່ອງເຕັກນິກ ທີ່ທ່ານປະເມີນເອບໂຕ້ຕິມທິ: ສຳລັບທຸກການຕັດສິນໃຈໃຫຍ່, ຮຽກຮ້ອງ decision doc (ເອກະສານການຕັດສິນໃຈ) ໜຶ່ງໜ້າ ຈາກວິສະວະກອນຜູ້ສະເໜີ: ບັນຫາ, ທາງເລືອກ 2–3 ທາງ, ຕົ້ນທຶນ, ຄວາມສ່ຽງ, ຂໍ້ແນະນຳ. ຈາກນັ້ນດຳເນີນກອງປະຊຸມ ດ້ວຍເຈັດຄຳຖາມຂອງທ່ານຈາກ Module 5. ທ່ານບໍ່ແມ່ນສະຖາປະນິກທີ່ສະຫຼາດທີ່ສຸດໃນຫ້ອງ ແລະ ບໍ່ຈຳເປັນຕ້ອງເປັນຈັກເທືອ — ທ່ານຄືຄົນທີ່ເຮັດໃຫ້ທາງ ເລືອກທີ່ ຖືກໄຕ້ແຍ້ງມາດີທີ່ສຸດ ຊະນະ, ຕາມກຳນົດເວລາ, ໂດຍການແລກປ່ຽນເລື່ອງເງິນ ແລະ ຄວາມສ່ຽງຖືກເຮັດ ໃຫ້ຈະແຈ້ງ. ວິສະວະກອນນັບຖືສິ່ງນີ້ຢ່າງເລິກເຊິ່ງ ເມື່ອມັນຖືກເຮັດຢ່າງຊື່ສັດ; ຈິດໄວ້ວ່າໃຜທ່ານາຍຫຍັງ (Decision Journal ຂອງທ່ານອີກແລ້ວ) ແລະ ທົບທວນຄຳທ່ານາຍທຸກໂຕມາດ — ມັນປັບຄວາມແມ່ນຢ່າທັງຂອງ ທ່ານ ແລະ ຂອງເຂົາ.

Module 11 (ເດືອນທີ 6–24): ເສັ້ນໂຄ້ງຄວາມໜ້າເຊື່ອຖື, ຢ່າງກົງໄປກົງມາ

ປະສົບການທີ່ຕິພົມ ຈາກວັນນະຄະດີການນຳພາດ້ານວິສະວະກຳ ບັນຈົບກັນທີ່ເສັ້ນເວລານີ້, ແລະ ການທຳທ່າວ່າບໍ່ ແມ່ນ ຄືວິທີທີ່ຜູ້ກໍ່ຕັ້ງສາຍບໍ່ແມ່ນເຕັກນິກລົ້ມເຫຼວ: ~90 ວັນ ເພື່ອດຳເນີນຈັງຫວະທຸລະກິດຢ່າງມີຄວາມສາມາດ · ~6 ເດືອນ ສູ່ປະສິດທິຜົນຂັ້ນພື້ນຖານ (ປະຊຸມຄ່ອງແຄ້ວ, ຕັດສິນໃຈມີໂຄງສ້າງ, ທີມໝັ້ນຄົງ) · 12–18 ເດືອນ ກ່ອນ ສັນຊາດຕະຍານທາງເຕັກນິກຂອງທ່ານ ຈະມີຄ່າດ້ວຍຕົວມັນເອງ · ~2 ປີ ກ່ອນທ່ານຈະຢືນຢູ່ໄດ້ຢ່າງແທ້ຈິງ ຕໍ່ໜ້າ ວິສະວະກອນອາວຸໂສ ໃນເລື່ອງການແລກປ່ຽນທາງສະຖາປັດຕະຍະກຳ. ວິທີບັນເທົາໃນຂະນະທີ່ເສັ້ນໂຄ້ງໄຕ້ຂຶ້ນ: ຍືມຄວາມໜ້າເຊື່ອຖື (ໝາຍເລກສອງຂອງທ່ານ ນຳສະເໜີເຄິ່ງເຕັກນິກຂອງກອງປະຊຸມຂາຍ — ລູກຄ້າຕ້ອງເຫັນ

ທີ່ມາສຳຮອງຢູ່ແລ້ວ), ຢ່າຖືວ່າເດັດຂາດ (ຖືວ່າທຳລາຍຄວາມໜ້າເຊື່ອຖືທີ່ໄວທີ່ສຸດ; “I don’t know — walk me through it” (ຂ້ອຍບໍ່ຮູ້ — ພາຂ້ອຍຜ່ານມັນແດ່) ຄືປະໂຫຍກຂອງຜູ້ນຳ), ແລະ ປ່ອຍໃຫ້ຄຳຖາມຂອງທ່ານເປັນຜູ້ເວົ້າແທນ: ຜູ້ນຳທີ່ຖາມ “what’s our RPO and who signed off on it?” (RPO ຂອງເຮົາແມ່ນເທົ່າໃດ ແລະ ໃຜອະນຸມັດ?) ຟັງຄືມີແຜ່ນເປັນສາມສິບປີ, ແລະ ຫຼັງຫຼັກສູດນີ້, ທ່ານຈະໝາຍຄວາມຕາມນັ້ນແທ້.

Module 12 (ຕໍ່ເນື່ອງ): ຊັ້ນຕະຫຼາດໄທ

ນຳເອົາລາຍລະອຽດຈາກ TSI Part 5 ມາເປັນຄວາມຮູ້ປະຕິບັດງານ: **PDPA** ໃນຖານະທັງໝົດທີ່ຕາມກົດໝາຍ ແລະ ຜະລິດຕະພັນ (Module 6); **ບັນໄດຄູ່ຮ່ວມທຸລະກິດ** (AWS Select ຕ້ອງການພະນັກງານມີ cert ຈຳນວນໜຶ່ງ + 3 ຕົວທີ່ launch ແລ້ວ; Microsoft Solutions Partner ຕ້ອງການຄະແນນຄວາມສາມາດ 70/100 — certification ຂອງທີມທ່ານ ຄືຊັບສິນການຂາຍໂດຍກົງ, ອີກເຫດຜົນໜຶ່ງທີ່ຄວນອອກຫິນໃຫ້); **ການລົງເສີມ BOI** ສຳລັບການຖືຫຸ້ນຕ່າງຊາດ 100% ຂອງບໍລິສັດ; ຂັ້ນເງິນເດືອນບາງກອກ (ຈູເນຍ ฿50–75k → architect ฿180–280k/ເດືອນ) ເພື່ອຕັ້ງລາຄາປະມູນ ແລະ ຂໍສະເໜີຈ້າງໄດ້ຖືກຕ້ອງ; ແລະ ເລກຄະນິດການຂາຍຂອງຊ່ວງເວລານີ້ — ການລົງທຶນ data center ທີ່ອະນຸມັດແລ້ວກວ່າ \$27B, ຄຳສັ່ງ cloud-first ຂອງລັດຖະບານ, ແລະ ຊ່ອງວ່າງທັກສະ 70,000 ຄົນ/ປີ ທີ່ທີມງານມີ cert ຂອງທ່ານມີໄວ້ເພື່ອຕື່ມເຕັມ.

ຫຼັກສູດຫຍໍ້ໜຶ່ງໜ້າ

ເມື່ອໃດ	ຈຸດເນັ້ນ	ຫຼັກຖານພາຍນອກ
ອາທິດທີ 1–2	Cloud ແມ່ນຫຍັງ; IaaS/PaaS/SaaS; region/AZ	—
ອາທິດທີ 3–4	Compute, storage, database	—
ອາທິດທີ 5–6	ເຄືອຂ່າຍ; ການອ່ານແຜນວາດ ສະຖາປັດຕະຍະກຳ	—
ອາທິດທີ 7–8	DevOps, IaC, ພິທີກຳຂອງທີມ, ຕົວ ຊີວິດ ops	ນັດສອນ Cloud Practitioner
ອາທິດທີ 9–10	ວິຈາລະນະຍານດ້ານ ສະຖາປັດຕະຍະກຳ; ເຈັດຄຳຖາມ	—
ອາທິດທີ 11–12	ຄວາມປອດໄພ; PDPA; shared responsibility	ສອນ AWS Cloud Practitioner

ເມື່ອໃດ	ຈຸດເນັ້ນ	ຫຼັກຖານພາຍນອກ
ອາທິດທີ 13–14	ເສດຖະສາດ cloud; FinOps	FinOps Practitioner (ກຽມ ≈2 ອາທິດ)
ອາທິດທີ 15–16	ຄ້າຍຝຶກຄວາມຄ່ອງແຄ້ວ; ຕໍານ ທິດສອບຈໍາລອງ	ຈົບຊັ້ນ: ຕໍານທິດສອບ
ເດືອນທີ 5–8	ເນື້ອຫາ SAA-C03; Azure AZ-900	ສອບ SAA (ທາງເລືອກ)
ເດືອນທີ 5–12	ການຈ້າງຄົນ; ໝາຍເລກສອງດ້ານ ເຕັກນິກ; ເວທີຕັດສິນໃຈ	ການຈ້າງຄັງທໍາອິດທີ່ເຮັດໄດ້ດີ
ເດືອນທີ 6–24	ເສັ້ນໂຄງຄວາມໜ້າເຊື່ອຖື; ຊັ້ນ ຕະຫຼາດໄທ	ຂັ້ນ partner; ສັນຍາ retainer ຄັງ ທໍາອິດ

ຄໍາສັ່ງທ້າຍຈາກຄູຂອງທ່ານ. ອີກສິບຫົກອາທິດຈາກນີ້ ທ່ານຈະຕິດຕາມທຸກບົດສົນທະນາໃນຫ້ອງປະຊຸມໄດ້. ນັ້ນບໍ່ແມ່ນເສັ້ນໄຊ — ມັນຄືໃບອະນຸຍາດໃຫ້ເລີ່ມຕົ້ນ. ເສັ້ນໂຄງສອງປີ ສູ່ວິຈາລະນະຍານທາງເຕັກນິກຕົວຈິງ ບໍ່ແມ່ນກໍາແພງ; ມັນຄືຄູນ້ຳປ້ອງກັນ: ທຸກອາທິດຂອງມັນທີ່ທ່ານເຮັດສໍາເລັດ ຄືອາທິດທີ່ຜູ້ກໍ່ຕັ້ງສາຍບໍ່ແມ່ນເຕັກນິກຂອງຄູແຂ່ງບໍ່ໄດ້ເຮັດ. ຮຽນທຸກມື້, ບັນທຶກທຸກການຕັດສິນໃຈ, ຢ່າຕົວະເດັດຂາດ, ແລະ ຈ້າງຄົນທີ່ເກັ່ງກວ່າທ່ານ ແລ້ວເຮັດໃຫ້ເຂົາຕິໃຈທີ່ໄດ້ມາ. ນັ້ນຄືວຽກທັງໝົດ.

ເອກະສານປະກອບຂອງ “The Thailand Strategic Investment (TSI),” Part 5. ການປະເມີນເວລາກຽມສອບ certification ແລະ ເສັ້ນເວລາການນໍາພາ ອ້າງອີງຈາກແຫຼ່ງທີ່ອ້າງໄວ້ໃນນັ້ນ (CBT Nuggets, StudyTech, FinOps Foundation, First Round Review, The Pragmatic Engineer).

ບົດຮຽນໃໝ່ — ສະບັບເດືອນສິງຫາ 2026

ໃນເດືອນສິງຫາ 2026, ບົດຮຽນໃໝ່ໄດ້ຖືກເພີ່ມເຂົ້າໃນສາມຫຼັກສູດແບບໂຕ້ຕອບທີ່ b4lcilc.org — ຫຼັກສູດຜູ້ນໍາດ້ານ Cloud (Cloud Leader), ວິສະວະກອນ Cloud (Cloud Engineer), ແລະ ສະຖາປະນິກ Cloud (Cloud Architect). ເອກະສານເພີ່ມເຕີມສະບັບນີ້ ນໍາສະບັບອ່ານໃຫ້ທັນສະໄໝ ດ້ວຍບົດສະຫຼຸບຫຍໍ້ທີ່ຊື່ສັດຕໍ່ເນື້ອຫາໃໝ່ນັ້ນ, ໃນນ້ຳສຽງດຽວກັນກັບສ່ວນອື່ນຂອງປຶ້ມເຫຼົ່ານີ້.

ຂໍເວົ້າຈັກຄໍາກ່ຽວກັບສິ່ງທີ່ທ່ານຖືຢູ່ໃນມື. ໜ້າເຫຼົ່ານີ້ຄືສະບັບອ່ານ: ແນວຄິດ, ຄໍາປຽບທຽບ, ແລະ ປະໂຫຍກທີ່ຄຸ້ມຄໍາແກ່ການເວົ້າອອກສຽງ. ສະບັບເຕັມແບບໂຕ້ຕອບທາງອອນລາຍ ມີເນື້ອຫາຫຼາຍກວ່ານີ້ຫຼາຍ — ທຸກບົດຮຽນໃໝ່ມາພ້ອມກັບແບບທິດສອບ (quiz) ທີ່ກວດຄວາມເຂົ້າໃຈຂອງທ່ານກ່ອນຈະໃຫ້ກ້າວຕໍ່ໄປ, ຫ້ອງທິດລອງແບບລົງມື

ເຮັດຈິງ (ລວມທັງ terminal ຝຶກຫັດ ທີ່ເຮັດວຽກຄືກັບຂອງແທ້ ແຕ່ບໍ່ສາມາດເສຍເງິນໄດ້), ວິດີໂອ, ບົດຝຶກຄຳສັບ, ແລະ — ສຳລັບເນື້ອຫາ capstone (ບົດຝຶກລວມຍອດ) — ບົດຝຶກອອກແບບແບບມີຄຳແນະນຳ ຜ່ານລາຍການ ກວດຄວາມຕ້ອງການ ແລະ ຄຳຕອບຕົວຢ່າງທີ່ເຮັດໃຫ້ເບິ່ງ. ຈົ່ງອ່ານບົດສະຫຼຸບເຫຼົ່ານີ້ ເພື່ອໃຫ້ຮູ້ບ່ອນຂອງແຕ່ລະ ແນວຄິດເຂົ້າໄປຢູ່ໃນຫົວຂອງທ່ານ; ຈາກນັ້ນ, ເມື່ອມີໂອກາດ, ໃຫ້ນັ່ງເຮັດສະບັບໂຕ້ຕອບ ເພື່ອໃຫ້ຄວາມຮູ້ ຕິດແໜ້ນ.

ເນື້ອຫາທີ່ເພີ່ມແບ່ງອອກເປັນສາມກຸ່ມ. ສຳລັບຫຼັກສູດ **ຜູ້ນຳດ້ານ Cloud (Cloud Leader)**: ການແນະນຳ terminal ຢ່າງນຸ່ມນວນ, ກົດໝາຍຄວາມເປັນສ່ວນຕົວຂອງໄທເອງ, ຄູ່ມືຜູ້ນຳສຳລັບຂັ້ນສະເໝີ AI ແລະ ໃບບິນ AI, ແລະ ບົດຝຶກ capstone ທີ່ປະກອບຫຼັກສູດທັງໝົດເຂົ້າເປັນໜຶ່ງໜ້າ. ສຳລັບຫຼັກສູດ **ວິສະວະກອນ Cloud (Cloud Engineer)**: serverless computing — ຮູບແບບທີ່ປ່ຽນສະຄຣິບປະຕິບັດງານນ້ອຍໆ ໃຫ້ກາຍເປັນ function ທີ່ແລ່ນໂດຍບໍ່ເສຍຄ່າຫຍັງ. ສຳລັບຫຼັກສູດ **ສະຖາປະນິກ Cloud (Cloud Architect)**: ເລກຄະນິດຂອງ ຄຳສັນຍາຄວາມຮ່ວມໃຊ້ງານ, ຮູບແບບ Retrieval-Augmented Generation ທີ່ທຸກມື້ນິລູກຄ້າຖາມເຖິງທຸກ ອາທິດ, ແລະ ບົດຝຶກຄິດໄລ່ຕົ້ນທຶນເຕັມຮູບແບບ — ຕີລາຄາການອອກແບບ ກ່ອນທີ່ໃຜຈະລົງມືສ້າງ.

ສຳລັບຫຼັກສູດຜູ້ນຳດ້ານ Cloud

ໜ້າຕ່າງສີດຳ — ວິທີເວົ້າກັບຄອມພິວເຕີໂດຍກົງ

ກ່ອນທ້ອງທົດລອງທຳອິດຂອງທ່ານ, ມາຮູ້ຈັກເຄື່ອງມືທີ່ມັນແລ່ນຢູ່ໃນນັ້ນກ່ອນ. terminal (ເທີມິນອລ — ເອີ້ນອີກ ຢ່າງວ່າ *command line* ຫຼື *console*) ຄືໂປຣແກຣມທີ່ເບິ່ງຄືໜ້າຕ່າງສີດຳທຳມະດາ ມີຕົວໜັງສືກະພິບຢູ່. ມັນຄືວິທີ ໃຊ້ຄອມພິວເຕີທີ່ເກົ່າແກ່ທີ່ສຸດ, ແລະ ຍັງຄົງເປັນວິທີທີ່ວິສະວະກອນເຮັດວຽກຈິງຈັງ — ເພາະຄຳສັ່ງທີ່ພົບອອກມາ ມີຄວາມແມ່ນຍຳ, ເຮັດຊ້າໄດ້, ແລະ ຂຽນບັນທຶກ ແລະ ແບ່ງປັນຕໍ່ໄດ້ ໃນແບບທີ່ການກົດຜ່ານເມນູບໍ່ມີວັນເຮັດໄດ້.

Terminal ຈະສະແດງ **prompt** (ພຣອມ) — ໂດຍປົກກະຕິເປັນແຖວສັ້ນໆ ລົງທ້າຍດ້ວຍ \$ — ເຊິ່ງໝາຍເຖິງສິ່ງ ດຽວເໝົານັ້ນ: *ຂ້ອຍກຳລັງຟັງຢູ່. ຄຳສັ່ງ (command)* ຄືປະໂຫຍກທີ່ມີໄວຍາກອນ: ຄຳທຳອິດຄື *ເຄື່ອງມື* ທີ່ທ່ານ ກຳລັງເອີ້ນໃຊ້ (ເຊັ່ນ git ຫຼື aws), ຄຳຕໍ່ໄປບອກວ່າ *ໃຫ້ເຮັດຫຍັງ*, ແລະ ສ່ວນທີ່ຂຶ້ນຕົ້ນດ້ວຍເຄື່ອງໝາຍຂີດ, ເອີ້ນ ວ່າ **flags** (ເຊັ່ນ --output table), ປັບແຕ່ງວ່າ *ຈະເຮັດແນວໃດ*.

ມີສອງທ່ານຽມທີ່ເຮັດໃຫ້ຜູ້ມາໃໝ່ທຸກຄົນແປກໃຈ. ຢ່າງທຳອິດ: **ຄວາມງຽບ ໂດຍທົ່ວໄປໝາຍເຖິງຄວາມສຳເລັດ**. ຫຼາຍຄຳສັ່ງ, ເມື່ອເຮັດວຽກໄດ້ຢ່າງສົມບູນ, ຈະບໍ່ພິມຫຍັງອອກມາເລີຍ — ເຄື່ອງຖືວ່າການເຮັດວຽກໃຫ້ສຳເລັດ ນັ້ນ ກໍເປັນຄຳຕອບທີ່ພຽງພໍແລ້ວ. ຢ່າງທີສອງ: terminal ບໍ່ເດົາ. ບ່ອນທີ່ search engine ໃຫ້ອະໄພການສະກົດ ຜິດ, terminal ບໍ່ໃຫ້ອະໄພຫຍັງເລີຍ — ຜິດຕົວອັກສອນດຽວ ມັນກໍຈະບອກທ່ານຢ່າງສຸພາບວ່າຄຳສັ່ງນັ້ນບໍ່ມີຢູ່. ຄວາມເຂັ້ມງວດນັ້ນຮູ້ສຶກໂຫດຮ້າຍຢູ່ມືດຽວ ແລ້ວກໍກາຍເປັນຄວາມອຸ່ນໃຈ: ເຄື່ອງບໍ່ເຄີຍເຮັດ *ປະມານໆ* ຕາມສິ່ງທີ່ ທ່ານຂໍ.

ໄວຍາກອນອີກຂໍ້ໜຶ່ງ: ຄອມພິວເຕີຈັດລະບຽບທຸກຢ່າງເປັນ files (ໄຟລ) ທີ່ຢູ່ພາຍໃນ folders (ໂຟນເດີ — ເອີ້ນອີກຢ່າງວ່າ directories) — ດັ່ງນັ້ນ mkdir decisions ຈຶ່ງໝາຍຄວາມວ່າ “make directory — ສ້າງ directory ຊື່ decisions.” ຫ້ອງທົດລອງຂອງຫຼັກສູດແລ່ນຢູ່ໃນ simulator (ເຄື່ອງຈຳລອງ): terminal ຝຶກຫັດ ທີ່ບໍ່ສາມາດສ້າງ, ລຶບ, ຫຼື ເສຍເງິນຫຍັງໄດ້ເລີຍ — ແຕ່ຄຳສັ່ງແມ່ນຂອງແທ້, ຖືກຕ້ອງທຸກຕົວອັກສອນ. ຝຶກມັນໃຫ້ຊຳນານຢູ່ບ່ອນນີ້, ແລ້ວມືທ່ານນັ່ງຢູ່ໜ້າ terminal ຂອງແທ້, ມືຂອງທ່ານຈະຮູ້ທາງຢູ່ແລ້ວ.

PDPA ຂອງໄທ — ກົດໝາຍຄວາມເປັນສ່ວນຕົວທີ່ໃກ້ບ້ານທີ່ສຸດ

ສ່ວນອື່ນຂອງ module ການປະຕິບັດຕາມກົດລະບຽບ (compliance) ແມ່ນລະດັບໂລກ; ນິຕິກົດໝາຍທີ່ຜູ້ຄຸມກົດ (regulator) ຂອງມັນ ສາມາດມາເຄາະປະຕູໄດ້ແທ້ໆ. PDPA ຂອງໄທ — Personal Data Protection Act (ກົດໝາຍຄຸ້ມຄອງຂໍ້ມູນສ່ວນບຸກຄົນ), B.E. 2562 (2019) — ມີຜົນບັງຄັບໃຊ້ເຕັມຮູບແບບໃນປີ 2022 ແລະ ນຳໃຊ້ກັບທຸກອົງກອນໃນປະເທດໄທ ທີ່ຈັດການຂໍ້ມູນສ່ວນບຸກຄົນ. ອົງກອນການກຸສົນໃນແມ່ສອດ ທີ່ຖືບັນທຶກຂໍ້ມູນນັກຮຽນ ຢູ່ໃນຂອບເຂດຂອງມັນຢ່າງເຕັມຕົວ.

ຂ່າວດີແມ່ນດີແທ້ໆ: PDPA ເກືອບເປັນສຳເນົາຂອງ GDPR — General Data Protection Regulation (ລະບຽບການຄຸ້ມຄອງຂໍ້ມູນທົ່ວໄປ) ຂອງເອີຣົບ. ຜູ້ຮ່າງກົດໝາຍຂອງໄທ ເອົາລະບຽບການເອີຣົບມາເປັນແມ່ແບບ, ດັ່ງນັ້ນ GDPR ໄດ້ສອນທ່ານໄປແລ້ວປະມານເກົາສິບເປີເຊັນຂອງກົດໝາຍນີ້. ຂໍ້ມູນສ່ວນບຸກຄົນເປັນຂອງບຸກຄົນທີ່ມັນບັນຍາຍເຖິງ; ທ່ານຕ້ອງມີພື້ນຖານທາງກົດໝາຍ — ສ່ວນຫຼາຍແມ່ນ **ການຍິນຍອມ (consent)**, ທີ່ໃຫ້ຢ່າງເສລີ ແລະ ຖອນຄືນໄດ້ — ເພື່ອຈະເກັບກຳມັນ; ແລະ **ການຈຳກັດຈຸດປະສົງ (purpose limitation)** ຫ້າມນຳຂໍ້ມູນທີ່ເກັບມາເພື່ອຈຸດປະສົງໜຶ່ງ ໄປໃຊ້ເພື່ອຈຸດປະສົງອື່ນ: ບັນທຶກທີ່ເກັບເພື່ອລົງທະບຽນນັກຮຽນ ຈະກາຍເປັນລາຍຊື່ສົ່ງຈົດໝາຍລະດົມທຶນຢ່າງງຽບໆ ບໍ່ໄດ້.

ຜູ້ຄຸມກົດແມ່ນ **PDPC — Personal Data Protection Committee (ຄະນະກຳມະການຄຸ້ມຄອງຂໍ້ມູນສ່ວນບຸກຄົນ)** — ເຊິ່ງໂທດຂອງມັນໄປເຖິງຄວາມຮັບຜິດຊອບທາງອາຍາ, ບໍ່ແມ່ນພຽງຄຳປັບໄໝ. ໂມງນັບເວລາເມື່ອຂໍ້ມູນຮົວໄຫຼ ກໍ່ຮຸ້ນເຄີຍເຊັ່ນກັນ: ການຮົວໄຫຼທີ່ອາດສ້າງຄວາມສ່ຽງຕໍ່ບຸກຄົນ ຕ້ອງລາຍງານຕໍ່ PDPC ພາຍໃນ 72 ຊົ່ວໂມງ.

ຈຸດສຳຄັນດ້ານການນຳພາ ສົມຄວນເວົ້າໃຫ້ແຈ້ງ. ອົງກອນທີ່ຖືບັນທຶກກ່ຽວກັບນັກຮຽນ — ຊື່, ຄອບຄົວ, ແລະ ໃນພາກພື້ນນີ້ ບາງຄັ້ງກໍ່ລວມເຖິງສະຖານະເອກະສານປະຈຳຕົວ, ເຊິ່ງນັບວ່າອ່ອນໄຫວທີ່ສຸດເທົ່າທີ່ຂໍ້ມູນຈະອ່ອນໄຫວໄດ້ — **ຕ້ອງຕອບກົດໝາຍນີ້ກ່ອນໝູ່:** ຄຳຕອບຕ້ອງເຮັດໃຫ້ບາງກອກພໍໃຈກ່ອນ ຈຶ່ງຄ່ອຍໄປກັງວົນເລື່ອງບຣັດເຊລ. ຄຳຖາມທີ່ຄວນຖາມທີ່ມີຂອງທ່ານ: ພວກເຮົາຖືຂໍ້ມູນສ່ວນບຸກຄົນຫຍັງແດ່, ແລະ ພວກເຮົາໄດ້ລະບຸຈຸດປະສົງຕອນເກັບກຳມັນບໍ່? ຖ້າມີຄົນຮ້ອງຂໍ, ພວກເຮົາສາມາດລຶບບັນທຶກຂອງນັກຮຽນຄົນໜຶ່ງອອກຢ່າງສົມບູນໄດ້ບໍ່? ຖ້າຖືກເຈາະລະບົບໃນຄືນນີ້, ພວກເຮົາຈະບອກ PDPC ພາຍໃນ 72 ຊົ່ວໂມງໄດ້ບໍ່ ວ່າອັນໃດຖືກເອົາໄປ ແລະ ເປັນຂອງໃຜ? ບໍ່ມີຂໍ້ໃດຕ້ອງອາໄສທະນາຍຄວາມເພື່ອຖາມ. ແຕ່ທຸກຂໍ້ຕ້ອງອາໄສຜູ້ນຳ ເພື່ອຍືນຍັນໃຫ້ເຮັດ.

ນຳພາຜ່ານຄືນ AI

ບໍ່ໄວກໍ່ຊ້າ — ແລະ ໄລຍະຫຼັງນີ້ມັກຈະໄວ — ການອອກແບບທີ່ມີ AI ຢູ່ໃນນັ້ນ ຈະມາຮອດໂຕຂອງທ່ານ. ວຽກຂອງທ່ານບໍ່ປ່ຽນ: ທ່ານສອບຖາມເຈາະເລິກ, ທ່ານບໍ່ອອກແບບເອງ.

ເຄື່ອງຈັກເບື້ອງຫຼັງຄຸນສົມບັດ AI ສ່ວນໃຫຍ່ຄື LLM — Large Language Model (ໂມເດວພາສາຂະໜາດໃຫຍ່) — ແລະ ຂັ້ນຕອນທາງການຄ້າທີ່ສໍາຄັນທີ່ສຸດ: ທ່ານເກືອບບໍ່ເຄີຍເປັນເຈົ້າຂອງມັນເອງ; ທ່ານເຊົາການເຂົ້າເຖິງຜ່ານ AI API (Application Programming Interface — ຊ່ອງທາງໃຫ້ໂປຣແກຣມເວົ້າກັນ), ຄິດເງິນຕາມ token — ຕ່ອນຄໍາທີ່ຍາວປະມານສາມສ່ວນສີຂອງຄໍາພາສາອັງກິດ — ດັ່ງນັ້ນຄຸນສົມບັດໜຶ່ງຈະແພງຂຶ້ນທຸກຄັ້ງທີ່ຄໍາຖາມ ຫຼື ຄໍາຕອບຍາວຂຶ້ນ. ຂໍເບິ່ງເລກຄິດໄລ່ token, ຕໍ່ຜູ້ໃຊ້, ຕໍ່ເດືອນ; ຖ້າຍັງບໍ່ມີໃຜເຮັດ, ຄຸນສົມບັດນັ້ນຍັງບໍ່ມີລາຄາ, ມີພຽງຄວາມຫວັງ.

ທາງເລືອກອີກຢ່າງ — ແລ່ນໂມເດວຂອງຕົນເອງ — ໝາຍເຖິງການເຊົາ GPUs (Graphics Processing Units — ຊິບປະມວນຜົນກຣາຟິກ), ຊິບສະເພາະທາງທີ່ AI ຕ້ອງການ: ຫາຍາກ, ລາຄາແພງກວ່າ server ທໍາມະດາຫຼາຍເທົ່າຄິດ, ແລະ ຄິດເງິນທຸກຊົ່ວໂມງທີ່ເປີດຢູ່ ບໍ່ວ່າກໍາລັງຄິດວຽກ ຫຼື ຢູ່ລ້າໆ. ກົດຂໍ້ທໍາອິດຂອງສາຍ AI: GPU ທີ່ຢູ່ລ້າຄືຄວາມຢູ່ລ້າທີ່ແພງທີ່ສຸດໃນອາຄານ. Server ທົດສອບທີ່ຖືກລືມ ເສຍເງິນເທົ່າຄ່າກາເຟ; GPU instance ທີ່ຖືກລືມ ເສຍເງິນເທົ່າເງິນເດືອນຄົນໜຶ່ງ. ແລະ ເມື່ອມີໃຜເວົ້າວ່າ ມາ fine-tune ໂມເດວຂອງເຮົາເອງເທາະ, ຈິ່ງຈື່ລໍາດັບທີ່ຊື່ສັດ: ຂຽນ prompt ຢ່າງລະມັດລະວັງເທິງ API ທີ່ເຊົາກ່ອນ, ຈາກນັ້ນຄ່ອຍເອົາເອກະສານຂອງທ່ານໃຫ້ໂມເດວໃນເວລາຖາມ, ແລະ ຕໍ່ເມື່ອທັງສອງຢ່າງບໍ່ພຽງພໍເທົ່ານັ້ນ, ຈິ່ງຄ່ອຍ fine-tune.

ສາມຄໍາຖາມພາທ່ານຜ່ານທຸກຂັ້ນຕອນ AI ໄດ້. ໝຶ່ງ: ຄໍາຕອບທີ່ຜິດ ເສຍຫາຍເທົ່າໃດ? LLM ຕອບຜິດຢ່າງໜັ້ນໃຈໃນອັດຕາສ່ວນນ້ອຍໆທີ່ຫຼຸດລົງອີກບໍ່ໄດ້; ອັດຕາຄວາມຜິດພາດທີ່ຍອມຮັບໄດ້ ແມ່ນການຕັດສິນໃຈທາງທຸລະກິດ — ເຊິ່ງໝາຍຄວາມວ່າມັນເປັນຂອງທ່ານ. **ສອງ: ຂໍ້ມູນຫຍັງອອກໄປຈາກເຮົາ?** ທຸກຄໍາຖາມທີ່ສົ່ງໄປຫາ AI API ເດີນທາງໄປຫາ server ຂອງຜູ້ໃຫ້ບໍລິການ; ຖ້າມັນມີຊື່ນັກຮຽນ ຫຼື ສະຖານະຜູ້ລົງໄພຢູ່ນຳ, module ການປະຕິບັດຕາມກົດລະບຽບ ນຳໃຊ້ເຕັມຮູບແບບ. **ສາມ: ຕົ້ນທຶນຕໍ່ຜູ້ໃຊ້ຕໍ່ເດືອນ ເມື່ອຜູ້ໃຊ້ເພີ່ມຂຶ້ນສິບເທົ່າ ແມ່ນເທົ່າໃດ?** ລາຄາ token ຂະຫຍາຍເປັນເສັ້ນຊື່ — ໂຄງການທົດລອງ \$30 ທີ່ສະບາຍໃຈ ຈະກາຍເປັນ \$300 ເມື່ອມັນສໍາເລັດ, ໂດຍບໍ່ມີສ່ວນຫຼຸດຕາມປະລິມານມາຊ່ວຍທ່ານ. ສາມເສົາຄ້າເກົາຄີເດີມ — ຄວາມເຊື່ອຖືໄດ້, ຄວາມປອດໄພ, ຕົ້ນທຶນ — ພຽງແຕ່ນຸ່ງຄໍາສັບໃໝ່. ເທັກໂນໂລຊີແມ່ນຂອງໃໝ່; ການສອບຖາມເຈາະເລິກບໍ່ແມ່ນ.

Capstone — ບົດຝຶກດຽວທີ່ປະກອບຫຼັກສູດທັງໝົດເຂົ້າກັນ

ບົດນີ້ຫຼັກສູດຈົບລົງດ້ວຍບົດຝຶກອອກແບບອັນດຽວ ທີ່ໃຊ້ທຸກຢ່າງໃນຫຼັກສູດ. ໂຈດ: ທ່ານຫາກໍຖືກແຕ່ງຕັ້ງເປັນຜູ້ນຳດ້ານ cloud ຂອງອົງກອນການກຸສົນດ້ານການສຶກສາ ໃນເຂດຊາຍແດນທີ່ແມ່ສອດ. ຊັບສິນຂອງມັນ — ເວັບໄຊສາທາລະນະ, database ບັນທຶກນັກຮຽນປະມານ 2,000 ຄົນ (ລວມທັງ, ສໍາລັບບາງຄົນ, ສະຖານະເອກະສານປະຈຳຕົວ), ສູນການຮຽນຮູ້ສາມແຫ່ງ, ແລະ ວິສະວະກອນທີ່ເກັ່ງແຕ່ອິດເມື່ອຍໜຶ່ງຄົນ ຜູ້ຖືການເຂົ້າເຖິງ production ພຽງຜູ້ດຽວ ແລະ ຖືໂທລະສັບເວນ on-call ພຽງເຄື່ອງດຽວ. ໃບບິນ cloud ເດືອນແລ້ວ ມາຮອດເກືອບສອງເທົ່າຂອງຈໍານວນປົກກະຕິ ແລະ ບໍ່ມີໃຜບອກໄດ້ວ່າຍ້ອນຫຍັງ. ວາລະດ້ານເທັກໂນໂລຊີອັນດຽວຂອງສະພາບລິຫານ: “ພວກເຮົາໃຊ້ AI ຊ່ວຍສອນເສີມໄດ້ບໍ່?” ໜ້າວຽກ: 90 ມື້ທໍາອິດຂອງທ່ານ ໃນໜຶ່ງໜ້າ, ພ້ອມເຫດຜົນຮອງຮັບການຈັດລໍາດັບ.

ສະບັບໂຕ້ຕອບ ໃຫ້ຄະແນນແຜນຕາມລາຍການກວດ: ສໍາຫຼວດຊັບສິນກ່ອນອອກຄວາມເຫັນ; ແຜນທີ່ສາມລາຍການ ຂອງເຄື່ອງ, ປ້າຍກໍາກັບ (labels), ແລະ ເງິນ; ໄຊຊະນະໄວໜຶ່ງອັນ; ການກວດກາຄວາມປອດໄພ ແລະ ການທົດສອບກູ້ຄືນຂໍ້ມູນສໍາຮອງ (backup-restore) ພ້ອມລະບຸວັນທີ; ສັນຍານເຕືອນງົບປະມານ (budget

tripwire); ທ່າທີທີ່ຊື່ສັດຕໍ່ຄວາມສ່ຽງຈາກການມີວິສະວະກອນຄົນດຽວ; ການລະບຸຊື່ PDPA ເປັນກອບສຳລັບ
ບັນທຶກກຮຽນ; ຄຳຖາມເລື່ອງ AI ທີ່ຕອບດ້ວຍສາມຄຳຖາມ AI; ແລະ ການທົບທວນກັບສະພາບສະພາບໃນມື້ທີ 90,
ລາຍງານເປັນຫົວໜ່ວຍ (unit terms).
