WORKSHOP 2 ชั่วโมง · AI OBJECT DETECTION · CLAUDE CODE

Smart Intrusion Detection with YOLO26

ระบบตรวจจับผู้บุกรุกอัจฉริยะด้วย YOLO26 — ตรวจจับคนและรถ
แล้วแจ้งเตือนทางอีเมลพร้อมภาพและคลิป 8 วินาที



ออกแบบหลักสูตรโดยสถาบัน ZenityX

Format
Workshop 2 ชั่วโมง · สอนทีละ Prompt
วาง Prompt → AI เขียนโค้ด → รันดูผลทันที

Duration
120 นาที · 9 ช่วงการเรียน
ปูพื้น AI Detection แล้วลงมือสร้างระบบทีละเฟส

Core Tool
Claude Code + YOLO26
Python · OpenCV · Ultralytics · ffmpeg · Gmail SMTP

Core Output
ระบบกล้องแจ้งเตือนผู้บุกรุก
อีเมลพร้อมภาพ + คลิป 8 วินาที (ก่อนเหตุ 3 วิ หลังเหตุ 5 วิ)

1. ภาพรวมหลักสูตร

Workshop 2 ชั่วโมงที่ผู้เรียนสร้างระบบกล้องวงจรปิดอัจฉริยะได้เองตั้งแต่โฟลเดอร์ว่าง โดยใช้ **Claude Code** เป็น AI Agent เขียนโค้ด Python + OpenCV + **YOLO26** ให้ทีละเฟส ผู้เรียนเพียงวาง Prompt ที่เตรียมไว้ แล้วรันดูผลได้ทันที

เส้นเรื่องของ Workshop

จากกล้องที่ “บันทึก” แต่ไม่มีใครดู → ระบบที่รู้ว่า “อะไร” เข้ามา แล้วส่งหลักฐานถึงมือถือทันที



กล้องวงจรปิดทั่วไป

บันทึก แต่ไม่มีใครดู

รู้ตัวเมื่อเสียหายไปแล้ว

Motion Detection

เตือนทุกอย่างที่ขยับ

ใบไม้ เงา แมว — เตือนหลอกจนคนเลิกสนใจ

AI Detection

รู้ว่า “อะไร” ขยับ

แจ้งเฉพาะคน/รถ ในโซนที่สนใจ ในเวลาที่ไม่ควรมีใคร

ใช้งานได้กับ บ้านกลางคืน · ร้านค้าหลังปิด · โกดัง · ลานจอดรถ · ห้อง server · แปลงเกษตร

2. วัตถุประสงค์การเรียนรู้

เมื่อจบ Workshop ผู้เรียนจะทำสิ่งเหล่านี้ได้ด้วยตัวเอง บนเครื่องของตัวเอง

01 **อธิบาย AI Object Detection ได้**
แยกจาก Classification และ Segmentation ได้

02 **เข้าใจว่า YOLO26 ทำงานอย่างไร**
มองครั้งเดียว ทายกรอบ คลาส confidence

03 **ใช้ Claude Code สร้างระบบเองทีละเฟส**
สร้างด้วย Python + OpenCV + YOLO26

04 **เลือกขนาดโมเดลให้เหมาะกับเครื่อง**
เทียบ YOLO26 n / s / m ด้วย benchmark

05 **กำหนดช่วงเวลา โซน และคลาสได้**
เฝ้าระวังเฉพาะสิ่งที่สนใจ และลดการแจ้งเตือนหลอก

06 **ส่งอีเมลแจ้งเตือนพร้อมหลักฐาน**
ภาพ + คลิป 8 วินาที ผ่าน Gmail App Password

3. กลุ่มผู้เรียนและพื้นฐาน

กลุ่มผู้เรียน

ผู้เริ่มต้นที่อยากใช้ AI ทำระบบจริง

Claude Code เป็นผู้เขียนโค้ด
ผู้เรียนสั่งงาน รันระบบ และตรวจผลลัพธ์

- ✓ เจ้าของกิจการ ร้านค้า และผู้ดูแลอาคาร
- ✓ นักศึกษาและครูสายคอมพิวเตอร์
- ✓ ไม่ต้องเขียนโค้ดเป็นก็เรียนได้

พื้นฐานที่ต้องมี

ติดตั้ง Claude Code + Python มาก่อนเข้าคลาส

- ✓ โน้ตบุ๊ก Mac หรือ Windows (RAM 8 GB+)
- ✓ บัญชี Gmail ที่เปิด 2-Step Verification

ไม่จำเป็นต้องมี

- ✗ GPU — YOLO26n รันบน CPU ได้
- ✗ กล้องจริง — ในคลาสใช้คลิปวิดีโอ

Work Style สอนทีละ Prompt · เริ่มจากโฟลเดอร์ว่าง + คลิปทดสอบที่แจกให้

1. **วาง Prompt ใน Claude Code** — Prompt พร้อมใช้ทุกเฟส
2. **AI เขียนโค้ดให้** — เขียนและแก้ไฟล์ให้
3. **รันดูผลทันที** — เห็นกรอบบนวิดีโอจริง

4. ผลลัพธ์หลังเรียน

ระบบที่ทำงานได้จริงบนเครื่องของผู้เรียน พร้อมต่อยอดกับกล้องจริงผ่าน webcam หรือ RTSP



01

ตรวจจับคนและรถ

detector.py แสดงกรอบ จำนวน FPS จากไฟล์ webcam หรือ RTSP

02

ผล benchmark n / s / m

ตารางเทียบขนาดโมเดล เลือกรุ่นที่เหมาะกับเครื่อง

03

ช่วงเวลา + โซนเฝ้าระวัง

ARMED 22:00–06:00 และโซนที่วาดเองบนภาพจริง

04

กรองคลาส กันเตือนหลอก

person · car · motorcycle
ยืนยัน 3 เฟรม + cooldown 60 วิ

05

หลักฐานทุกเหตุการณ์

snapshot + คลิป 8 วิ (H.264) + event.json

06

อีเมลแจ้งเตือนจริง

Gmail SMTP ภาพในตัว + ไฟล์แนบ

ไฟล์โปรเจกต์ที่ได้กลับไป

`detector.py` ตัวระบบหลัก	`config.yaml` ไฟล์ตั้งค่าระบบ	`benchmark.py` เทียบขนาดโมเดล	`.env` เก็บรหัสลับ	`CLAUDE.md` บันทึกโปรเจกต์

5. ตารางเวลา

120 นาที — ปูพื้นความเข้าใจสั้น ๆ แล้วใช้เวลาส่วนใหญ่ลงมือสร้างระบบทีละเฟส จนส่งอีเมลแจ้งเตือนได้จริง

25 นาที	10	75 นาที	10
เปิดคลาส + ทฤษฎี	Setup	สร้างระบบ 5 เฟส	รันเต็ม

0:00–0:10	INTRO	**เปิดคลาส** แนะนำผู้สอน · ดูผลลัพธ์ปลายทาง (อีเมลแจ้งเตือนจริง) · เป้าหมายวันนี้
0:10–0:25	LECTURE	**AI Detection & YOLO26** Detection คืออะไร · YOLO26 คืออะไร ทำไมต้องใช้ ทำงานอย่างไร · ช่วยกันขโมยอย่างไร
0:25–0:35	SETUP · S0	**เครื่องมือ + Setup** Claude Code · Python · OpenCV · Ultralytics/YOLO26 · ffmpeg · Gmail
0:35–0:50	PHASE 1 · P1	**ตรวจจับคน** แหล่งภาพ 3 แบบ: ไฟล์วิดีโอ / webcam / RTSP
0:50–1:00	PHASE 2 · P2	**เลือกขนาดโมเดล** เทียบ n / s / m · เสริม P2B ลองงาน segmentation และ pose
1:00–1:15	PHASE 3 · P3	**ช่วงเวลา + โซน** เฝ้าระวัง 22:00–06:00 · วาดโซนบนภาพจริง
1:15–1:25	PHASE 4 · P4	**กรองคลาส** เลือกจาก 80 คลาส · กันแจ้งเตือนหลอก
1:25–1:50	PHASE 5 · P5	**หลักฐาน + อีเมล** P5A เก็บภาพ + คลิป · สร้าง App Password · P5B ส่งอีเมล
1:50–2:00	RUN	**รันระบบเต็ม + ต่อยอด** Bonus RTSP / แผงควบคุมเว็บ · ถาม-ตอบ

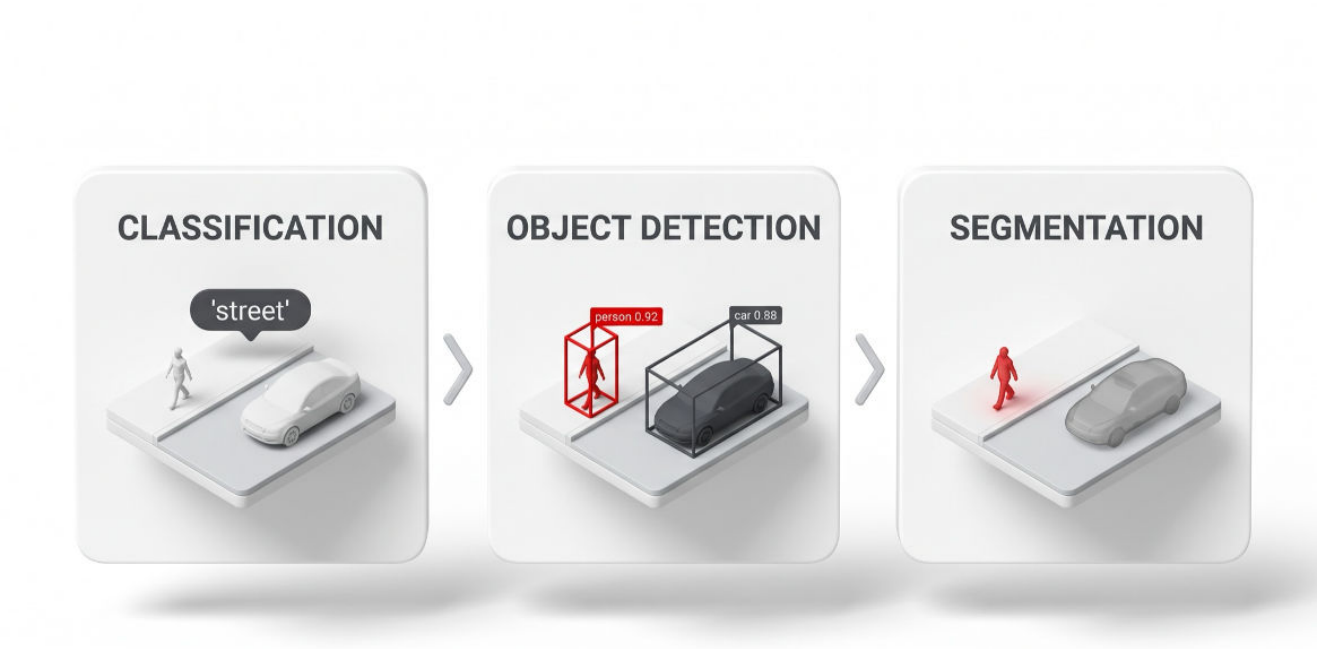
เวลาปรับได้ตามความเหมาะสม — ตารางเวลาเป็นแนวทาง ผู้สอนอาจปรับเวลาแต่ละช่วงให้เหมาะกับความคืบหน้าของผู้เรียน โดยเนื้อหาและผลลัพธ์หลักยังครบถ้วน

S0 · P1 – P5B · RUN คือรหัส Prompt ของแต่ละช่วง ผู้เรียนคัดลอกไปวางใน Claude Code ได้ทันที

LECTURE 0:10–0:25 · 15 นาที

AI Detection คืออะไร

Computer Vision มี 3 ระดับ — Workshop นี้ใช้ระดับกลาง คือ **Object Detection** — มีอะไร อยู่ตรงไหน กี่ชิ้น



LEVEL 1
Classification
ภาพนี้คืออะไร

LEVEL 2 · ใช้ในคลาส
Object Detection
มีอะไร อยู่ตรงไหน กี่ชิ้น

LEVEL 3
Segmentation
รูปร่างของวัตถุละเอียดทีละพิกเซล

ผลลัพธ์ของ Detection ต่อ 1 วัตถุ

กรอบ
`(x, y, w, h)` ตำแหน่งในภาพ

คลาส
`person` วัตถุนั้นคืออะไร

ความมั่นใจ
`confidence 0.87`

วิดีโอ = ภาพนิ่งหลายสิบภาพต่อวินาที
ระบบจึงตรวจจับได้ทุกเฟรมแบบ real-time ซึ่งเป็นหัวใจของกล้องที่แจ้งเตือนได้เอง

LECTURE 0:10–0:25 · 15 นาที

YOLO26 คืออะไร ทำงานอย่างไร

YOLO = You Only Look Once — มองภาพครั้งเดียว ทายทุกวัตถุพร้อมกัน จึงเร็วพอสำหรับวิดีโอสด



จากภาพเข้าสู่ผลลัพธ์ ในภาพเดียวจบ

01 · Backbone	→	02 · Neck	→	03 · Head
ดึงลักษณะเด่น		**รวมลักษณะหลายขนาด**		**ทายผลทันที**
ขอบ รูปร่าง พื้นผิว		ทั้งของใหญ่และของเล็ก		กรอบ + คลาส + confidence

รุ่นล่าสุดของ Ultralytics

ออก มกราคม 2026 เร็วบนเครื่องทั่วไปและอุปกรณ์ edge

COCO 80 คลาส

คน รถยนต์ มอเตอร์ไซค์ รถบรรทุก จักรยาน ฯลฯ ใช้ได้ทันทีไม่ต้องเทรน

ทำได้หลายงาน

Detection · Segmentation · Pose · Classification · OBB (กรอบเอียง)

LECTURE 0:10–0:25 · 15 นาที

ทำไมต้อง YOLO26 และเลือกขนาดไหน



01 End-to-end ไม่ต้อง NMS
ทายกรอบสุดท้ายได้เลย ไม่ต้องกรองกรอบซ้ำ ขั้นตอนน้อย นำไปใช้ง่าย

02 เร็วบน CPU
ผู้พัฒนารายงานว่า YOLO26n เร็วกว่า YOLO11n บน CPU สูงสุดราว 43%

03 เบา export ง่าย
ตัด DFL ออก ส่งออก ONNX / TensorRT / CoreML / OpenVINO ได้

04 จับวัตถุเล็กดีขึ้น (STAL)
เหมาะกับกล้องวงจรปิดมุมสูงที่คนในภาพดูตัวเล็ก

3 LINES `from ultralytics import YOLO` → `model = YOLO("yolo26n.pt")` → `model(frame)`

5 ขนาดของ YOLO26 (COCO 640px · ข้อมูล Ultralytics)

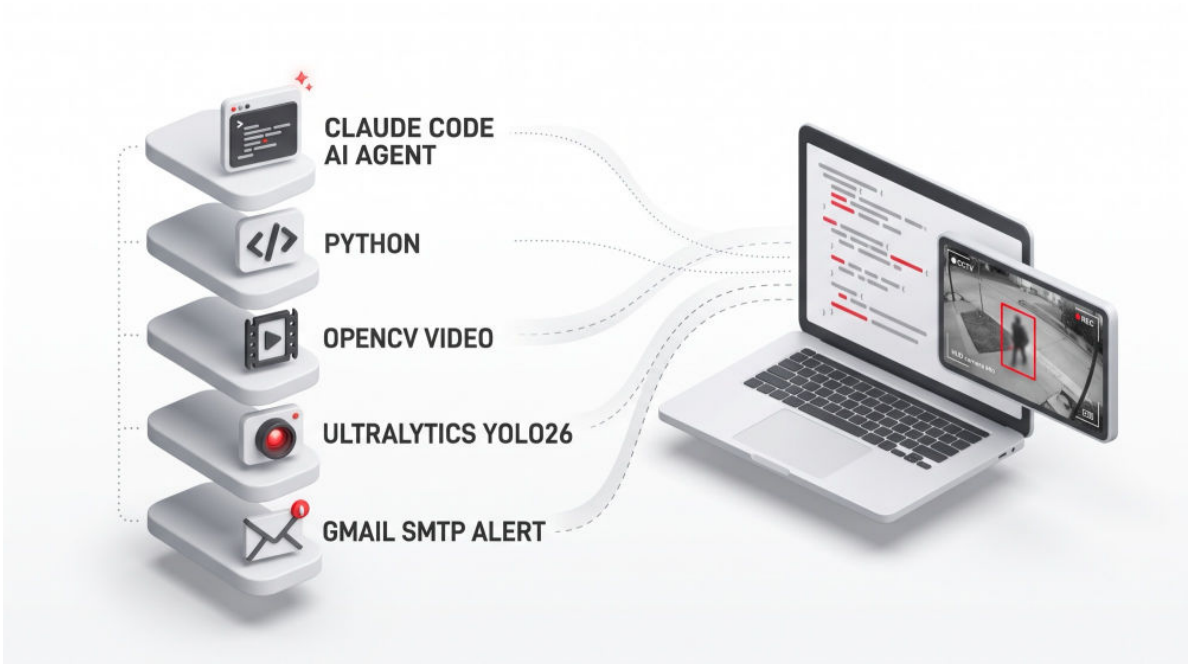
รุ่น	mAP50-95	CPU ONNX (ms)	Params (M)	ในคลาส
YOLO26n	40.9	38.9	2.4	เทียบใน Phase 2
YOLO26s	48.6	87.2	9.5	เทียบใน Phase 2
YOLO26m	53.1	220.0	20.4	เทียบใน Phase 2
YOLO26l	55.0	286.2	24.8	—
YOLO26x	57.5	525.8	55.7	—

SETUP 0:25–0:35 · 10 นาที

>_ Prompt S0

เครื่องมือวันนี้ และทำไมต้องใช้

เตรียมโปรเจกต์: สร้าง `.venv` ติดตั้งไลบรารี ตรวจคลิปทดสอบ และสร้าง `CLAUDE.md` ให้ AI จำบริบทโปรเจกต์



เครื่องมือ	บทบาท	ทำไมใช้
Claude Code	AI Agent เขียนโค้ด รันโค้ด แก้ error ให้ในโฟลเดอร์โปรเจกต์	สั่งเป็นภาษาไทย ไม่ต้องเขียนโค้ดเอง
Python 3.10+	ภาษาหลักของ AI	ไลบรารี AI ครบที่สุด
OpenCV	อ่านวิดีโอ/กล้อง วาดกรอบ บันทึกคลิป	ต่อ webcam / RTSP ได้
Ultralytics + YOLO26	“สมอง” ตรวจจับวัตถุ	โมเดลล่าสุด เร็ว ใช้ 3 บรรทัด
imageio-ffmpeg	แปลงคลิปเป็น H.264	เปิดได้ทุกเครื่องและมือถือ
Gmail SMTP + App Password	ส่งอีเมลแจ้งเตือน	ฟรี ปลอดภัยกว่าใช้รหัสจริง
python-dotenv (.env)	เก็บรหัสแยกจากโค้ด	รหัสไม่หลุดไปกับโค้ด

Claude Code
วิศวกรที่เราสั่งงาน

Ultralytics
กล่องเครื่องมือ

YOLO26
สมองที่มองเห็น

OpenCV
ตาและมือ

Gmail
ผู้ส่งสาร

PHASE 1 0:35–0:50 · 15 นาที

>_ Prompt P1

ตรวจจับคนด้วย YOLO26

ให้ Claude Code เขียน `detector.py` + `config.yaml` ที่อ่านวิดีโอ ตรวจจับคนทุกเฟรม แล้ววาดกรอบให้เห็นทันที



ภาพจริงจากระบบที่สร้างในคลาส — กรอบตรวจจับ จำนวนคน และ FPS

01 detector.py + config.yaml
ไฟล์ระบบหลัก และไฟล์ตั้งค่า (source · model · confidence 0.5)

02 กรอบแดง + จำนวนคน + FPS
แสดงบนวิดีโอทุกเฟรม พร้อมป้าย เช่น person 0.87

03 แหล่งภาพ 3 แบบ
ไฟล์วิดีโอ · `0` = webcam · `rtsp://` = กล้องวงจรปิด

04 เลือกอุปกรณ์ประมวลผลอัตโนมัติ
cuda › mps (Mac) › cpu — เครื่องทั่วไปก็รันได้

>_ **AI ทดสอบให้ก่อน แล้วผู้เรียนรันเอง**
Claude Code ทดสอบด้วย `--no-window` ก่อน แล้วบอกคำสั่งให้ผู้เรียนรันแบบเปิดหน้าต่าง (กด q เพื่อออก) พร้อมอธิบายโค้ดทีละส่วนสั้น ๆ

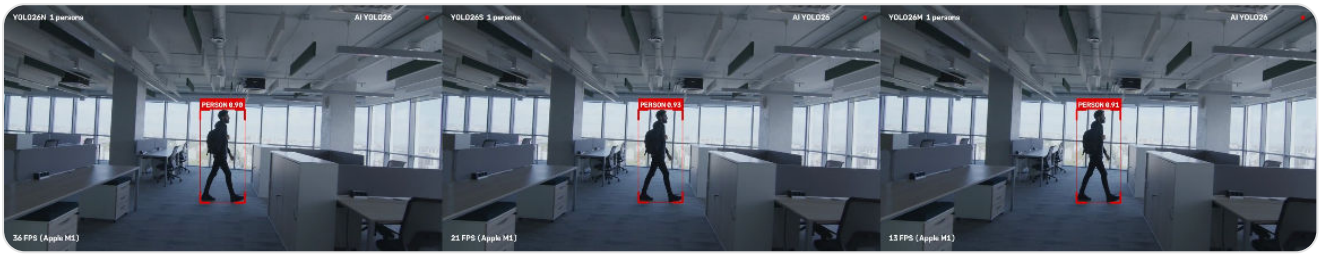
ชิ้นงานที่ได้
detector.py ตรวจจับคนได้จริง
เห็นกรอบตรวจจับ จำนวนคน และ FPS บนคลิปทดสอบที่แจกให้

PHASE 2 0:50–1:00 · 10 นาที

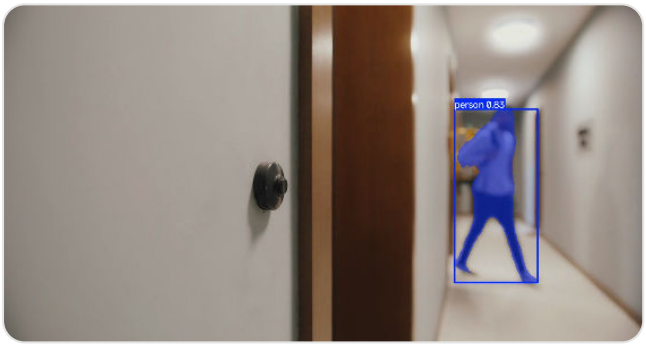
`>_ Prompt P2 + P2B`

เลือกขนาดโมเดลให้เหมาะกับเครื่อง

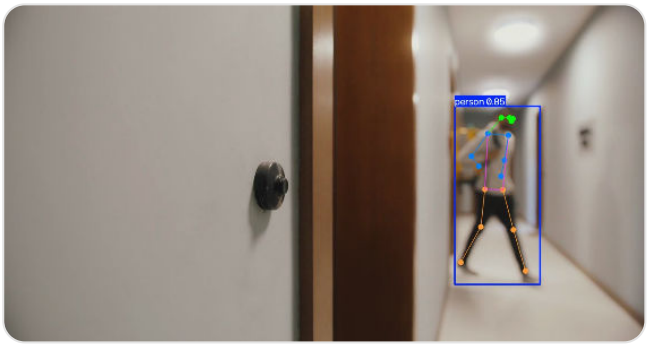
เพิ่มตัวเลือก `--model` แล้วให้ `benchmark.py` วัดผล YOLO26 n / s / m บนเครื่องของผู้เรียนเอง



benchmark.jpg — ผลเทียบ n / s / m จากระบบจริง



P2B · Segmentation — รูปร่างทีละพิกเซล



P2B · Pose — จุดข้อต่อร่างกาย

01 เพิ่ม --model
สลับ n / s / m ได้จากคำสั่ง ไม่ต้องแก้โค้ด

02 benchmark.py
ตารางเทียบ n / s / m + ภาพ benchmark.jpg

03 เสริม P2B
ลองงาน segmentation และ pose ของ YOLO26

04 หลักเลือกขนาด
FPS ต่ำ → ใช้ yolo26n หรือลด imgsz เป็น 480

YOLO26N
เร็วที่สุด
38.9 ms บน CPU · mAP 40.9

YOLO26S
ตรงกลาง
87.2 ms บน CPU · mAP 48.6

YOLO26M
แม่นที่สุดใน 3 รุ่น
220.0 ms บน CPU · mAP 53.1

ชิ้นงานที่ได้
ผล benchmark บนเครื่องตัวเอง
รู้ว่าเครื่องนี้ควรใช้ YOLO26 ขนาดไหน ระหว่างความเร็วกับความแม่น

PHASE 3 1:00–1:15 · 15 นาที

>_ Prompt P3

กำหนดช่วงเวลาและโซนเฝ้าระวัง

ระบบที่ดีเฝ้าเฉพาะ **เวลาที่ไม่ควรมีใคร** และเฉพาะ **พื้นที่ที่สนใจ** — ภาพยังแสดงตลอด แต่นับเป็นเหตุการณ์เฉพาะเมื่อเข้าเงื่อนไข



ภาพจริงจากระบบ — โซนลานจอดรถ สถานะ ARMED 22:00–06:00

01 ช่วงเวลา 22:00–06:00
รองรับช่วงข้ามเที่ยงคืน · ทดสอบด้วย `--now HH:MM`

02 สถานะ ARMED / DISARMED
ป้ายแดงเมื่อเฝ้าอยู่ · ป้ายเทาเมื่ออยู่นอกช่วงเวลา

03 วาดโซนด้วย --draw-zone
คลิกเมาส์วาดรูปหลายเหลี่ยมบนเฟรมแรก แยกตามกล้อง

04 นับเฉพาะ “จุดเท้า” ในโซน
ในโซนกรอบสีแดง นอกโซนกรอบสีเทา

ผู้เรียนลองวาดโซนเอง
ฝึกกับคลิป `13_cctv_carport_motorcycle` · คลิปหรือกล้องที่ยังไม่มีโซน ระบบถือว่าทั้งภาพเป็นโซน

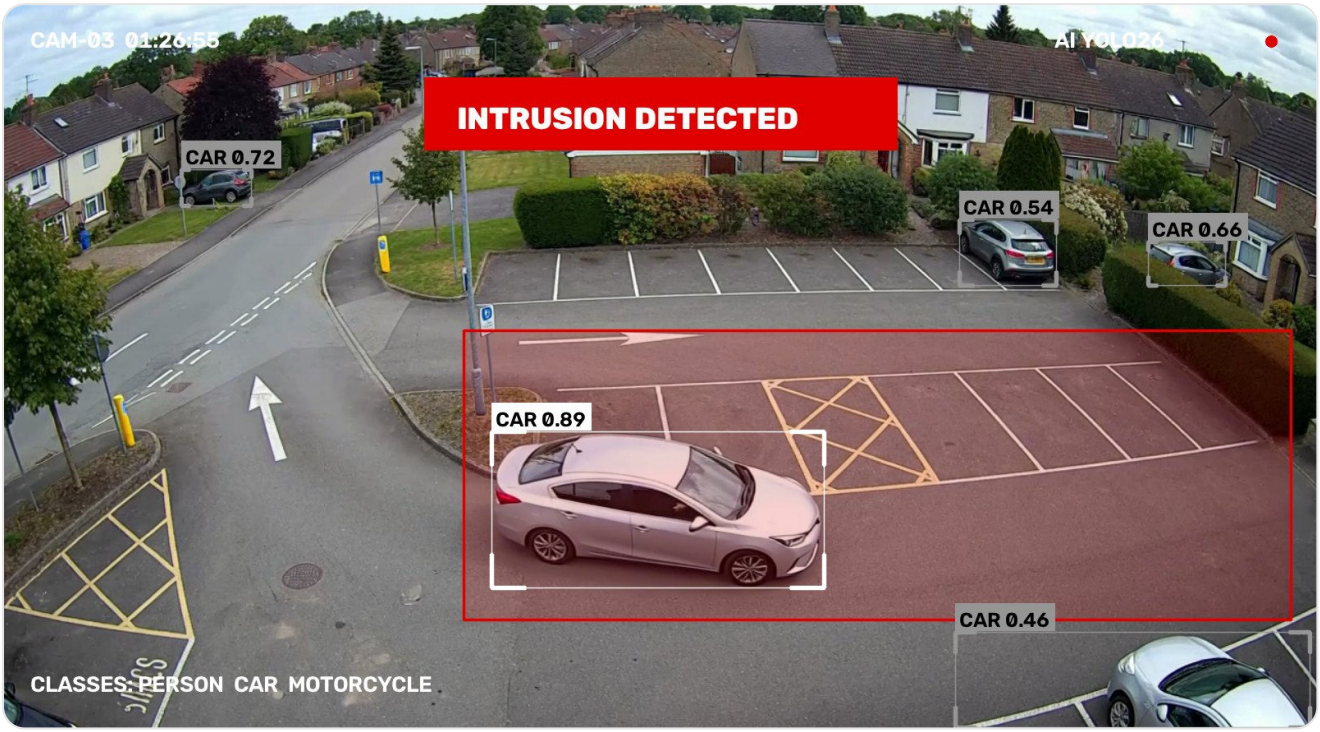
ชิ้นงานที่ได้
ระบบเฝ้าระวังตามเวลาและโซน
โซนบันทึกใน config.yaml ใช้ได้กับทุกความละเอียดของภาพ

PHASE 4 1:15–1:25 · 10 นาที

>_ Prompt P4

กรองคลาส กันแจ้งเตือนหลอก

YOLO26 รู้จัก 80 คลาส — เลือกเฉพาะสิ่งที่ต้องเฝ้าระวัง แล้วยืนยันด้วยจำนวนเฟรมก่อนนับเป็นเหตุการณ์



ภาพจริงจากระบบ — ตรวจพบรถยนต์ในโซน ขึ้นป้าย INTRUSION DETECTED

01 เลือกคลาส
`[person, car, motorcycle]` ใน config.yaml

02 --list-classes
ดูรายชื่อ 80 คลาสพร้อมชื่อภาษาไทย

03 สีต่อคลาส
แยกคน รถยนต์ มอเตอร์ไซค์ ให้อ่านง่ายบนจอ

04 เจอติดกัน 3 เฟรม + cooldown 60 วิ
ไม่แจ้งซ้ำรัว ๆ และไม่เตือนจากภาพหลุดเฟรมเดียว

ยังเตือนหลอกบ่อย?
เพิ่ม confidence เป็น 0.6 · วาดโซนให้แคบลง · ปรับ min_consecutive_frames เป็น 5

PHASE 5 · 1:25–1:50 · 25 นาที

`>_ Prompt P5A + P5B`

เก็บหลักฐาน และส่งอีเมลแจ้งเตือน

เมื่อเกิดเหตุ ระบบบันทึกภาพและคลิปที่มีช่วง **ก่อนเหตุ** ด้วย แล้วส่งถึงอีเมลพร้อมภาพในตัวและไฟล์แนบ



ภาพจริงจากระบบ — กลางคืน IR · EVENT SAVED snapshot + clip.mp4 (8 วิ)

ก่อนเหตุ 3 วิ
จาก buffer ย้อนหลัง

⚠ เหตุการณ์

หลังเหตุ 5 วิ
บันทึกต่อจนครบ

P5A · EVIDENCE

เก็บหลักฐาน

- ✓ snapshot + คลิป 8 วิ แปลงเป็น H.264 ด้วย imageio-ffmpeg
- ✓ เปิดได้ทุกเครื่องและบนมือถือ
- ✓ บันทึก `events/…/event.json` ทุกเหตุการณ์

P5B · EMAIL

ส่งอีเมลแจ้งเตือน

- ✓ Gmail SMTP พอร์ต 465 SSL · รหัสเก็บใน .env
- ✓ อีเมล HTML ภาษาไทย ภาพในตัว + แนบไฟล์
- ✓ `EMAIL_SEND=false` → เก็บเป็น email.eml

ชิ้นงานที่ได้

อีเมลแจ้งเตือนถึงกล่องจดหมายจริง

ภาพเหตุการณ์ในตัวอีเมล + คลิป 8 วินาทีเป็นไฟล์แนบ ทดสอบด้วย --test-email

PHASE 5 1:25–1:50 · 25 นาที

สร้าง Google App Password ทีละขั้น

ผู้เรียนทำเองบนบัญชี Gmail ของตัวเอง ระหว่าง P5A กับ P5B — ระบบใช้รหัสนี้แทนรหัส Gmail จริง

1 เปิดการยืนยันแบบ 2 ขั้นตอน
ที่ `myaccount.google.com/security` เปิด **2-Step Verification** ถ้ายังไม่เปิด

2 เปิดหน้า App Passwords
ที่ `myaccount.google.com/apppasswords` (ล็อกอินอีกครั้งถ้าถูกถาม)

3 ตั้งชื่อแอป แล้วกดสร้าง
เช่น `YOLO26 Alert` → กด **สร้าง**

4 คัดลอกรหัส 16 ตัว
รหัสแสดงครั้งเดียว · เว้นวรรคหรือไม่ก็ได้

5 พิมพ์ลงไฟล์ .env เอง
เปิดไฟล์ `.env` ในโปรเจกต์ แล้วใส่ค่าตามกล่องด้านขวา

6 ทดสอบส่งอีเมล
รัน `.venv/bin/python detector.py --test-email` แล้วเช็คกล่องจดหมาย

.ENV

```
SMTP_USER=อีเมลของคุณ
SMTP_PASSWORD=รหัส16ตัว
EMAIL_TO=อีเมลผู้รับ
EMAIL_SEND=true
```

WINDOWS

`.venv\Scripts\python detector.py --test-email`

ห้ามวางรหัสในแชท AI
พิมพ์ลง .env เองเท่านั้น รหัสจึงไม่หลุดไปกับโค้ดหรือแชท

ถ้าติดปัญหา

ไม่เห็นเมนู App Password
ยังไม่เปิด 2-Step หรือเป็นบัญชีองค์กร (Google Workspace) ที่ปิดเมนูนี้ไว้

ใช้รหัส Gmail ปกติไม่ได้
ต้องเป็น App Password เท่านั้น — ถ้าเจอ `535` ให้สร้างรหัสใหม่

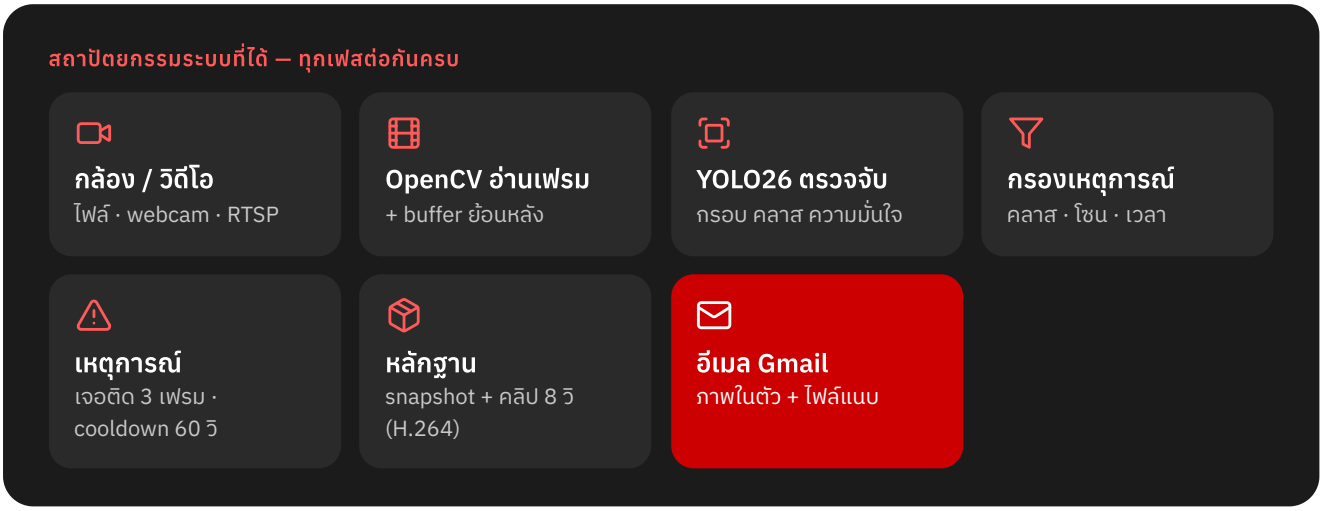
ยกเลิกได้ทุกเมื่อ
ลบ App Password ได้ที่หน้าเดิมเมื่อเลิกใช้ระบบ

RUN + BONUS · 1:50–2:00 · 10 นาที

>_ Prompt RUN

รันระบบเต็ม และต่อยอด

รันทุกเฟสรวมกันกับคลิปกลางคืน IR แล้วให้ Claude Code อัปเดต `CLAUDE.md` ว่าระบบทำอะไรได้บ้าง และรันด้วยคำสั่งอะไร



BONUS B1

ต่อกล้องจริง

RTSP / webcam · อ่าน URL กล้องจาก .env · ต่อใหม่เองเมื่อหลุด

BONUS B2

แผงควบคุมบนเว็บ

ผู้สอนโชว์ demo แผงควบคุม Flask: ดูภาพสด เลือกคลาส/โมเดล ดูเหตุการณ์

Q&A

ถาม-ตอบ

ทบทวนระบบที่ได้ และแนวทางต่อยอดกับสถานที่จริงของผู้เรียน

แก้ปัญหาที่พบบ่อย

อาการ	สาเหตุ	วิธีแก้
`ModuleNotFoundError: ultralytics`	รันด้วย Python นอก .venv	ใช้ `.venv/bin/python` (Windows `.venv\Scripts\python`)
FPS ต่ำมาก	ใช้โมเดลใหญ่บน CPU	ใช้ yolo26n · ลด imgsz เป็น 480
ไม่มีเหตุการณ์เลย	อยู่นอกช่วงเวลา (DISARMED)	ตั้ง schedule ให้ครอบเวลาปัจจุบัน
คลิปเปิดในมือถือไม่ได้	ยังเป็น mp4v	แปลง H.264 ด้วย imageio-ffmpeg (P5A)
webcam เปิดไม่ได้บน Mac	ยังไม่ให้สิทธิ์กล้อง	System Settings → Privacy → Camera
RTSP ต่อไม่ได้	URL/รหัสผิด หรือคนละวง LAN	ทดสอบ URL ใน VLC ก่อน · ใช้ sub stream

7. ขอบเขตการสอน

กำหนดชัดตั้งแต่ต้นว่า 2 ชั่วโมงนี้สอนอะไร และอะไรอยู่นอกขอบเขต

✓ อยู่ในขอบเขต

- พื้นฐาน AI Object Detection และ YOLO26
- ใช้ Claude Code สร้างระบบทีละเฟส
- ตรวจจับจากไฟล์วิดีโอ webcam และ RTSP
- กำหนดช่วงเวลา โซน และคลาส
- หลักฐานภาพ + คลิป 8 วินาที
- อีเมลแจ้งเตือนผ่าน Gmail App Password

✕ นอกขอบเขต

- การเทรนโมเดลด้วยข้อมูลของตัวเอง (ใช้โมเดล COCO 80 คลาสที่พร้อมใช้)
- การติดตั้งกล้องและเดินระบบเครือข่ายจริง
- การนำระบบขึ้น server หรือ cloud
- การเขียนโค้ดเชิงลึกด้วยตัวเอง

8. การประเมินผลในคลาส

ประเมินจากระบบที่รันได้จริงบนเครื่องของผู้เรียน — ไม่มีข้อสอบ

- ✓ detector.py แสดงกรอบ จำนวน และ FPS บนคลิปทดสอบได้
- ✓ benchmark n / s / m รันได้ และอธิบายได้ว่าควรใช้รุ่นไหน
- ✓ ระบบเฝ้าเฉพาะช่วงเวลาและโซนที่กำหนด (ARMED / DISARMED)
- ✓ แจ้งเหตุเฉพาะคลาสที่เลือก หลังเจอติดกัน 3 เฟรม
- ✓ `--test-email` ส่งถึงกล่องจดหมายจริง พร้อมภาพและคลิป 8 วินาที

หลักความปลอดภัยของรหัส

รหัสอยู่ใน .env
แยกจากโค้ด ไม่ปนอยู่ในโค้ด

ไม่วางในแชท AI
ผู้เรียนพิมพ์รหัสลงไฟล์เอง

ยกเลิกได้ทุกเมื่อ
ลบ App Password ได้ที่หน้าเดิม

9. สิ่งที่ผู้เรียนต้องเตรียม

เตรียมให้พร้อมก่อนวันเรียน เพื่อให้ทุกช่วงของ 2 ชั่วโมงได้ลงมือสร้างระบบจริง

1 **โน้ตบุ๊ก Mac หรือ Windows (RAM 8 GB+)**
ไม่ต้องมี GPU — YOLO26n รันบน CPU ได้

2 **ติดตั้ง Claude Code และล็อกอินแล้ว**
แพ็กเกจ Pro ขึ้นไป

3 **Python 3.10+**
ติดตั้งให้เรียบร้อยก่อนเข้าคลาส

4 **บัญชี Gmail ที่เปิด 2-Step Verification**
ใช้สร้าง App Password ในช่วง Phase 5

5 **ดาวน์โหลดคลิปทดสอบจาก Hub**
ไฟล์ `yolo26-test-videos.zip`

6 **อินเทอร์เน็ต**
สำหรับติดตั้งไลบรารีและดาวน์โหลดโมเดล

ไม่ต้องมีกล้องจริง
ใช้คลิปทดสอบ แล้วต่อกล้องจริงได้ภายหลัง

ไม่ต้องเขียนโค้ดเป็น
Claude Code เขียนโค้ดให้ ผู้เรียนสั่งงาน

คลิปทดสอบ 7 คลิปที่แจกให้ — เริ่มจากโฟลเดอร์ว่าง + คลิปเหล่านี้

`01_door_entry` ทางเข้าประตู	`02_hallway_door` ทางเดิน
`04_office_wide` สำนักงานมุมกว้าง	`11_cctv_office_day` สำนักงานกลางวัน
`12_cctv_office_night_ir` สำนักงานกลางคืน IR	`13_cctv_carport_motorcycle` โรงรถ + มอเตอร์ไซค์
`14_cctv_parking_car` ลานจอดรถยนต์	

10. ผู้สอน



INSTRUCTOR

เอกรัตน์ ใจชนะ (นิว)

AI Instructor, ZenityX

จุดเชื่อมกับ Workshop นี้

รางวัลชนะเลิศด้านนวัตกรรม คือผลงาน **“ระบบกล้องวงจรปิดตรวจจับอัจฉริยะ”** — Workshop นี้จึงสอนจากประสบการณ์ทำระบบนี้จริง

รางวัลและผลงาน

รางวัลชนะเลิศการนำเสนอบทความวิจัย แบบบรรยาย ระดับดีเด่น อันดับที่ 1
AUCC 2026 · Track Information Technology
การประชุมวิชาการระดับปริญญาตรีด้านคอมพิวเตอร์ภูมิภาคเอเชีย ครั้งที่ 14

รางวัลชนะเลิศ ประเภท Oral Presentation (ด้านนวัตกรรม)
ผลงาน “ระบบกล้องวงจรปิดตรวจจับอัจฉริยะ” — การประกวดผลงานสหกิจศึกษาและการศึกษาเชิงบูรณาการกับการทำงาน (CWIE) ดีเด่น เครือข่ายพัฒนา CWIE ภาคเหนือตอนล่าง

รางวัลระดับดีเด่น Best of Track
Track Information Technology — การนำเสนอผลงานวิจัย

PUBLICATION

Chaiwannakoop, K., **Jaichana, A.**, & Thunjai, S. (2026). *Home Visit Information System for Student Behavior Management.* Rattanakosin Journal of Science and Technology (RJST), 8(2), 152–166.

11. ปิดท้าย

หลังจบ 2 ชั่วโมง

ผู้เรียนกลับไปพร้อมระบบกล้องวงจรปิดอัจฉริยะ ที่ตรวจจับคนและรถ แล้วส่งหลักฐานถึงอีเมลได้เอง

YOLO26
ตรวจจับคนและรถ
แบบ real-time

เวลา · โซน · คลาส
เฉ้าเฉพาะสิ่งที่สนใจ

ภาพ + คลิป 8 วิ
หลักฐานทุกเหตุการณ์

อีเมลแจ้งเตือน
ผ่าน Gmail
App Password



Workshop 2 ชั่วโมง · 120 นาที
สอนทีละ Prompt · เริ่มจากโฟลเดอร์ว่าง
รันดูผลได้ทันทีทุกเฟส

Claude Code + YOLO26
Python · OpenCV · Ultralytics
imageio-ffmpeg · Gmail SMTP

ออกแบบหลักสูตรโดยสถาบัน ZenityX

ช่องทางติดต่อ

โทร
062-262-9644

อีเมล
admin@zenityxai.com