



คู่มือการเรียนรู้

ชุดทดลองสำหรับเรียนรู้ทางด้านการประมวลผลและวิเคราะห์สัญญาณ ภาพและวิดีโอด้วยเทคโนโลยีปัญญา

ประดิษฐ์

สำหรับ

ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยบูรพา

จัดทำโดย

บริษัท แรมสแตค จำกัด

27 พฤษภาคม 2565



สารบัญ

	หน้า
สารบัญ	1
LAB 1 Introduction to VAM Platform	1
VAM Platform	1
LAB 1.1 การติดตั้งและประกอบชุดทดลอง VAM Platform	2
Local area network (LAN)	2
IP Camera	4
1.1.1 วิธีที่ 1 การติดตั้งและประกอบชุดทดลองเข้ากับอุปกรณ์เสริม	4
อุปกรณ์ที่เกี่ยวข้อง	4
ขั้นตอนการติดตั้งและประกอบชุดการทดลอง	5
1.1.2 วิธีที่ 2 การติดตั้งและประกอบชุดทดลองเข้ากับเครื่องของผู้เรียน	9
อุปกรณ์ที่เกี่ยวข้อง	9
ขั้นตอนการติดตั้งและประกอบชุดการทดลอง	10
การตั้งค่าเครือข่ายอินเทอร์เน็ตภายในเครื่องคอมพิวเตอร์ของผู้เรียน	13
ระบบปฏิบัติการ Linux	13
ระบบปฏิบัติการ Windows	15
ระบบปฏิบัติการ Mac	18
LAB 1.2 การเข้าใช้งาน VAM Platform	20
สิทธิ์การใช้งานแพลตฟอร์ม	20
1.2.1 ขั้นตอนการเข้าสู่ระบบของ VAM Platform	21
1.2.2 Analytics Store	22
1.2.3 Assignment	23
1.2.4 System Configuration	24
1.2.4.1 Sources Management	24
ขั้นตอนการเข้าหน้าการจัดการข้อมูลนำเข้า	25
1.2.4.2 Users Management	25
ขั้นตอนการเข้าหน้าการจัดการบัญชีผู้ใช้	25

1.2.4 Resources	26
LAB 1.3 การจัดการข้อมูลนำเข้าประเภท RTSP	27
Real Time Streaming Protocol (RTSP)	27
ข้อมูลลงทะเบียนกล้อง Hikvision H.265+ Exir Fixed Cube Network Camera	28
1.3.1 ขั้นตอนการลงทะเบียนข้อมูลนำเข้า	29
1.3.2 ขั้นตอนการแก้ไขข้อมูลนำเข้า	32
1.3.3 ขั้นตอนการลบข้อมูลนำเข้า	33
LAB 1.4 การจัดการโปรแกรมการประมวลผลและวิเคราะห์ข้อมูล	34
1.4.1 ขั้นตอนการผูกกล้อง	34
1.4.2 ขั้นตอนการเปลี่ยนสถานะการทำงาน	36
1.4.3 ขั้นตอนการลบโปรแกรมการประมวลผล	38
1.4.4 ขั้นตอนการตั้งค่าพื้นที่ที่สนใจ	40
LAB 2 VAM Studio	43
VAM Studio	43
LAB 2.1 การเข้าใช้งาน VAM Studio	44
2.1.1 ขั้นตอนการเข้าใช้งานโปรแกรมบน VAM Studio	44
LAB 2.2 การนำตัวอย่างโปรแกรมเข้ามาใช้งานใน VAM Studio	48
2.2.1 ขั้นตอนการนำตัวอย่างโปรแกรมเข้ามาใช้งานใน VAM Studio	48
LAB 2.3 การใช้งานตัวอย่างโปรแกรมพื้นฐานของ OpenCV บน VAM Studio	52
OpenCV	52
พิกเซลและสี	52
Simple Thresholding	54
2.3.1 ขั้นตอนการใช้งานตัวอย่างโปรแกรมการอ่านและบันทึกรูปภาพ	56
2.3.2 ขั้นตอนการใช้งานตัวอย่างโปรแกรมการปรับภาพสีเทา	58
2.3.3 ขั้นตอนการใช้งานตัวอย่างโปรแกรมการตัดภาพเฉพาะส่วน	61
2.3.4 ขั้นตอนการใช้งานตัวอย่างโปรแกรมการปรับภาพใบหนารี	63
โฉม	65
LAB 3 VAM SDK	67
VAM SDK	67
LAB 3.1 พัฒนาสำหรับการรับข้อมูลกล้องจาก VAM Platform	68
3.2.1 พัฒนาสำหรับการรับข้อมูลของกล้องที่อยู่ใน VAM Platform	68
พัฒนาที่เกี่ยวข้อง	68

ขั้นตอนการใช้ตัวอย่างโปรแกรมการรับข้อมูลของกล้อง	69
3.2.2 พังก์ชันสำหรับการรับภาพจากกล้องที่อยู่ใน VAM Platform	71
พังก์ชันที่เกี่ยวข้อง	71
ขั้นตอนการนำตัวอย่างโปรแกรมการรับภาพจากกล้องมาใช้ใน VAM Studio	71
3.2.3 พังก์ชันสำหรับการรับข้อมูลพื้นที่ที่ส่งมาจาก VAM Platform	73
พังก์ชันที่เกี่ยวข้อง	73
ขั้นตอนการนำตัวอย่างโปรแกรมการรับข้อมูลพื้นที่ที่ส่งมาใช้ใน VAM Studio	74
LAB 3.2 พังก์ชันพื้นฐานสำหรับการประมวลผลและวิเคราะห์ข้อมูล	76
3.2.1 พังก์ชันการตรวจสอบบุคคล	76
พังก์ชันที่เกี่ยวข้อง	76
ขั้นตอนการนำตัวอย่างโปรแกรมการตรวจสอบบุคคลมาใช้ใน VAM Studio	76
3.2.2 พังก์ชันการตรวจสอบบุคคล	79
พังก์ชันที่เกี่ยวข้อง	79
ขั้นตอนการนำตัวอย่างโปรแกรมการตรวจสอบบุคคลมาใช้ใน VAM Studio	79
LAB 3.3 พังก์ชันสำหรับการส่งข้อมูลที่ประมวลผลเข้า VAM Platform	82
3.3.1 พังก์ชันการส่งภาพผลลัพธ์การประมวลผลไปที่ VAM Platform	82
พังก์ชันที่เกี่ยวข้อง	82
ขั้นตอนการนำตัวอย่างโปรแกรมการส่งภาพผลลัพธ์มาใช้ใน VAM Studio	82
3.3.2 พังก์ชันการส่งข้อมูลผลลัพธ์การประมวลผลไปที่ VAM Platform	84
พังก์ชันที่เกี่ยวข้อง	84
ขั้นตอนการนำตัวอย่างโปรแกรมการส่งผลลัพธ์มาใช้ใน VAM Studio	85
โจทย์	86
LAB 4 VAM AlaaS	88
RESTful API	88
VAM AlaaS	88
LAB 4.1 การติดตั้งและประกอบชุดทดลอง VAM Platform สำหรับ AlaaS	89
4.1.1 วิธีที่ 1 การติดตั้งและประกอบชุดทดลองเข้ากับอุปกรณ์เสริม	89
อุปกรณ์ที่เกี่ยวข้อง	89
ขั้นตอนการติดตั้งและประกอบชุดการทดลอง	90
4.1.2 วิธีที่ 2 การติดตั้งและประกอบชุดทดลองเข้ากับเครื่องของผู้เรียน	95
อุปกรณ์ที่เกี่ยวข้อง	95
ขั้นตอนการติดตั้งและประกอบชุดการทดลอง	96

LAB 4.2 โปรแกรมการลงทะเบียนภาพใบหน้า (Face Registration)	100
4.2.1 EnrollFace	100
ขั้นตอนการใช้ตัวอย่างโปรแกรมการลงทะเบียนใบหน้า	100
LAB 4.3 โปรแกรมการจดจำใบหน้า (Face Recognition)	103
4.3.1 FaceFeature	103
ขั้นตอนการใช้ตัวอย่างโปรแกรมการจดจำใบหน้า	103
LAB 4.4 โปรแกรมการตรวจจับใบหน้าและการตรวจสอบคุณลักษณะ (Face Detection and Quality Verification)	107
4.4.1 CompareFace	107
ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับใบหน้าและการตรวจสอบคุณลักษณะ	107
โจทย์	109
LAB 5 VAM Security & Surveillance	111
IP Camera Network Architecture	111
VAM Security & Surveillance	111
LAB 5.1 โปรแกรมตรวจจับผู้บุกรุก (Intrusion Detection)	112
5.1.1 ขั้นตอนการใช้โปรแกรมการตรวจจับผู้บุกรุกบน VAM Platform	112
5.1.2 intrusion-detection	113
ค่าที่เกี่ยวข้อง	113
ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับผู้บุกรุกบน VAM Studio	114
LAB 5.2 โปรแกรมการตรวจจับบุคคล (Human Detection)	117
5.2.1 ขั้นตอนการใช้โปรแกรมการตรวจจับบุคคลบน VAM Platform	117
5.2.2 human-detection	118
ค่าที่เกี่ยวข้อง	118
ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับบุคคลบน VAM Studio	119
LAB 5.3 โปรแกรมการตรวจจับสีชุดของบุคคล (People Suit Color Detection)	122
5.3.1 ขั้นตอนการใช้โปรแกรมการตรวจจับสีชุดของบุคคลบน VAM Platform	122
5.3.2 human-suit-color	123
ค่าที่เกี่ยวข้อง	123
ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับสีชุดของบุคคลบน VAM Studio	124
LAB 5.4 โปรแกรมการตรวจจับหน้ากากอนามัยบนใบหน้า (Face Mask Detection)	127
5.4.1 ขั้นตอนการใช้โปรแกรมการตรวจจับหน้ากากอนามัยบนใบหน้าบน VAM Platform	127
5.4.2 face-mask-detection	128
ค่าที่เกี่ยวข้อง	128

ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับหน้ากากอนามัยบนใบหน้าบน VAM Studio	129
LAB 5.5 โปรแกรมการนับบุคคลในพื้นที่ที่กำหนด (People Counting in ROI)	132
5.5.1 ขั้นตอนการใช้โปรแกรมการนับบุคคลในพื้นที่ที่กำหนดบน VAM Platform	132
5.5.2 people-counting-in-roi	133
ค่าที่เกี่ยวข้อง	133
ขั้นตอนการใช้ตัวอย่างโปรแกรมการนับบุคคลในพื้นที่ที่กำหนดบน VAM Studio	134
LAB 5.6 โปรแกรมการนับบุคคลเข้า-ออก (People Counting in-out)	137
5.6.1 ขั้นตอนการใช้โปรแกรมการนับบุคคลเข้า-ออกบน VAM Platform	137
5.6.2 people-counting-in-out	139
ค่าที่เกี่ยวข้อง	140
ขั้นตอนการใช้ตัวอย่างโปรแกรมการนับบุคคลเข้า-ออกบน VAM Studio	140
LAB 6 VAM Industry 4.0 - Machine Vision	144
Machine Vision	144
องค์ประกอบของระบบกล้องจักษุวิทัศน์	144
ประเภทการทำงานของกล้องจักษุวิทัศน์	145
VAM Industry 4.0 - Machine Vision	145
LAB 6.1 Identification	147
6.1.1 ฟังก์ชันการอ่าน OCR	147
ฟังก์ชันที่เกี่ยวข้อง	147
ขั้นตอนการใช้ตัวอย่างโปรแกรมการอ่าน OCR	148
6.1.2 ฟังก์ชันการอ่าน QRCode และ Barcode	151
ฟังก์ชันที่เกี่ยวข้อง	151
ขั้นตอนการใช้ตัวอย่างโปรแกรมการอ่าน QRcode	151
LAB 6.2 Inspection	153
6.2.1 ฟังก์ชันการตรวจจับรูปร่างวัตถุ (Shape Detection)	154
ฟังก์ชันที่เกี่ยวข้อง	154
ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับรูปร่างวัตถุ	155
6.2.2 ฟังก์ชันการตรวจจับเส้นขอบวัตถุ (Edge Detection)	158
ฟังก์ชันที่เกี่ยวข้อง	158
ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับเส้นขอบวัตถุ	159
6.2.3 ฟังก์ชันการตรวจนับจำนวนวัตถุ (Object Counting)	161
ฟังก์ชันที่เกี่ยวข้อง	161

ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจนับจำนวนวัตถุ	162
6.3.4 ฟังก์ชันการหาคุณภาพในภาพ (Color Pattern Matching)	164
ฟังก์ชันที่เกี่ยวข้อง	164
ขั้นตอนการใช้ตัวอย่างโปรแกรมเพื่อหาคุณภาพในภาพ	165
โจทย์	1
LAB 7 VAM Industrial Camera	168
Industrial Camera	169
LAB 7.1 การติดตั้งและประกอบชุดทดลองของกล้องอุตสาหกรรม	170
7.1.1 วิธีที่ 1 การติดตั้งและประกอบชุดทดลองเข้ากับอุปกรณ์เสริม	170
อุปกรณ์ที่เกี่ยวข้อง	170
ขั้นตอนการติดตั้งและประกอบชุดการทดลอง	171
7.1.2 วิธีที่ 2 การติดตั้งและประกอบชุดทดลองเข้ากับเครื่องของผู้เรียน	174
อุปกรณ์ที่เกี่ยวข้อง	174
ขั้นตอนการติดตั้งและประกอบชุดการทดลอง	175
LAB 7.2 การดึงข้อมูลภาพจากกล้องอุตสาหกรรม	179
7.2.1 ขั้นตอนการใช้ตัวอย่างโปรแกรมการดึงภาพจากกล้องอุตสาหกรรม	179
โจทย์	181
LAB 8 Object Classification	181
Object Classification	182
Diagram Object Classification	182
8.1 ขั้นตอนการใช้ตัวอย่างโปรแกรมการจำแนกประเภทวัตถุ	183
LAB 9 Object Detection	187
Object Detection	187
Diagram Object Detection	187
9.1 ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับวัตถุ	188
LAB 10 Performance Assessment	191
Performance Assessment	192
Confusion Matrix	192
Accuracy	193
Precision	193
Recall	193

F1-Score	194
LAB 10.1 การคำนวณ Confusion Matrix	194
10.1.1 การหาค่าความถูกต้องในการทำนายผลไม้	194
10.1.2 การประเมินประสิทธิภาพความแม่นยำในการทำนายใบหน้า	196
โจทย์	199
LAB 11 Data Visualization	200
Data Visualization	201
รูปแบบพื้นฐานของการทำ Data Visualization	201
ประโยชน์ของการทำ Data Visualization	204
Vue.js	204
11.1 การเข้าถึงตัวอย่างโปรแกรมสำหรับสร้าง Bar Chart	205
ขั้นตอนการเข้าถึงตัวอย่างโปรแกรมสำหรับสร้าง Bar Chart	205
11.2 การเข้าถึงตัวอย่างโปรแกรมสำหรับสร้าง Pie Chart	209
ขั้นตอนการเข้าถึงตัวอย่างโปรแกรมสำหรับสร้าง Pie Chart	209
11.3 การเข้าถึงตัวอย่างโปรแกรมสำหรับสร้าง Line Chart	213
ขั้นตอนการเข้าถึงตัวอย่างโปรแกรมสำหรับสร้าง Line Chart	213
11.4 ขั้นตอนการใช้ตัวอย่างโปรแกรมแสดงผลข้อมูลในลักษณะแผนภูมิ	216
บรรณานุกรม	219

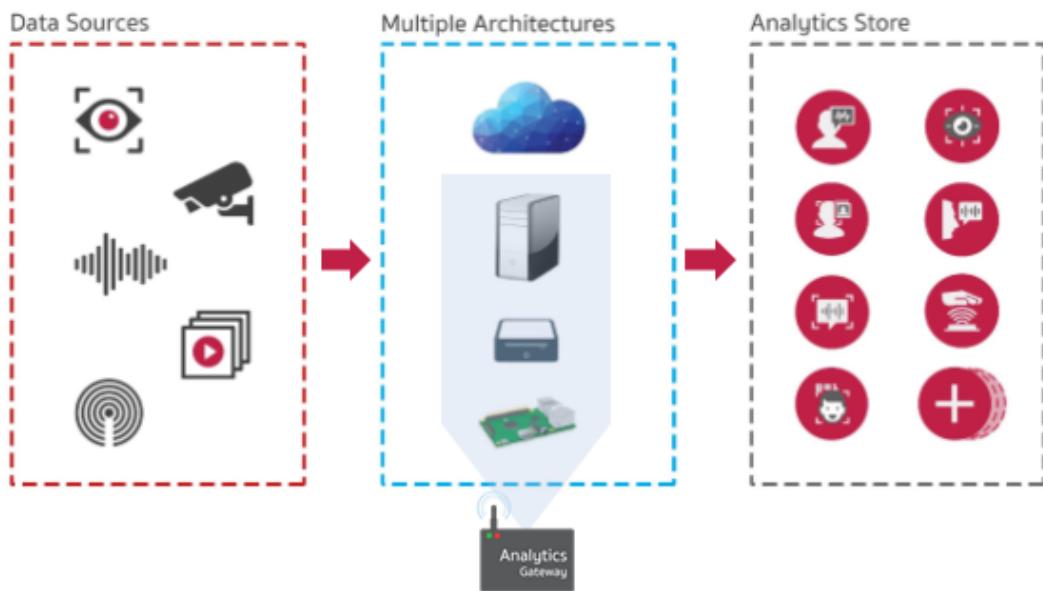
LAB 1 | Introduction to VAM Platform

VAM Platform

Video Analytics Management Platform (VAM Platform) คือ แพลตฟอร์มที่รวมโปรแกรมการประมวลผลและวิเคราะห์ข้อมูล (Analytics) ที่สามารถประยุกต์ใช้ในอาคารและสถานที่ต่าง ๆ เช่น ที่พักอาศัย ออฟฟิศ ร้านค้าปลีก โรงพยาบาล โรงงานอุตสาหกรรม เป็นต้น โปรแกรมจะประมวลผลและวิเคราะห์ข้อมูลจากข้อมูลนำเข้า (Input Source) ประเภทรูปภาพที่แพลตฟอร์มรองรับ ได้แก่ ภาพจากกล้องวงจรปิด (RTSP) วิดีโอไฟล์ ไฟล์ภาพ และข้อมูลภาพจากระบบจัดการวิดีโอ (VMS)

ผู้ใช้ต้องผูกข้อมูลนำเข้าที่แพลตฟอร์มรองรับเข้ากับโปรแกรมการประมวลผลและวิเคราะห์ที่อยู่ในหน้ารวมโปรแกรมการประมวลผลและวิเคราะห์ข้อมูล (Analytics Store) โดยที่หนึ่งโปรแกรมการประมวลผลและวิเคราะห์ข้อมูลสามารถผูกเข้ากับข้อมูลนำเข้าได้มากกว่าหนึ่งข้อมูลนำเข้าและผู้ใช้งานสามารถกำหนดพื้นที่ที่สนใจ (Region Of Interest: ROI) บนข้อมูลนำเข้า เพื่อใช้เป็นตัวช่วยกำหนดขอบเขตในการประมวลผลและวิเคราะห์ข้อมูล ทั้งนี้ผู้ใช้สามารถดูผลลัพธ์การประมวลผลผ่านแพลตฟอร์มได้

โปรแกรมการประมวลผลและวิเคราะห์ข้อมูลภายในแพลตฟอร์มที่นำมาใช้ร่วมกับชุดทดลองสำหรับเรียนรู้ทางด้านการประมวลผลและวิเคราะห์สัญญาณ ภาพและวิดีโอด้วยเทคโนโลยีปัญญาประดิษฐ์จะเกี่ยวข้องกับการรักษาความปลอดภัย เช่น การตรวจจับผู้กรุก การตรวจจับและนับบุคคลในบริเวณที่กำหนด เป็นต้น นอกจากแพลตฟอร์มจะมีโปรแกรมการประมวลผลและวิเคราะห์ข้อมูลที่สามารถนำไปติดตั้งในสถานที่ต่าง ๆ และรองรับข้อมูลนำเข้าที่หลากหลาย แพลตฟอร์มยังเปิดโอกาสให้ผู้ใช้หรือนักพัฒนา (Developer) เข้ามาพัฒนาโปรแกรมการประมวลผลและวิเคราะห์ข้อมูลของตนเองบนแพลตฟอร์มได้อีกด้วย



รูปที่ 1.1 แผนภาพลำดับการรับข้อมูลเข้าและประมวลผลของ VAM Platform

LAB 1.1 การติดตั้งและประกอบชุดทดลองของ VAM Platform

วัตถุประสงค์

1. เพื่อก่อตัวถึงอุปกรณ์ที่ใช้ในการติดตั้งและประกอบชุดการทดลอง
2. เพื่อเรียนรู้วิธีการติดตั้งและประกอบชุดการทดลอง VAM Platform

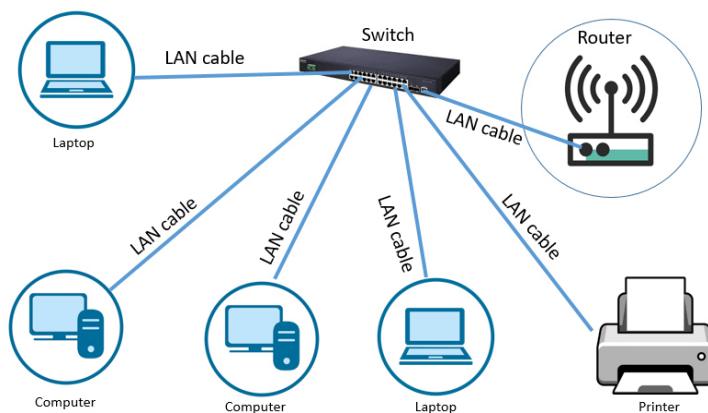
Local area network (LAN)

Local Area Network (LAN) คือ ระบบการเชื่อมโยงเครือข่ายบริเวณเฉพาะที่ เป็นการเชื่อมโยงเครือข่ายคอมพิวเตอร์เข้าด้วยกันทั้งหมดโดยอาศัยสื่อกลาง มีการแบ่งแยกเครือข่ายออกเป็น 2 รูปแบบการเชื่อมโยงคือ การเชื่อมโยงภายในพื้นที่ระยะใกล้หรือ แลน (LAN) และการเชื่อมโยงระยะไกลหรือแวน (WAN)

การเชื่อมโยงภายในพื้นที่ระยะใกล้ (LAN) ส่วนมากจะถูกนำมาใช้มต่อเพื่อใช้งานภายในอาคารหรือบริเวณสถานที่ใกล้เคียงที่สามารถสายถึงกันได้โดยตรง ซึ่งจะใช้สายเคเบิลหรือที่เรียกว่า “สายแลน” เป็นตัวกลางในการเชื่อมต่อ

การเชื่อมโยงเครือข่ายแบบแลน มี 3 รูปแบบ คือ

1. Bus มีการรับส่งข้อมูลด้วยความเร็ว 10-100 MB/s จะเชื่อมต่อกันบนสายสัญญาณเส้นเดียวกัน โดยจะมีอุปกรณ์ที่เรียกว่า T-Connector เป็นตัวแปลงสัญญาณข้อมูลเพื่อนำเข้าสู่ระบบคอมพิวเตอร์และ Terminator ในการปิดหัวท้ายของสายในระบบเครือข่ายเพื่อคุดชับข้อมูลไม่ให้เกิดการสะท้อนกลับของสัญญาณ
2. Star เป็นระบบที่มีเป็นการต่อแบบรวมศูนย์ โดยเครื่องคอมพิวเตอร์ทุกเครื่องจะต่อสายเข้าไปที่อุปกรณ์ที่เรียกว่า Hub หรือ Switch โดยอุปกรณ์ที่เรียกว่า Hub หรือ Switch จะทำหน้าที่เปรียบศูนย์กลางที่ทำหน้าที่กระจายข้อมูล โดยข้อดีของการต่อในรูปแบบนี้คือ หากสายสัญญาณเกิดชำรุดในคอมพิวเตอร์เครื่องใดเครื่องหนึ่ง เครื่องคอมพิวเตอร์อื่น ๆ จะสามารถใช้งานได้ตามปกติ แต่หากศูนย์กลางเกิดเสียหายจะทำให้ทั้งระบบไม่สามารถทำงานได้
3. Ring เป็นระบบที่มีการส่งข้อมูลไปในทิศทางเดียวกัน โดยจะมีเครื่อง Server หรือ Switch ในการปล่อย Token เพื่อตรวจสอบว่ามีเครื่องคอมพิวเตอร์ใดต้องการส่งข้อมูลหรือไม่และระหว่างการส่งข้อมูล เครื่องคอมพิวเตอร์อื่น ๆ ที่ต้องการส่งข้อมูลจะต้องทำการรอให้ข้อมูลก่อนหน้านั้นถูกส่งให้สำเร็จเสียก่อน



Local Area Network

รูปที่ 1.2 แผนภาพ Local Area Network แบบ Star

(ที่มา: itrelease.com/2021/04/what-is-local-area-network-lan-in-computer/)

IP Camera

IP Camera (Internet Protocol Camera) หรือ กล้องเครือข่าย (Network Camera) เป็นประเภทของกล้องวิดีโอดิจิตอล (Digital Video Camera) ที่ใช้สำหรับการเฝ้าระวัง สามารถรับและส่งข้อมูลผ่าน IP Address หรือระบบเครือข่ายอินเทอร์เน็ต ทำให้ผู้ใช้งานดูภาพแบบ Realtime ของกล้องตัวนั้นได้ทั้งในระยะใกล้และระยะไกลผ่านทางโปรแกรมที่มาพร้อมกับกล้องหรือผ่านทางเว็บเบราว์เซอร์ด้วยการใช้ IP Address



รูปที่ 1.3 กล้องประเภท IP Camera

1.1.1 วิธีที่ 1 การติดตั้งและประกอบชุดทดลองเข้ากับอุปกรณ์เสริม

วิธีนี้ผู้เรียนสามารถใช้งานแพลตฟอร์มผ่านอุปกรณ์เสริมที่ต่อเข้ากับเครื่องประมวลผลได้เพียงเครื่องเดียว

อุปกรณ์ที่เกี่ยวข้อง

1. อุปกรณ์หลัก

1.1. Intel NUC 11 enthusiast mini pc kit	1	เครื่อง
1.2. LITEON AC Adapter	1	ตัว
1.2.1. Input AC 100-240V ~ 50-60Hz 3.5A		
1.2.2. Output 19.5V 11.8A 230.0W		
1.3. POE Switch: D-Link DGS-1210-10P 8-Port	1	เครื่อง
1.4. LEADER ELETRONICS Power supply	1	ตัว
1.4.1. Input 100-240V ~ 50/60Hz 1.2A		
1.4.2. Output 54.0V 1.67A 90.0W		
1.5. Hikvision H.265+ Exir Fixed Cube Network Camera	1	ตัว
1.6. สาย LAN	2	เส้น

2. อุปกรณ์เสริม

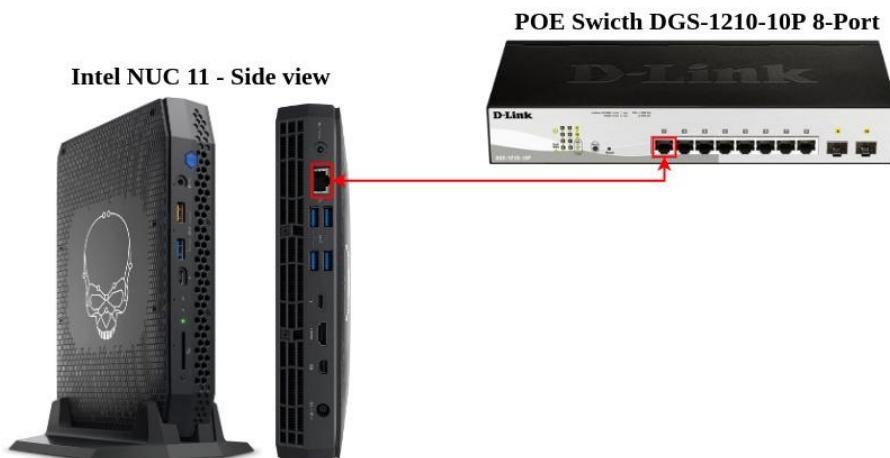
2.1. Azus ZenScreen	1	จอ
2.2. สาย USB Display type-C to Type-C	1	เส้น
2.3. คีย์บอร์ด	1	ตัว

2.4. เม้าส์

1 ตัว

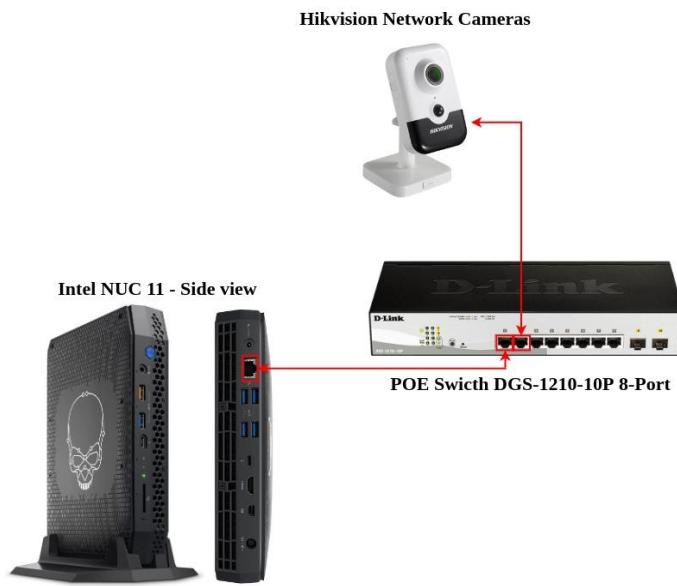
ขั้นตอนการติดตั้งและประกอบชุดการทดลอง

- เชื่อมต่อเครื่อง NUC และ POE Switch ด้วยสาย LAN



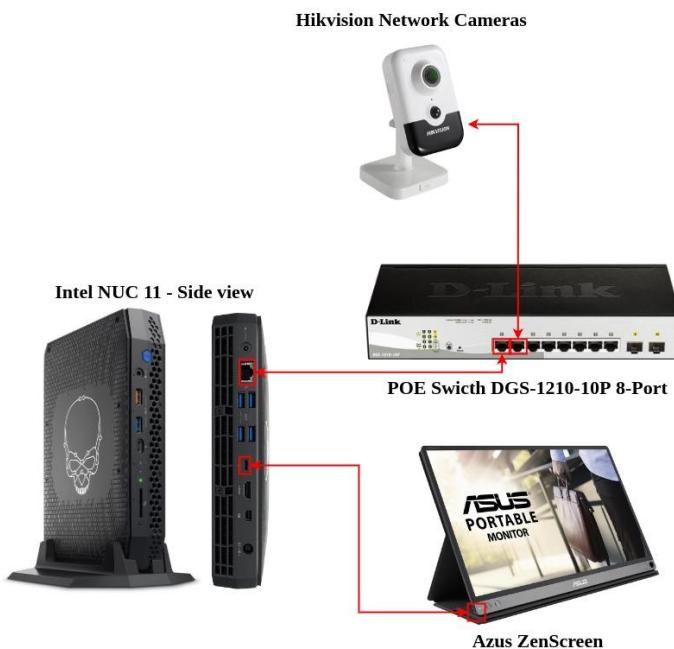
รูปที่ 1.4 การเชื่อมต่อเครื่อง NUC และ POE Switch

2. เชื่อมต่อกล้องและ POE Switch ด้วยสาย LAN



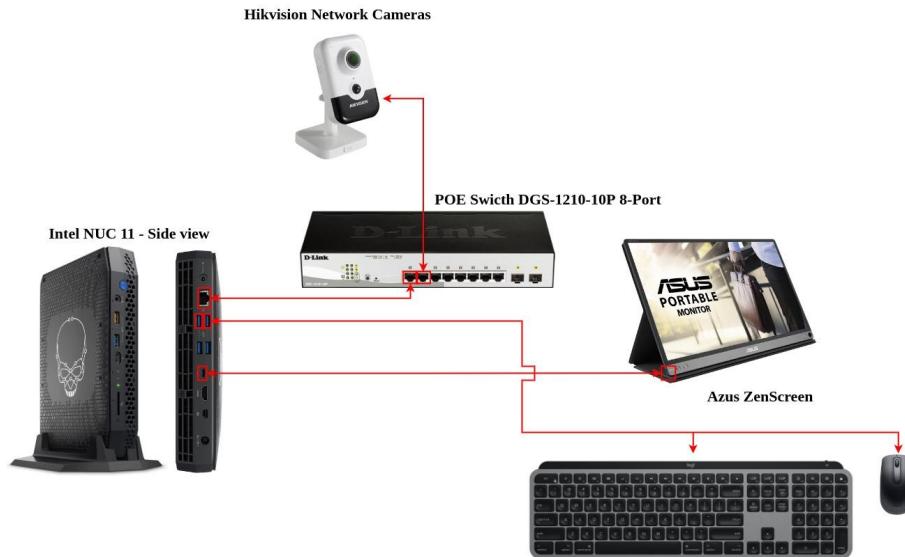
รูปที่ 1.5 การเชื่อมต่อกล้องและ POE Switch

3. เชื่อมต่อเครื่อง NUC และจอ ZenScreen ด้วยสาย USB Display type-C to Type-C



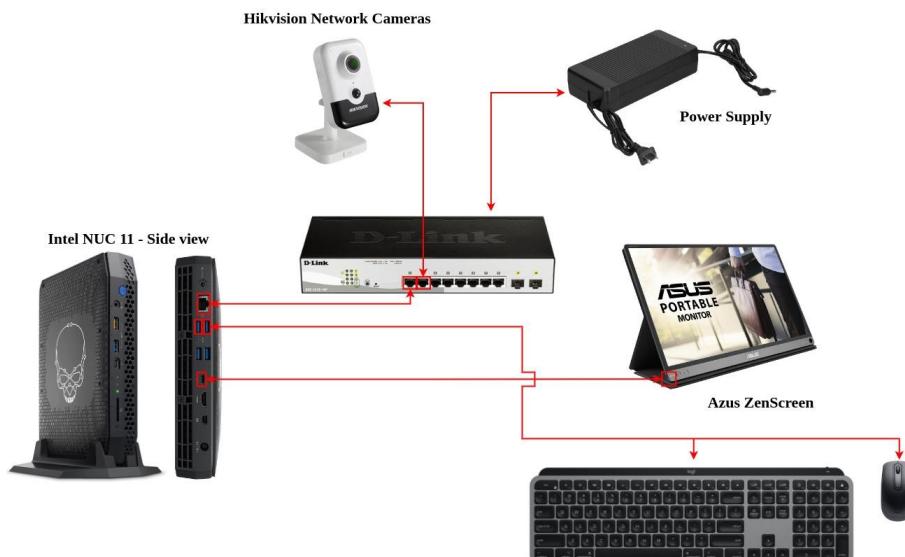
รูปที่ 1.6 การเชื่อมต่อเครื่อง NUC และจอ ZenScreen

4. เชื่อมต่อเครื่อง NUC คีย์บอร์ดและเมาส์



รูปที่ 1.7 การเชื่อมต่อเครื่อง NUC คีย์บอร์ดและเม้าส์

5. เชื่อมต่อแหล่งจ่ายไฟจาก LEADER ELETRONICS Power supply เข้า POE Switch



รูปที่ 1.8 การเชื่อมต่อแหล่งจ่ายไฟจาก Power supply เข้า POE Switch

6. เชื่อมต่อแหล่งจ่ายไฟจาก LITEON AC Adapter เข้าเครื่อง NUC



รูปที่ 1.9 การเชื่อมต่อแหล่งจ่ายไฟจาก AC Adapter เข้าเครื่อง NUC

7. ตรวจสอบการเชื่อมต่ออุปกรณ์ให้เรียบร้อย และวิ่งกดปุ่มเปิดเครื่อง NCU และปุ่มเปิดจอ ZenScreen



รูปที่ 1.10 ปุ่มเปิดเครื่อง NUC และปุ่มเปิดจอ ZenScreen

8. เมื่อเปิดเครื่อง NUC ให้ผู้เรียนทำการกรอกรหัส “eslab” เพื่อเข้าสู่ระบบของเครื่อง NUC

1.1.2 วิธีที่ 2 การติดตั้งและประกอบชุดทดลองเข้ากับเครื่องของผู้เรียน

วิธีนี้จะมีขั้นตอนการติดตั้งคล้ายกับวิธีที่ 1 แต่ต่างกันที่เครื่องประมวลผลไม่ต้องต่อเข้ากับอุปกรณ์เสริม แต่ต่อเข้ากับคอมพิวเตอร์ของผู้เรียนโดยตรง ทำให้ผู้เรียนสามารถใช้งานแพลตฟอร์มบนเครื่องของผู้เรียนได้ทั้งหมดเครื่องพร้อมกัน

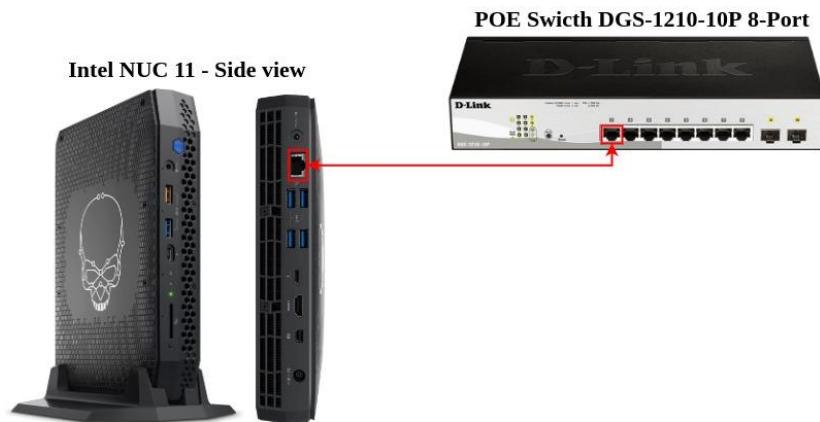
อุปกรณ์ที่เกี่ยวข้อง

1. อุปกรณ์หลัก

1.1. Intel NUC 11 enthusiast mini pc kit	1	เครื่อง
1.2. LITEON AC Adapter	1	ตัว
1.2.1. Input AC 100-240V ~ 50-60Hz 3.5A		
1.2.2. Output 19.5V 11.8A 230.0W		
1.3. POE Switch: D-Link DGS-1210-10P 8-Port	1	เครื่อง
1.4. LEADER ELETRONICS Power supply	1	ตัว
1.4.1. Input 100-240V ~ 50/60Hz 1.2A		
1.4.2. Output 54.0V 1.67A 90.0W		
1.5. Hikvision H.265+ Exir Fixed Cube Network Camera	1	ตัว
1.6. สาย LAN		
1.6.1. สำหรับต่อเข้าอุปกรณ์หลัก	2	เส้น
1.6.2. สำหรับต่อเข้าคอมพิวเตอร์ของผู้เรียน	1	เส้น/เครื่อง
2. เครื่องคอมพิวเตอร์ของผู้เรียน		
2.1. ระบบปฏิบัติการ Windows 10 ขึ้นไป		
2.2. ระบบปฏิบัติการ Ubuntu 18.04 หรือ 20.04		
2.3. ระบบปฏิบัติการ MacOS		

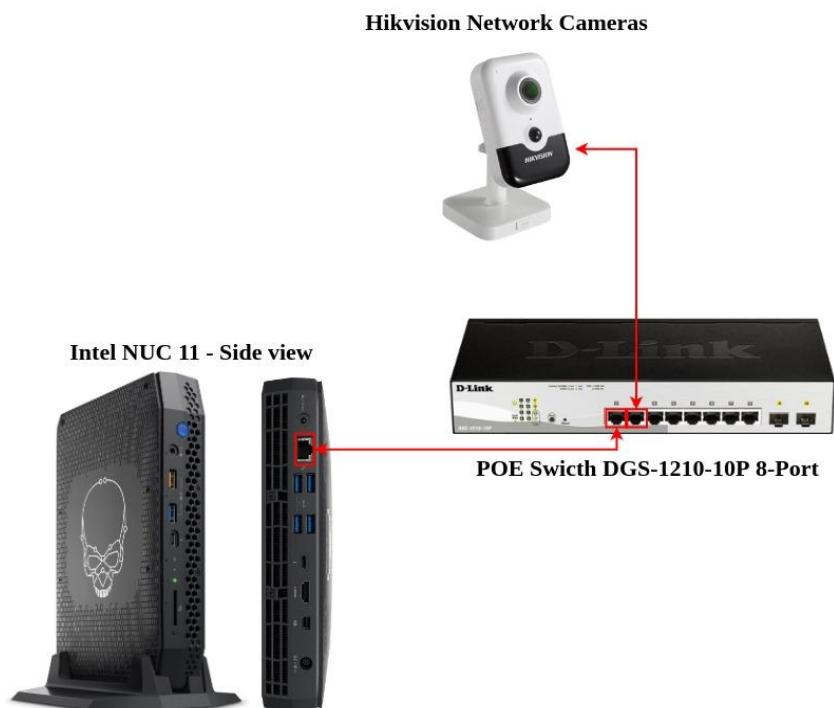
ขั้นตอนการติดตั้งและประกอบชุดการทดลอง

- เชื่อมต่อเครื่อง NUC และ POE Switch ด้วยสาย LAN



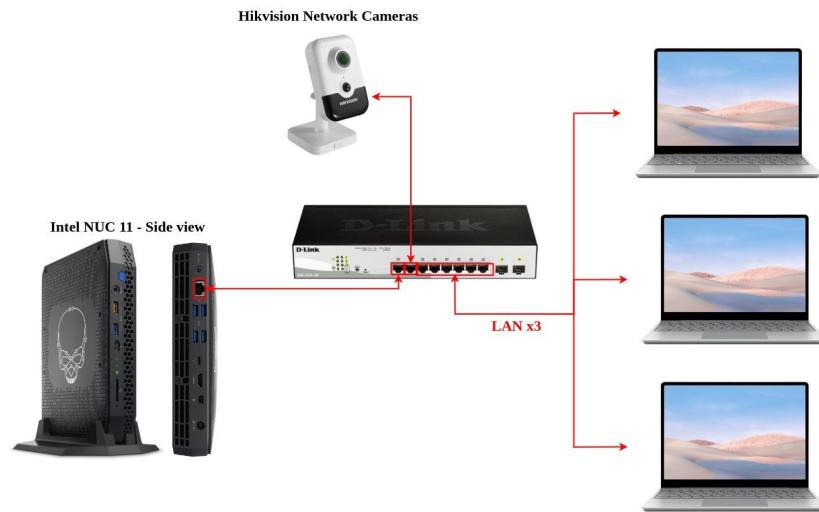
รูปที่ 1.11 การเชื่อมต่อเครื่อง NUC และ POE Switch

- เชื่อมต่อกล้องและ POE Switch ด้วยสาย LAN



รูปที่ 1.12 การเชื่อมต่อกล้องและ POE Switch

3. เชื่อมต่อคอมพิวเตอร์ของผู้เรียนและ POE Switch ด้วยสาย LAN



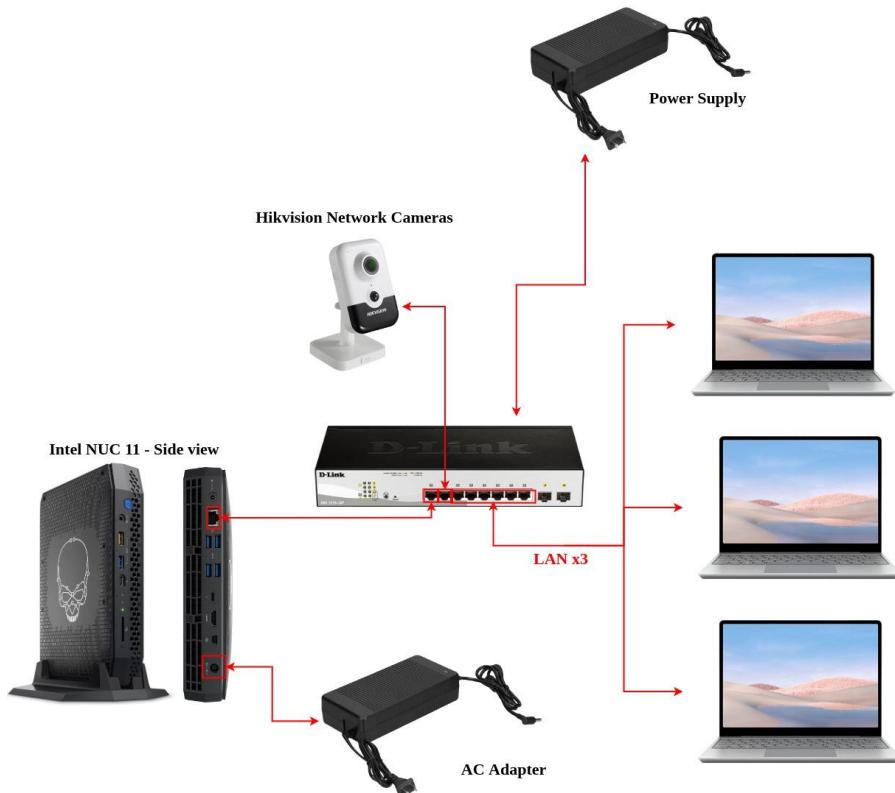
รูปที่ 1.13 การเชื่อมต่อคอมพิวเตอร์ของผู้เรียนและ POE Switch

4. เชื่อมต่อแหล่งจ่ายไฟจาก LEADER ELETRONICS Power supply เข้า POE Switch



รูปที่ 1.14 การเชื่อมต่อแหล่งจ่ายไฟจาก Power supply เข้า POE Switch

5. เชื่อมต่อแหล่งจ่ายไฟจาก LITEON AC Adapter เข้าเครื่อง NUC



รูปที่ 1.15 การเชื่อมต่อแหล่งจ่ายไฟจาก AC Adapter เข้าเครื่อง NUC

6. ตรวจสอบการเชื่อมต่ออุปกรณ์ที่เรียบร้อย และวิ่งกดปุ่มเปิดเครื่อง NUC



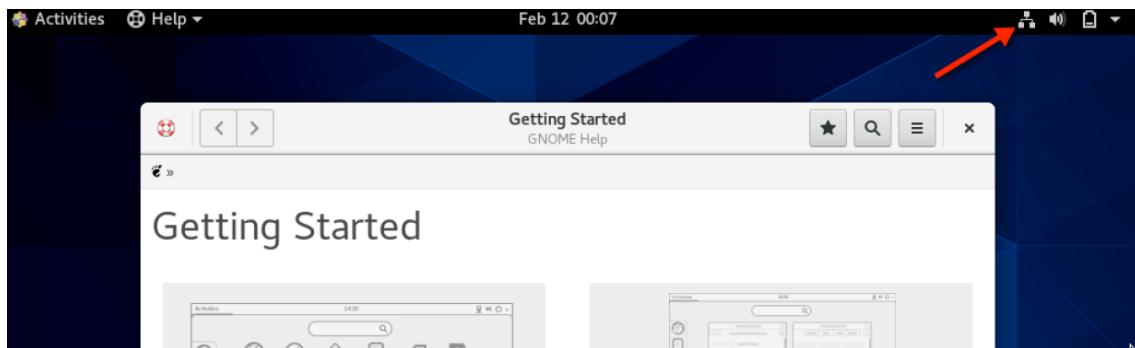
รูปที่ 1.16 ปุ่มเปิดเครื่อง NUC

7. ตั้งค่าเครือข่ายอินเทอร์เน็ตภายในเครื่องคอมพิวเตอร์ของผู้เรียน

การตั้งค่าเครือข่ายอินเทอร์เน็ตภายในเครื่องคอมพิวเตอร์ของผู้เรียน

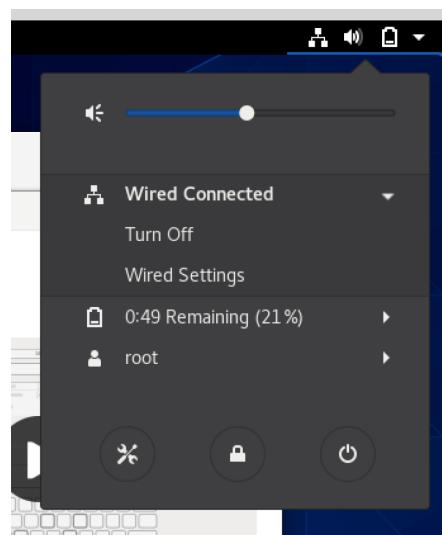
7.1. ระบบปฏิบัติการ Linux

- ตรวจสอบการเชื่อมต่อสายแลนระหว่างเครื่องของผู้เรียนกับ POE Switch เดียวกันกับเครื่องประมวลผล จากนั้นคลิกไอคอน “Network” บริเวณมุมบนด้านขวา



รูปที่ 1.17 การตั้งค่าเครือข่ายอินเทอร์เน็ตของระบบปฏิบัติการ Linux (1)

- คลิกที่ “Wire Connected” แล้วเลือก “Wired Settings”



รูปที่ 1.18 การตั้งค่าเครือข่ายอินเทอร์เน็ตของระบบปฏิบัติการ Linux (2)

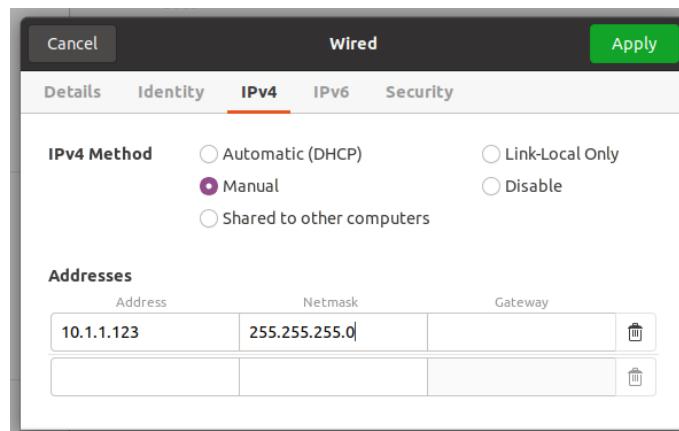
3. คลิกปุ่มรูปฟันเฟืองในตัวเลือก “Wired”



รูปที่ 1.19 การตั้งค่าเครือข่ายอินเทอร์เน็ตของระบบปฏิบัติการ Linux (3)

4. คลิกเลือก “IPv4”

- เปลี่ยน checkbox จาก “Automatic (DHCP)” เป็น “Manual”
- กำหนด “Address” ของเครื่องผู้เรียนเป็นวงเดือน 10.1.1.xxx
- กำหนด “Netmask” เป็น 255.255.255.0
- คลิก “Apply”

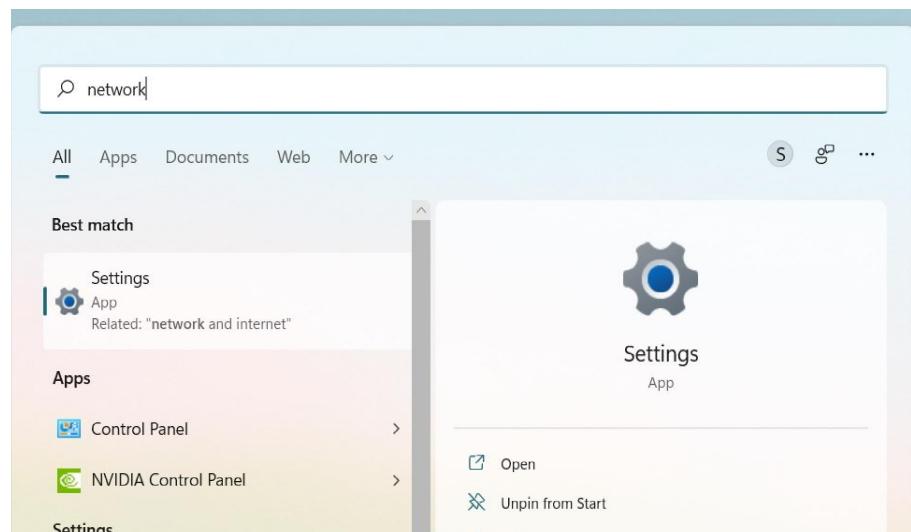


รูปที่ 1.20 การตั้งค่าเครือข่ายอินเทอร์เน็ตของระบบปฏิบัติการ Linux (4)

หมายเหตุ 10.1.1.xxx หมายถึง ให้ผู้เรียนใส่ชุดหมายเลขเลขเดียว ๆ โดยไม่ซ้ำกันและห้ามใช้ชุดหมายเลข 10.1.1.1, 10.1.1.212, 10.1.1.165, 10.1.1.166, และ 10.1.1.224 เนื่องจากชุดหมายเลขดังกล่าวถูกนำไปใช้ในการเชื่อมต่อเครือข่ายมวลผลและกล้อง

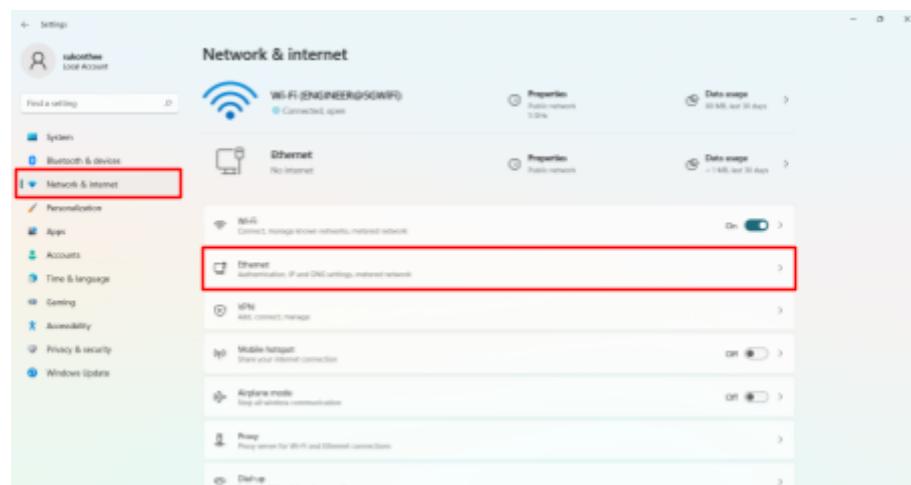
7.2. ระบบปฏิบัติการ Windows

- ตรวจสอบการเชื่อมต่อสายแลนระหว่างเครื่องของผู้เรียนกับ POE Switch เดียวกันกับเครื่องประมวลผล จากนั้นพิมพ์คำว่า “network” ในช่องค้นหาของระบบปฏิบัติการ windows และเลือก “Settings”



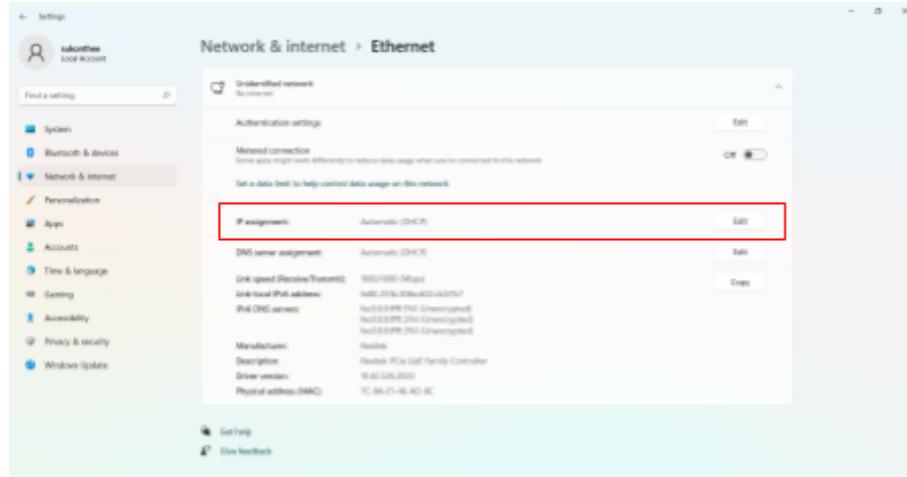
รูปที่ 1.21 การตั้งค่าเครือข่ายอินเทอร์เน็ตของระบบปฏิบัติการ Windows (1)

- คลิก “Network & Internet” บริเวณแถบด้านซ้าย และเลือก “Ethernet”



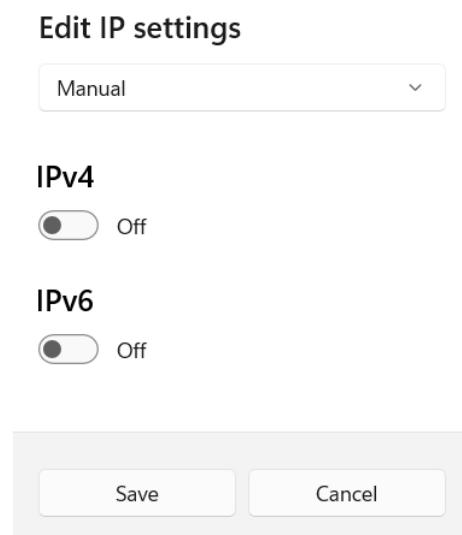
รูปที่ 1.22 การตั้งค่าเครือข่ายอินเทอร์เน็ตของระบบปฏิบัติการ Windows (2)

- กดปุ่ม “Edit” ในตัวเลือก “IP assignment”



รูปที่ 1.23 การตั้งค่าเครือข่ายอินเทอร์เน็ตของระบบปฏิบัติการ Windows (3)

4. เปลี่ยนตัวเลือกจาก “DHCP” เป็น “Manual”



รูปที่ 1.24 การตั้งค่าเครือข่ายอินเทอร์เน็ตของระบบปฏิบัติการ Windows (4)

5. ปรับ IPv4 ให้เป็น “On”

- กำหนด “IP Address” ของเครื่องผู้เรียนเป็นวงแลน 10.1.1.xxx
- กำหนด “Subnet mask” เป็น 255.255.255.0
- คลิก “save”

Edit IP settings

Manual

IPv4

On

IP address
10.1.1.124

Subnet mask
255.255.255.0

Gateway

Preferred DNS

Preferred DNS encryption
Unencrypted only

Alternate DNS

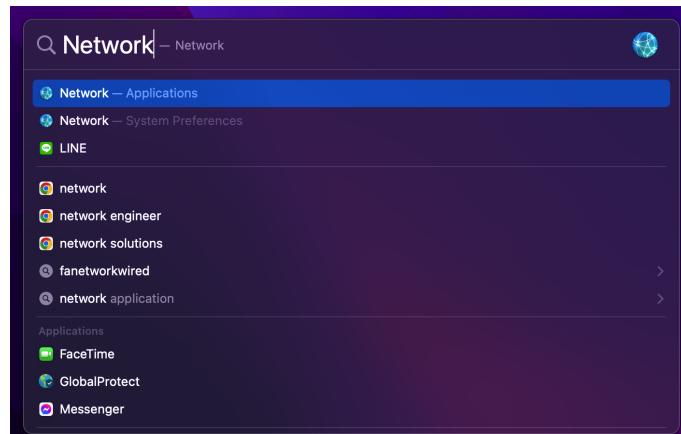
รูปที่ 1.25 การตั้งค่าเครือข่ายอินเทอร์เน็ตของระบบปฏิบัติการ Windows (5)

หมายเหตุ

10.1.1.xxx หมายถึง ให้ผู้เรียนใส่ชุดหมายเลขใด ๆ โดยไม่ซ้ำกันและห้ามใช้ชุดหมายเลข 10.1.1.1, 10.1.1.212, 10.1.1.165, 10.1.1.166, และ 10.1.1.224 เนื่องจากชุดหมายเลขดังกล่าวถูกนำไปใช้ในการเชื่อมต่อเครือข่ายมวลผลและกล้อง

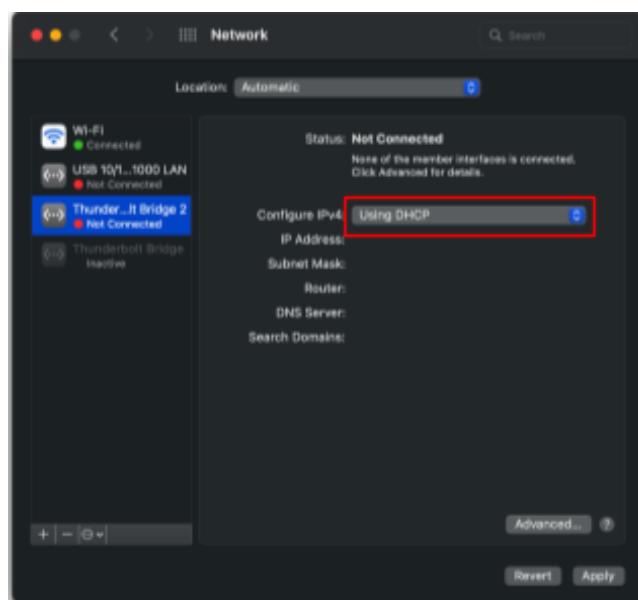
7.3. ระบบปฏิบัติการ Mac

- ตรวจสอบการเชื่อมต่อสายแลนระหว่างเครื่องของผู้เรียนกับ POE Switch เดียวกันกับเครื่องประเมินผล จากนั้นค้นหาคำว่า “Network” ด้วย Spotlight บน Mac และคลิกเลือก Network.



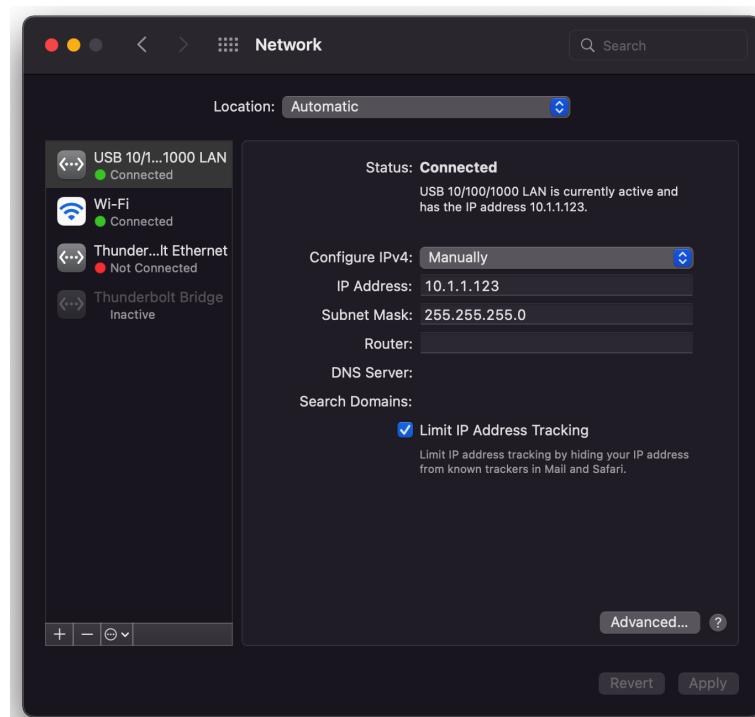
รูปที่ 1.26 การตั้งค่าเครือข่ายอินเทอร์เน็ตของระบบปฏิบัติการ Mac (1)

- เลือกการเชื่อมต่อผ่านสาย LAN และเปลี่ยนตัวเลือก Configure IPv4 ให้เป็น “Manually”



รูปที่ 1.27 การตั้งค่าเครือข่ายอินเทอร์เน็ตของระบบปฏิบัติการ Mac (2)

- กำหนด “IP Address” ของเครื่องให้เป็นวงเดน 10.1.1.xxx และ “Subnet Mask” เป็น 255.255.255.0 จากนั้นคลิก “Apply”



รูปที่ 1.28 การตั้งค่าเครือข่ายอินเทอร์เน็ตของระบบปฏิบัติการ Mac (3)

หมายเหตุ

10.1.1.xxx หมายถึง ให้ผู้เรียนใส่ชุดหมายเลขใด ๆ โดยไม่ซ้ำกันและห้ามใช้ชุดหมายเลข 10.1.1.1, 10.1.1.212, 10.1.1.165, 10.1.1.166, และ 10.1.1.224 เนื่องจากชุดหมายเลขตั้งกล่าวถูกนำไปใช้ในการเชื่อมต่อเครื่องประมวลผลและกล้อง

ในกรณีที่หน้าการตั้งค่าของ Mac ไม่เหมือนกัน สามารถตรวจสอบวิธีตั้งค่าของ Mac แต่ละเครื่องซึ่งได้จาก <https://support.apple.com/guide/mac-help/use-dhcp-or-a-manual-ip-address-on-mac-mchl/p2718/mac>

LAB 1.2 การเข้าใช้งาน VAM Platform

วัตถุประสงค์

1. เพื่อเรียนรู้วิธีการเข้าสู่ระบบ VAM Platform
2. เพื่อแนะนำตำแหน่งเมนูของ VAM Platform

อุปกรณ์ที่เกี่ยวข้อง

1. ชุดการทดลอง VAM Platform

สิทธิ์การใช้งานแพลตฟอร์ม

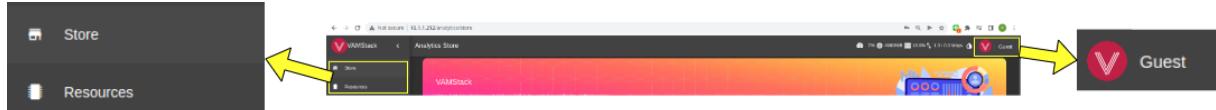
สิทธิ์การใช้งานแพลตฟอร์มจะมีความแตกต่างกันตามสถานะของผู้ใช้จะถูกแบ่งออกเป็น 2 สถานะ คือ สถานะผู้เข้าชม (Guest) และสถานะผู้ดูแลแพลตฟอร์ม (Admin) เริ่มแรกที่ผู้ใช้เข้าสู่หน้าแพลตฟอร์มโดยไม่ได้ทำการเข้าสู่ระบบ (Log in) จะมีสถานะเป็นผู้เข้าชม และสามารถเข้าสู่ระบบในสถานะผู้ดูแลแพลตฟอร์มด้วยการใช้บัญชีที่ทางทีมผู้พัฒนาจัดเตรียมไว้ให้ ซึ่งสถานะของผู้ใช้งานจะมีสิทธิ์การใช้งานแพลตฟอร์มแตกต่างกันดังตารางที่ 1.1

ตารางที่ 1.1 สิทธิ์การใช้งานแพลตฟอร์มของแต่ละสถานะ

สิทธิ์การใช้งานแพลตฟอร์ม	สถานะ	
	Guest	Admin
การเข้าถึงหน้ารวมโปรแกรมการประมวลผลและวิเคราะห์ข้อมูล (Analytics Store)	✓	✓
การจัดการข้อมูลนำเข้า (Sources Management)	✗	✓
การจัดการโปรแกรมประมวลผลและวิเคราะห์ข้อมูล (Assignment)	✗	✓
การเข้าถึงหน้าตรวจสอบจำนวนทรัพยากรการใช้งานของเครื่องที่ติดตั้งแพลตฟอร์ม (Resources Monitoring)	✓	✓

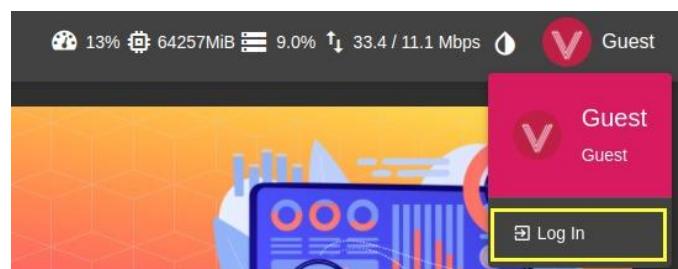
1.2.1 ขั้นตอนการเข้าสู่ระบบของ VAM Platform

1. เข้าสู่หน้า VAM Platform โดยพิมพ์ “[10.1.1.212](#)” บนเว็บเบราว์เซอร์



รูปที่ 1.29 สถานะและແຄບເມນຸເຮີມຕົ້ນຂອງ VAM Platform

2. ຄືກທີ່ສານະກາຣໃຊ້ຈານບຣິໄວນມູນບນດ້ານຂວາແລະເລືອກ “Log In”



รูปที่ 1.30 ເມນຸເຂົ້າສູ່ຮະບບຂອງ VAM Platform

3. ເຂົ້າສູ່ຮະບບໂດຍກາຣໃຊ້ບັນຫຼືທີ່ທາງທີ່ມີຜູ້ພໍມນາຈັດເຕຣີຍມໃຫ້ແລະເລືອກ “Log in”

3.1 username: team01 password: study01

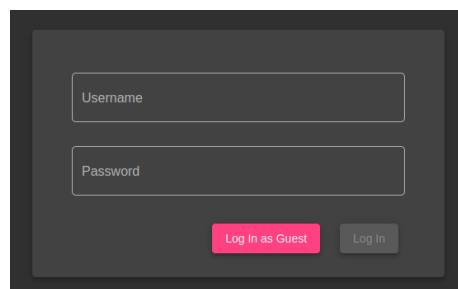
3.2 username: team02 password: study02

3.3 username: team03 password: study03

3.4 username: team04 password: study04

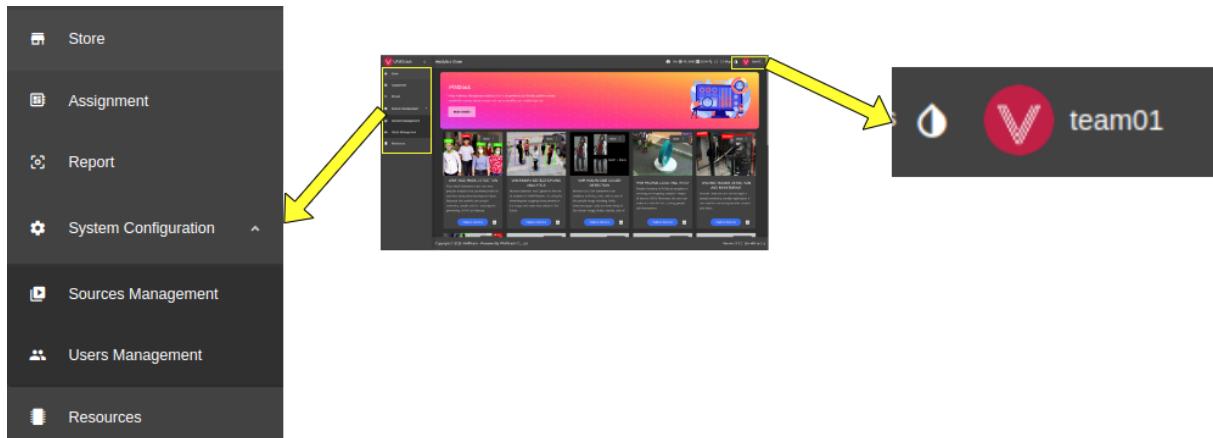
3.5 username: team05 password: study05

หมายເຫດ
ຜູ້ຮັບສານສາມາດເຂົ້າສູ່ຮະບບແພລດົກໂຮມ ໂດຍໃຫ້ 1 ບັນຫຼືຕ່ອ 1 ກລຸ່ມ
ໜ້າມໃຊ້ບັນຫຼືໜ້າກັນ



รูปที่ 1.31 หน้าต่างการเข้าสู่ระบบของแพลตฟอร์ม

4. หลังจากทำการเข้าสู่ระบบ สถานะการใช้งานแพลตฟอร์มจะเปลี่ยนเป็นผู้ดูแลแพลตฟอร์มที่ชื่อว่า “team0x” ตาม username ที่ใช้เข้าสู่ระบบ



รูปที่ 1.32 สถานะและแบบเมนูหลังเข้าสู่ระบบของ VAM Platform

1.2.2 Analytics Store

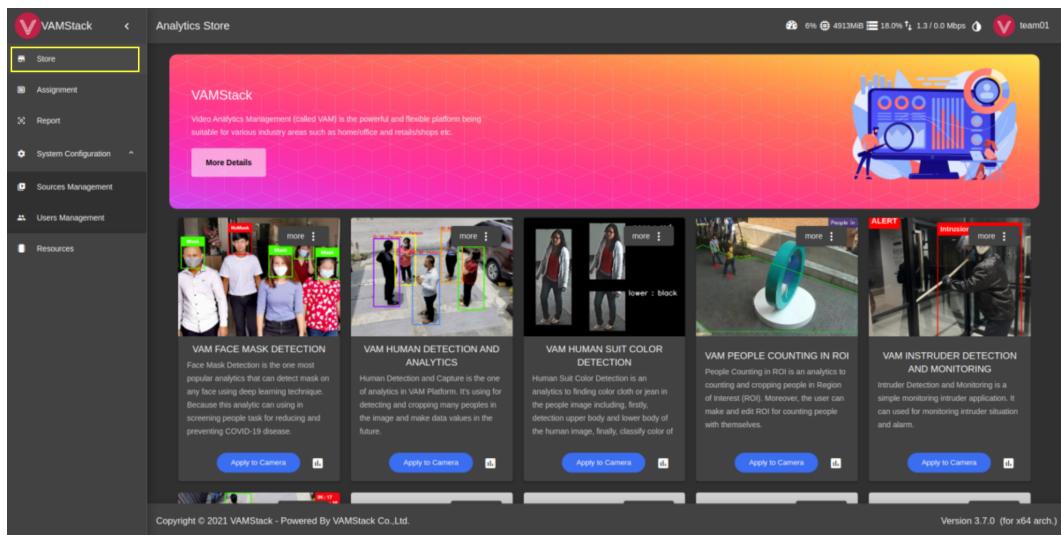
Analytics Store คือ หน้ารวมโปรแกรมการประมวลผลและวิเคราะห์ข้อมูลทั้งหมดในแพลตฟอร์มและเป็นหน้าสำหรับการผูกข้อมูลนำเข้าที่ลงทะเบียนในแพลตฟอร์ม ซึ่งบนแพลตฟอร์มจะมีโปรแกรมการประมวลผลและวิเคราะห์ข้อมูลดังตารางที่ 1.2

ตารางที่ 1.2 โปรแกรมการประมวลผลและวิเคราะห์ข้อมูลทั้งหมดในแพลตฟอร์ม

ชื่อ	คำอธิบาย
Face mask Detection	โปรแกรมสำหรับวิเคราะห์การสวมใส่หน้ากากอนามัยบนใบหน้า
Intrusion Detection	โปรแกรมสำหรับตรวจสอบจับและแจ้งเตือนผู้บุกรุก
Human Detection	โปรแกรมสำหรับตรวจสอบบุคคล
Human Suit Color Detection	โปรแกรมสำหรับแยกแยะสีของชุดบุคคล
People Counting	โปรแกรมสำหรับนับบุคคล
People Counting in-out	โปรแกรมสำหรับนับบุคคลเข้า-ออก

studio card	โปรแกรมสำหรับใช้เชื่อมต่อการทำงานกับ VAM Studio
-------------	---

การเข้าสู่หน้า Analytics Store สามารถทำได้โดยการคลิก “Store” ที่แถบเมนูด้านซ้าย นอกจากนี้ หลังจากที่ผู้ใช้เข้าสู่ระบบ จะพบหน้า Analytics Store โดยอัตโนมัติ

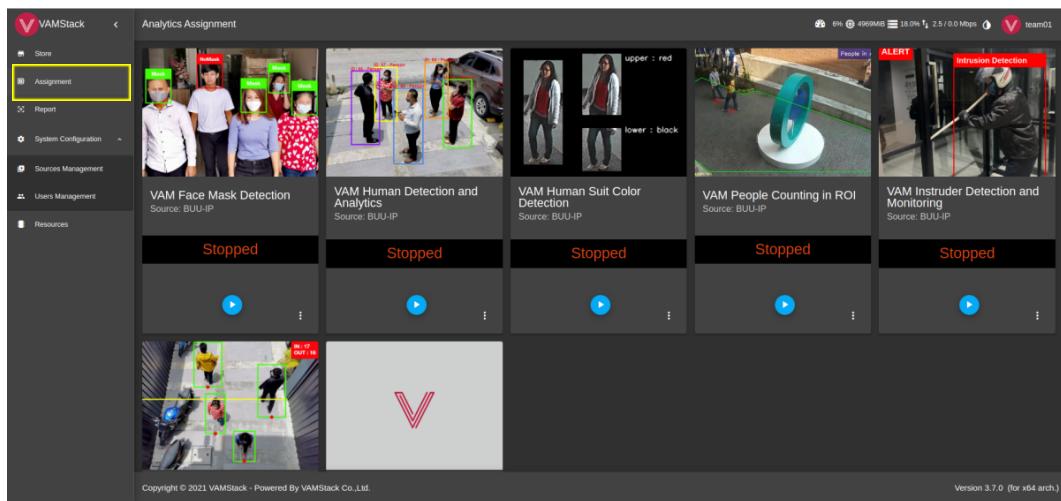


รูปที่ 1.33 หน้าเมนู Store

1.2.3 Assignment

Assignment คือ หน้ากำหนดการทำงานและจัดการโปรแกรมการประมวลผลและวิเคราะห์ข้อมูลที่ผูกเข้ากับข้อมูลนำเข้า โดยผู้ใช้สามารถสั่งเปิดหรือปิดการทำงานของตัวประมวลผลและสามารถกำหนดพื้นที่ที่สนใจ (Region Of Interest: ROI) บนข้อมูลนำเข้า

การเข้าสู่หน้า Assignment สามารถทำได้โดยการคลิก “Assignment” ที่แถบเมนูด้านซ้าย โดยทางทีมผู้พัฒนาได้ทำการผูกกล้องเข้ากับโปรแกรมประมวลผลและวิเคราะห์ข้อมูลที่ในแพลตฟอร์มจัดเตรียมไว้แล้ว



รูปที่ 1.34 หน้าเมนู Assignment

1.2.4 System Configuration

System Configuration คือ เมนูการจัดการข้อมูลจัดเก็บภายในระบบแพลตฟอร์ม ประกอบไปด้วย 3 ส่วน คือ การจัดการข้อมูลนำเข้า (Sources Management) การจัดการข้อมูลผู้ใช้ (Users Management) และ การจัดการข้อมูลสังกัดองค์กร (Organization management)

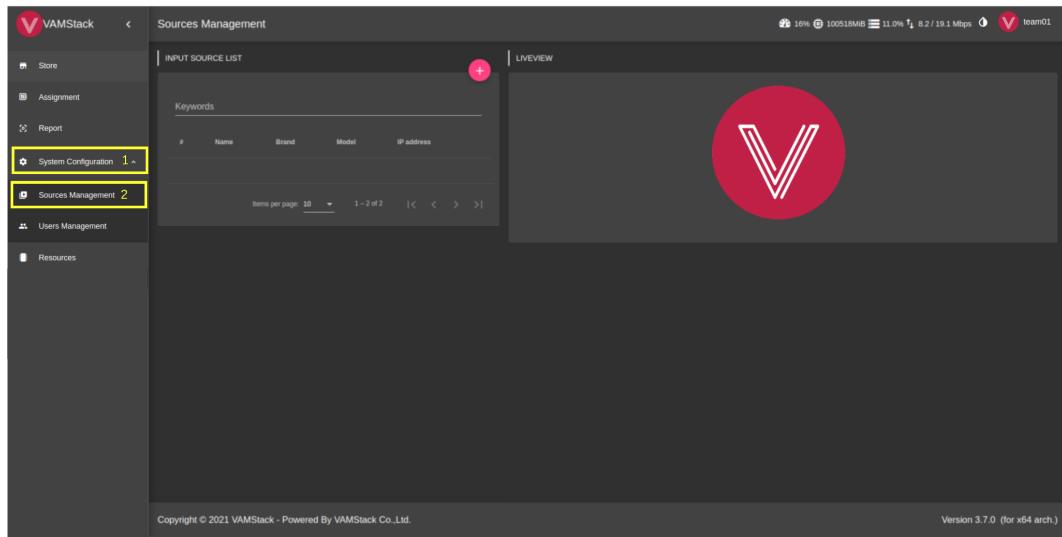
1.2.4.1 Sources Management

Sources Management คือ หน้าการจัดการข้อมูลนำเข้า ใช้สำหรับลงทะเบียน แก้ไขหรือลบข้อมูลนำเข้าภายในระบบแพลตฟอร์ม โดยจะแสดงข้อมูลดังนี้

1. ชื่อกล้อง
2. แบรนด์ของกล้อง
3. ไมเดลของกล้อง
4. IP Address ของกล้อง

ขั้นตอนการเข้าหน้าการจัดการข้อมูลนำเข้า

1. คลิก “System Configuration” ที่แถบเมนูด้านซ้าย
2. เลือก “Sources Management”



รูปที่ 1.35 หน้าเมนู Sources Management

1.2.4.2 Users Management

User Management คือ หน้าการจัดการบัญชีผู้ใช้ จะแสดงข้อมูลบัญชีของผู้ใช้ที่ถูกลงทะเบียนในแพลตฟอร์ม โดยข้อมูลที่แสดงจะประกอบข้อมูลดังนี้

1. ชื่อผู้ใช้สำหรับเข้าสู่ระบบ (Username)
2. ชื่อ-นามสกุล (Firstname - Lastname)
3. อีเมลล์ (E-mail)
4. สังกัดองค์กร (Organization)

ขั้นตอนการเข้าหน้าการจัดการบัญชีผู้ใช้

1. คลิก “System Configuration” ที่แถบเมนูด้านซ้าย
2. เลือก “Users Management”

รูปที่ 1.36 หน้า User Management

1.2.4 Resources

Resources คือ หน้าแสดงภาพรวมของการใช้ทรัพยากรของระบบ จะแสดงข้อมูลเช่น หน่วยประมวลผลกลาง (CPU), หน่วยความจำหลัก (RAM), ความเร็วในการอ่านข้อมูล (Disk Read), ความเร็วในการเขียนข้อมูล (Disk Write), ความเร็วในการรับข้อมูลผ่านเครือข่าย (Download) และความเร็วในการส่งข้อมูลผ่านเครือข่าย (Upload) เป็นต้น

การเข้าสู่หน้า Resources สามารถทำได้โดยการคลิก “Resources” ที่แถบเมนูด้านซ้าย

รูปที่ 1.37 หน้าเมนู Resources

LAB 1.3 การจัดการข้อมูลนำเข้าประเภท RTSP

วัตถุประสงค์

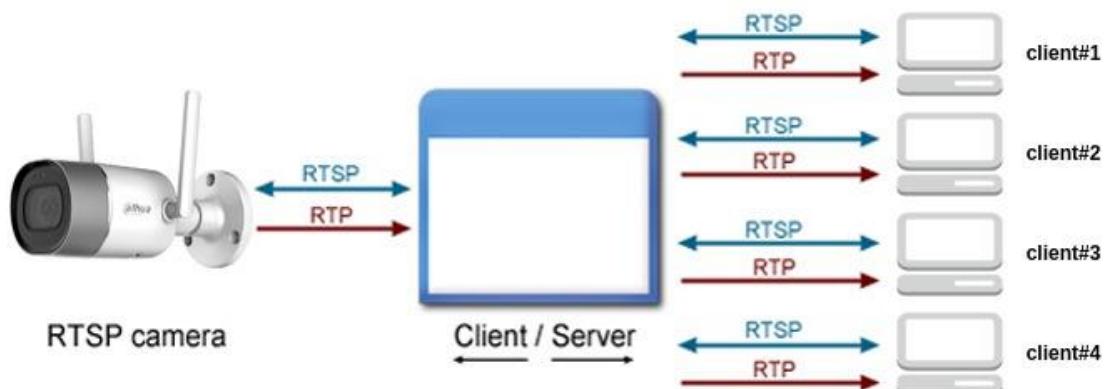
1. เพื่อเรียนรู้วิธีการลงทะเบียนข้อมูลกล้องลง VAM Platform
2. เพื่อเรียนรู้วิธีการจัดการข้อมูลกล้อง

อุปกรณ์ที่เกี่ยวข้อง

1. ชุดการทดลอง VAM Platform

Real Time Streaming Protocol (RTSP)

Real Time Streaming Protocol หรือย่อว่า RTSP เป็นโปรโตคอลที่ใช้รูปแบบ client/server ถูกออกแบบเพื่อใช้ในการแสดงสื่อมัลติมีเดีย ซึ่งโปรโตคอลเหล่านี้จะทำให้ผู้ใช้งานสามารถดึงข้อมูลภาพจากกล้องวงจรปิดมาใช้งานแบบ realtime ได้ เพียงแค่ผู้ใช้รู้ ip address, username, password ของอุปกรณ์กล้องวงจรปิดนั้น



รูปที่ 1.38 แผนภาพการทำงานของ Real Time Streaming Protocol

โดยprotoคือเหล่านี้ถูกสร้างและนำมาควบคุมกระແสข้อมูลระหว่าง client และ server ซึ่งจะทำหน้าที่เป็นรีโมทควบคุมข้อมูลให้มีความเข้มข้นกันตลอดเวลา เช่น การรับส่งข้อมูลเสียงหรือวิดีโอ

หลักการทำงานของ RTSP

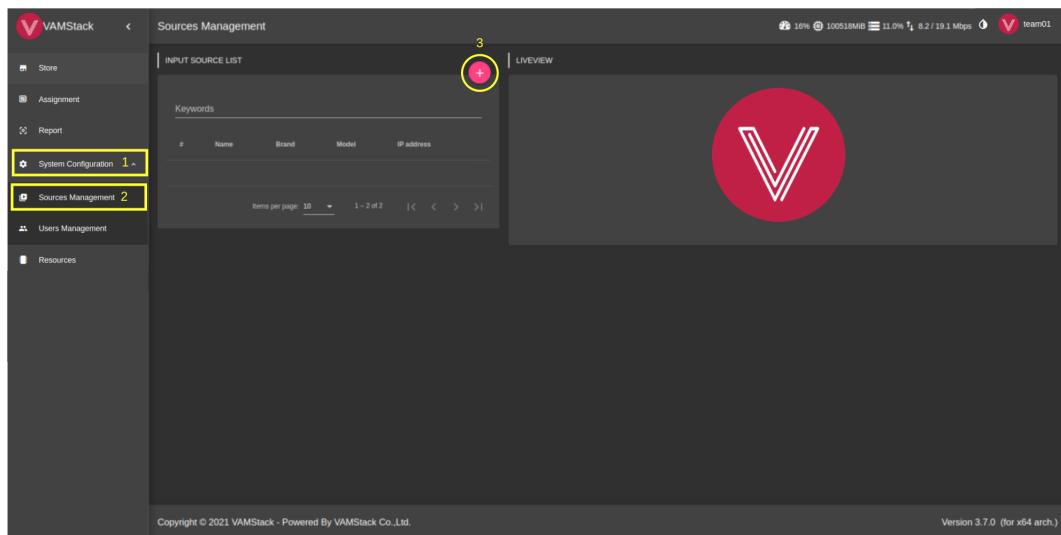
ผู้ใช้งานหรือแอปพลิเคชันที่พยายามส่งคำขอข้อมูลจากการควบคุมระยะไกลจะถูกมองว่าเป็น client ที่ส่งคำขอข้อมูลไปยังตัว server ที่มีชุดข้อมูลอยู่ ผ่านการกำหนดค่าต่าง ๆ ตามที่ client แต่ละตัวต้องการ เช่น การสั่งให้หยุดส่งข้อมูลชั่วคราว, การสั่งให้บันทึกข้อมูลจาก server เข้าสู่ไฟล์ client เป็นต้น ในมุมของการรับส่งข้อมูลสตรีมในรูปแบบของรูปภาพหรือวิดีโอด้วย client จะทำการส่งคำขอไปยัง server และเมื่อ server ตอบรับสำเร็จและส่งข้อมูลกลับมาให้ทาง client ผู้ใช้งานหรือแอปพลิเคชันที่เป็น client ก็จะเริ่มสตรีมข้อมูลที่รับมาจาก server ในรูปแบบของบิตสตรีม

ข้อมูลลงทะเบียนกล้อง Hikvision H.265+ Exir Fixed Cube Network Camera

1. Video Source Protocol : RTSP
2. Video Source Brand : HIK (Hikvision)
3. Video Source Model : CUBE
4. Source Name : BUU-IP
5. IP Address : 10.1.1.166
6. Username : admin
7. Password : P@ssw0rd

1.3.1 ขั้นตอนการลงทะเบียนข้อมูลนำเข้า

1. คลิก “System Configuration” ที่แถบเมนูด้านซ้าย
2. เลือก “Sources Management”
3. คลิกปุ่มบวกสีแดงแล้วทำการลงทะเบียนเพื่อเพิ่มกล้องเข้าสู่ VAM Platform



รูปที่ 1.39 ขั้นตอนการลงทะเบียนข้อมูลนำเข้า (1)

4. กรอกข้อมูลไปร์โตคอลของข้อมูลนำเข้าในรูปแบบของสตรีมภาพจากกล้องวงจรปิด (CCTV IP Camera)

4.1. ข้อมูลที่ต้องกรอกประกอบไปด้วย

- 4.1.1. Video Source Protocol : เลือกชนิดข้อมูลนำเข้าเป็น “RTSP”
- 4.1.2. Video Source Brand : ยี่ห้อของกล้อง (**ปั้งคับกรอก**)
- 4.1.3. Video Source Model : โมเดลของกล้อง

4.2. คลิก “Next” เมื่อกรอกข้อมูลครบตามที่กำหนด

รูปที่ 1.40 ขั้นตอนการลงทะเบียนข้อมูลนำเข้า (2)

5. กรอกข้อมูลของกล้องวงจรปิด (CCTV IP Camera)

5.1. ข้อมูลที่ต้องกรอกประกอบไปด้วย

- 5.1.1. Source Name : ชื่อของกล้องที่ผู้ใช้งานกำหนด (**บังคับกรอก**)
- 5.1.2. IP Address ของกล้อง (**บังคับกรอก**)
- 5.1.3. Username และ Password ของกล้อง (**บังคับกรอก**)
- 5.1.4. Serial Number : หมายเลขประจำตัวผลิตภัณฑ์ของกล้อง
- 5.1.5. MAC Address : ตัวเลขฐาน 16 จำนวน 12 หลักของอุปกรณ์ที่เชื่อมต่อเน็ตเวิร์ก
- 5.1.6. Extend Path : ส่วนต่อขยายใน URL

Add video source

What is video source protocol ? **2 Video Source Infomation** Done

Source Name *	BUU-IP
IP Address *	10.1.1.166
Username *	admin
	Password * P@ssw0rd
Serial Number	MAC Address
Extend Path	
Previous Done ✓	

รูปที่ 1.41 ขั้นตอนการลงทะเบียนข้อมูลนำเข้า (3)

6. ตรวจสอบข้อมูลในชั้นตอนสุดท้ายและคลิก “Add Video Source” เพื่อทำการลงทะเบียนกล้องของวงจรปิด หากมีข้อมูลที่ผิดพลาดสามารถย้อนกลับไปแก้ไขข้อมูลได้จากการคลิก “Previous”

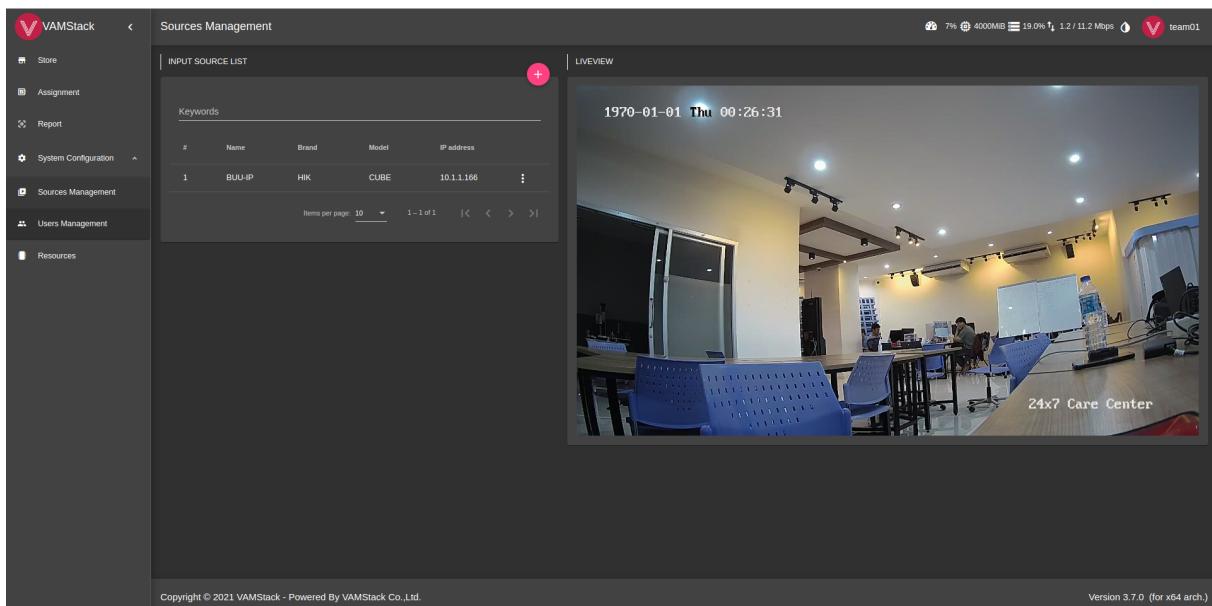
Add video source

What is video source protocol ? **2 Video Source Infomation** Done

Protocol	RTSP
Brand	HIK
Model	CUBE
Source Name	BUU-IP
IP Address	10.1.1.166
Username	admin
	Password P@ssw0rd
Previous Add video source ✓	

รูปที่ 1.42 ขั้นตอนการลงทะเบียนข้อมูลนำเข้า (4)

7. หากลงทะเบียนสำเร็จจะปรากฏรายละเอียดและภาพจากกล้องในหน้าต่างของ Sources Management

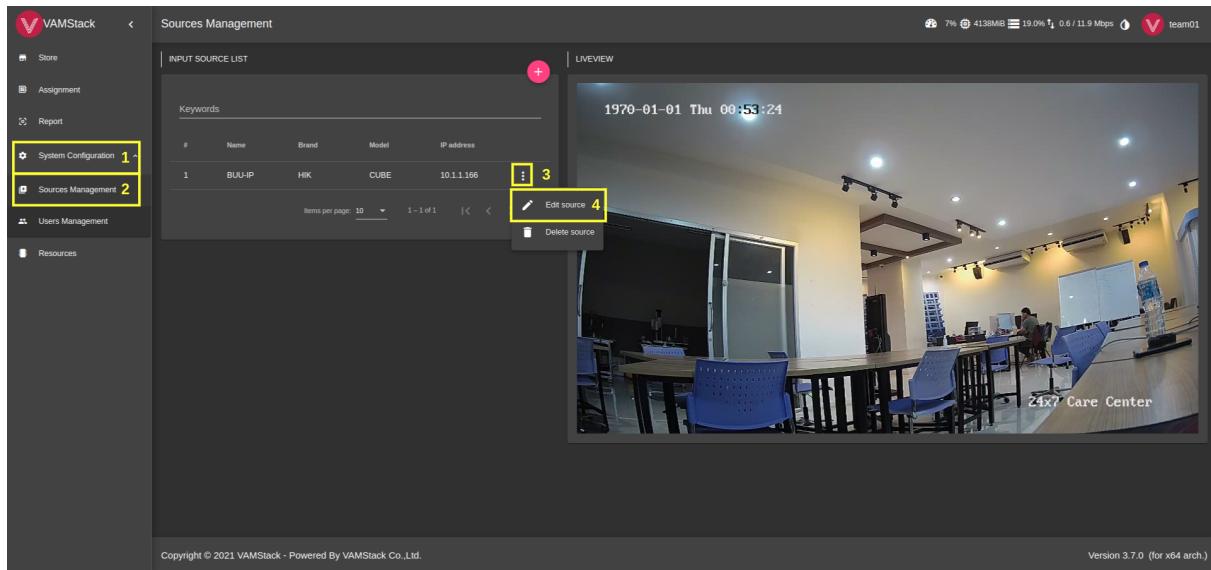


รูปที่ 1.43 ขั้นตอนการลงทะเบียนข้อมูลนำเข้า (5)

หมายเหตุ	<ol style="list-style-type: none"> กล้องต้องเชื่อมต่ออยู่ในเครือข่ายอินเตอร์เน็ตเดียวกันกับเครื่องที่ติดตั้งแพลตฟอร์ม การเรียกใช้ RTSP Protocol ของกล้อง IP Camera บางยี่ห้อจำเป็นต้องใส่ Extend Path ตัวอย่างเช่น กรณีที่เป็นกล้องยี่ห้อ “Axis” ผู้ใช้งานต้องกรอก Extend Path โดยใช้ข้อความว่า “/axis-media/media.amp” และแน่นอนจะไม่สามารถลงทะเบียนได้ หากข้อมูลภาพจากกล้องไม่ปรากฏอยู่ในอยู่ในส่วนของ “Liveview” มีความเป็นไปได้ว่าผู้ใช้กรอกข้อมูลในการลงทะเบียนไม่ถูกต้องหรือกล้องด้านนั้นอาจถูกปิดการใช้งาน
----------	--

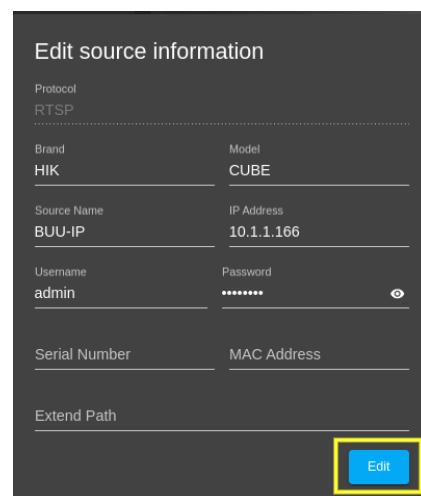
1.3.2 ขั้นตอนการแก้ไขข้อมูลนำเข้า

1. คลิก “System Configuration” ที่แถบเมนูด้านซ้าย
2. เลือก “Sources Management”
3. คลิกที่จุด 3 จุด บนແບບชื่อລັດໆອງຫວີ້ອໍານຸລຳນຳເຂົ້າທີ່ຕ້ອງການແກ້ໄຂ
4. เลือก “Edit Source” เพื่อເຂົ້າສູ່ໜ້າຕ່າງການແກ້ໄຂຂໍ້ມູນລຳນຳເຂົ້າ



รูปที่ 1.44 ขั้นตอนการแก้ไขข้อมูลนำเข้า (1)

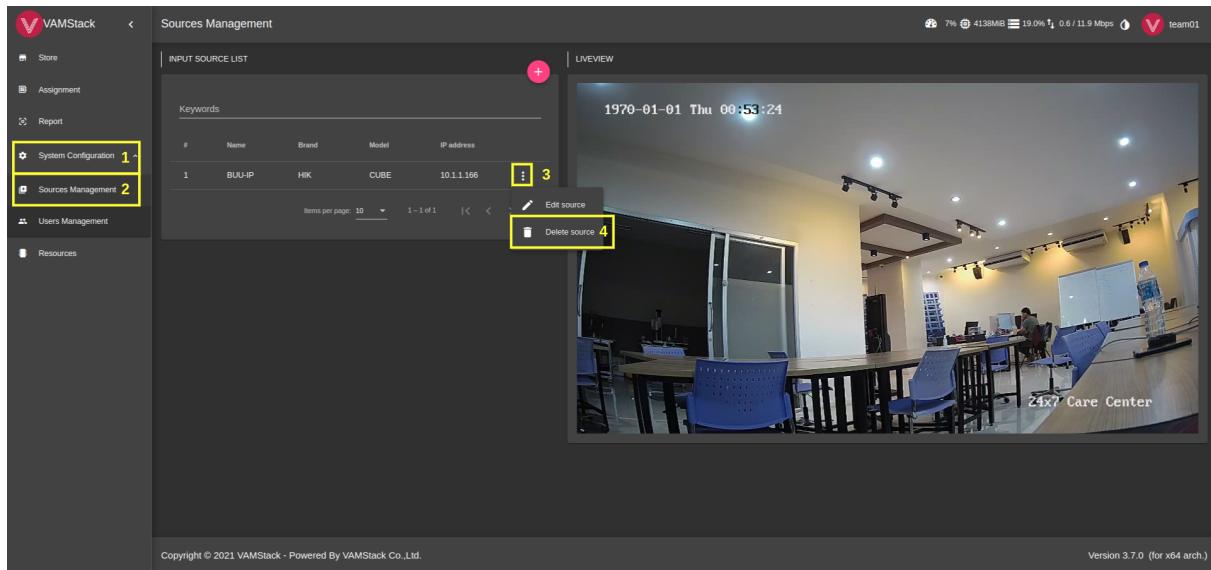
5. แก้ไขข้อมูลและคลิก “Edit” ยืนยันการแก้ไขข้อมูล



รูปที่ 1.45 ขั้นตอนการแก้ไขข้อมูลนำเข้า (2)

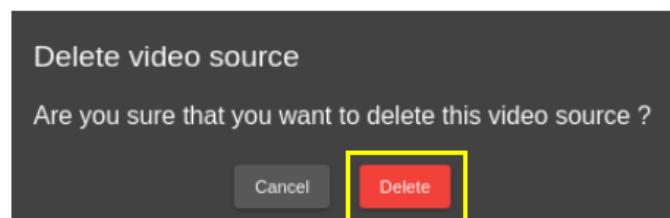
1.3.3 ขั้นตอนการลบข้อมูลนำเข้า

1. คลิก “System Configuration” ที่แถบเมนูด้านซ้าย
2. เลือก “Sources Management”
3. คลิกที่จุด 3 جد บนแถบข้อกอลังหรือข้อมูลนำเข้าที่ต้องการลบ
4. เลือก “Delete Source” เพื่อทำการลบข้อมูล



รูปที่ 1.46 ขั้นตอนการลบข้อมูลนำเข้า (1)

5. คลิก “Delete” ยืนยันการลบข้อมูล



รูปที่ 1.47 ขั้นตอนการลบข้อมูลนำเข้า (2)

LAB 1.4 การจัดการโปรแกรมการประมวลผลและวิเคราะห์ข้อมูล

วัตถุประสงค์

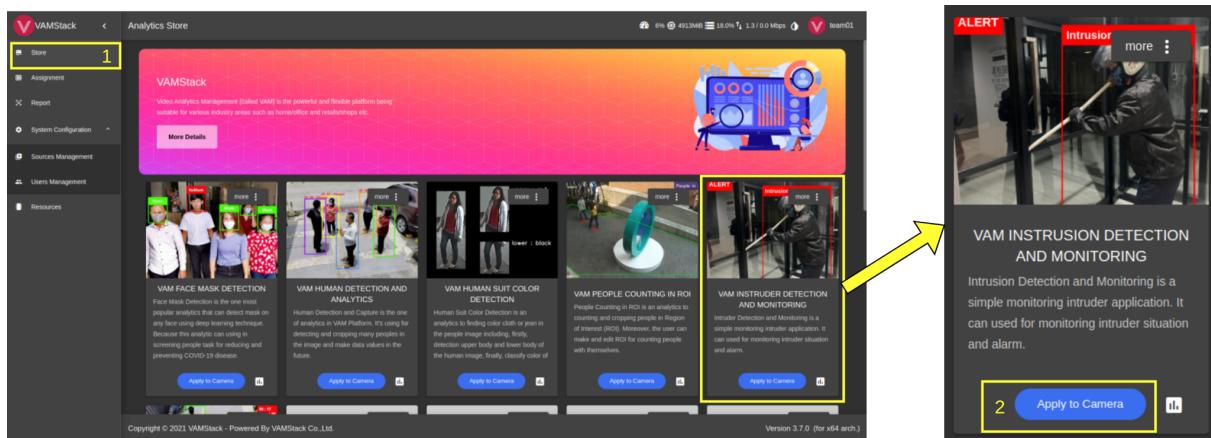
- เพื่อเรียนรู้วิธีการผูกกล้องเข้ากับโปรแกรมการประมวลผลและวิเคราะห์ข้อมูลใน VAM Platform
- เพื่อเรียนรู้วิธีการกำหนดการทำงานของโปรแกรมการประมวลผลและวิเคราะห์ข้อมูลใน VAM Platform
- เพื่อเรียนรู้วิธีการตั้งค่าพื้นที่ที่สนใจ (Region Of Interest: ROI)

อุปกรณ์ที่เกี่ยวข้อง

- ชุดการทดลอง VAM Platform

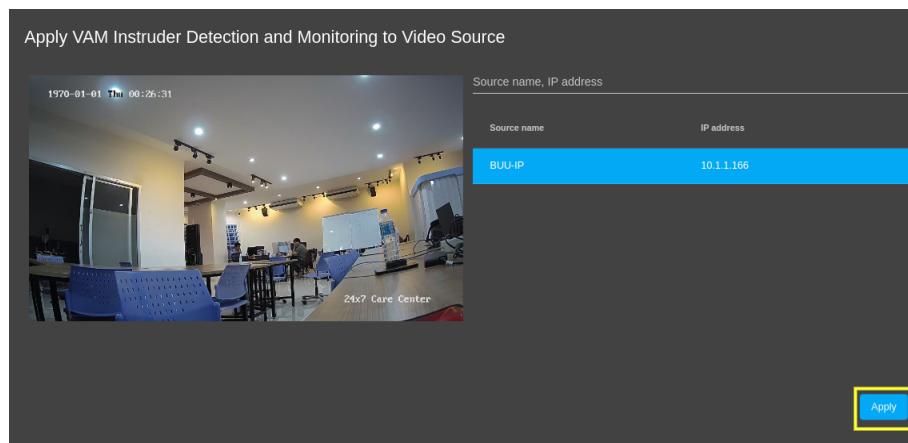
1.4.1 ขั้นตอนการผูกกล้อง

- คลิก “Store” ที่แถบเมนูด้านซ้าย
- คลิกปุ่ม “Apply to Camera” ของโปรแกรมที่ต้องการใช้งาน



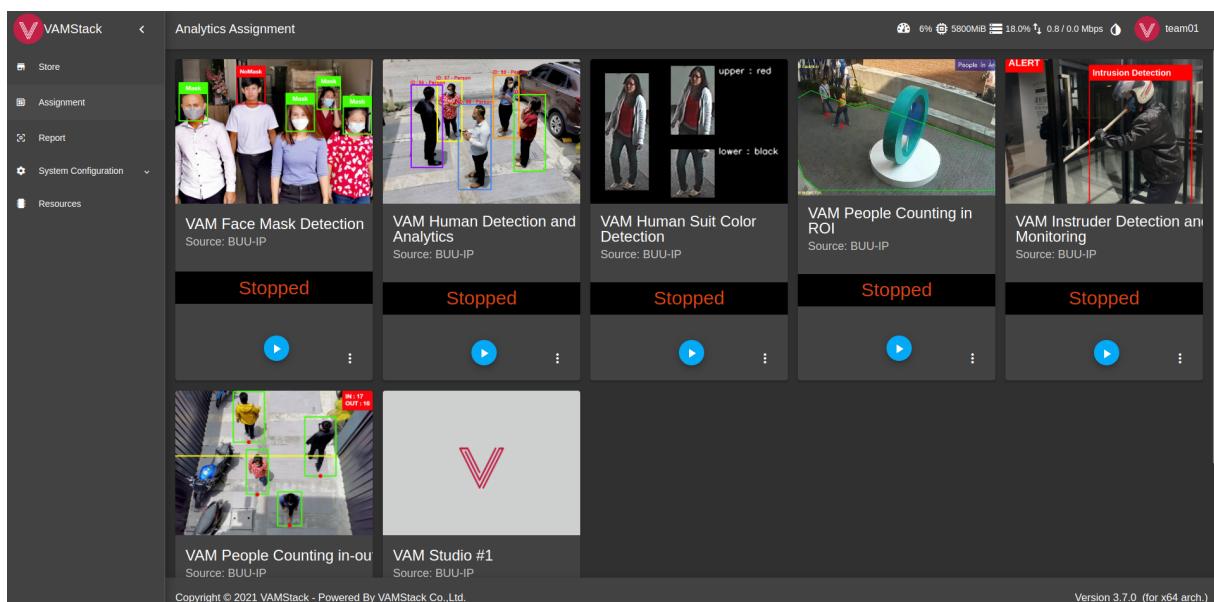
รูปที่ 1.48 ขั้นตอนการผูกข้อมูลนำเข้า (1)

3. เลือกข้อมูลนำเข้าที่เหมาะสมกับโปรแกรมและคลิก “Apply” เพื่อยืนยันการผูกข้อมูลนำเข้า



รูปที่ 1.49 ขั้นตอนการผูกข้อมูลนำเข้า (2)

4. หลังยืนยันการผูกข้อมูลนำเข้า จะกลับสู่หน้า “Store” โดยอัตโนมัติ
 5. คลิก “Assignment” ที่แถบเมนูด้านซ้าย จะพบการตัวประมวลผลที่ทำการผูกข้อมูลนำเข้าที่เลือกไว้อยู่ในสถานะหยุดทำ
งาน (Stopped) หากไม่พบการตัวประมวลผลที่ผูกข้อมูลนำเข้าที่เลือกไว้ให้ทำการผูกข้อมูลนำเข้าใหม่อีกครั้ง



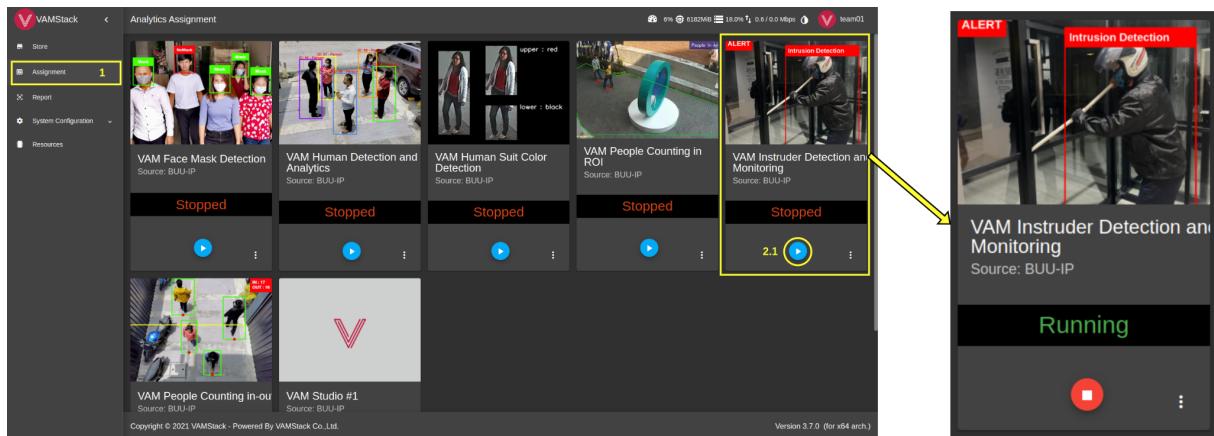
รูปที่ 1.50 ขั้นตอนการผูกข้อมูลนำเข้า (3)

1.4.2 ขั้นตอนการเปลี่ยนสถานะการทำงาน

1. คลิก “Assignment” ที่แถบเมนูด้านซ้าย

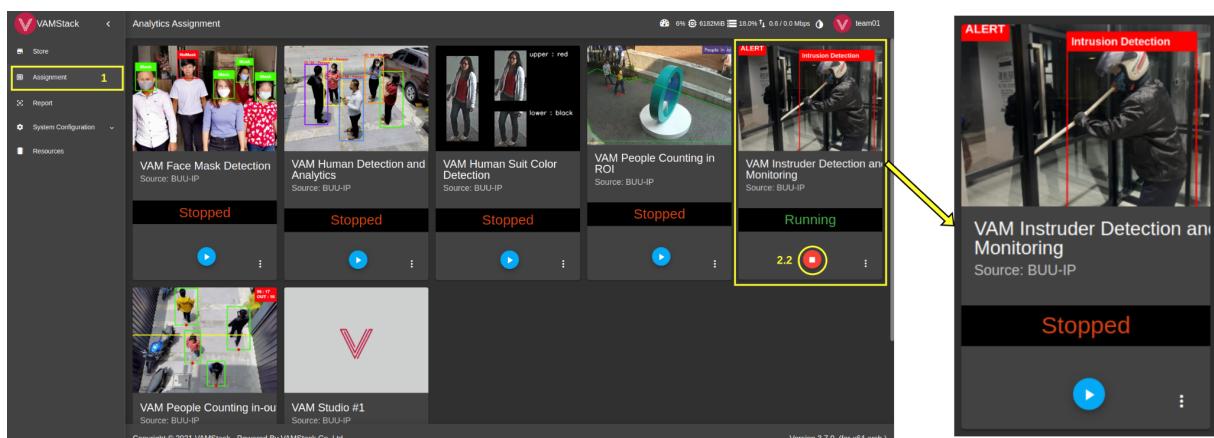
2. คลิกที่ปุ่มการทำงานของโปรแกรม เพื่อเปลี่ยนสถานะการทำงาน

2.1. ปุ่มเริ่มทำงาน



รูปที่ 1.51 การเปิดการทำงานของโปรแกรมการประมวลผล

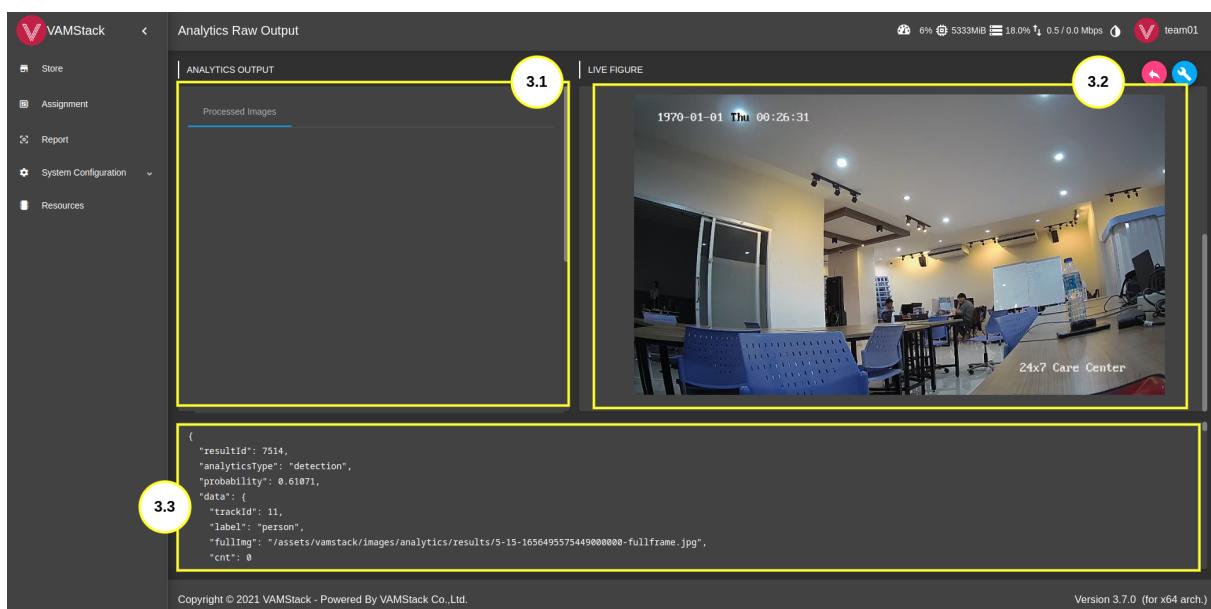
2.2. ปุ่มหยุดการทำงาน



รูปที่ 1.52 การปิดการทำงานของโปรแกรมการประมวลผล

3. เมื่อโปรแกรมเริ่มทำงาน ผู้ใช้สามารถคลิกที่โปรแกรมเพื่อดูผลลัพธ์การทำงาน ซึ่งหน้าผลลัพธ์การทำงานจะแบ่งส่วนการแสดงผลเป็น 3 ส่วนดังนี้

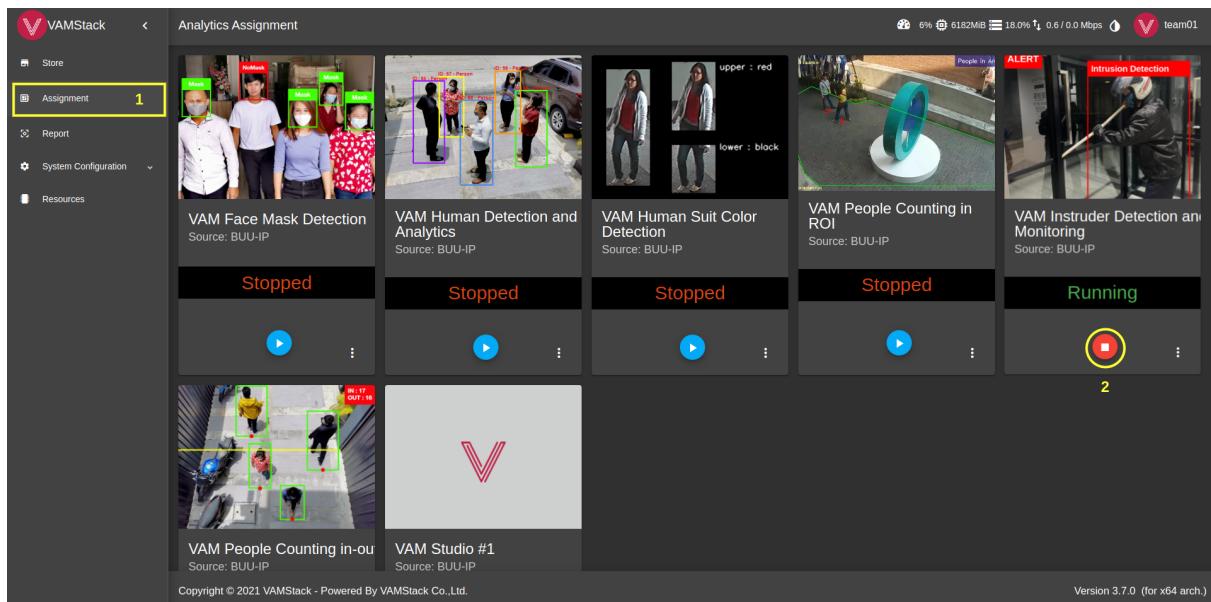
- 3.1. ภาพเฉพาะส่วนที่ถูกตรวจจับและบันทึกไว้
- 3.2. ภาพแสดงผลการทำงานร่วมกันระหว่างข้อมูลนำเข้าที่ผูกอยู่กับตัวประมวลผลนั้น ซึ่งจะแสดงผลแบบ Realtime หากไม่มีภาพแสดงในส่วนนี้แสดงว่าข้อมูลนำเข้าถูกปิดการใช้งาน
- 3.3. ข้อมูลของภาพที่ตรวจจับจะถูกแสดงผลในรูปแบบของข้อมูล Json ซึ่งผู้ใช้สามารถนำข้อมูลนี้ไปใช้งานต่อได้



รูปที่ 1.53 หน้าแสดงผลการทำงานของโปรแกรมประมวลผล

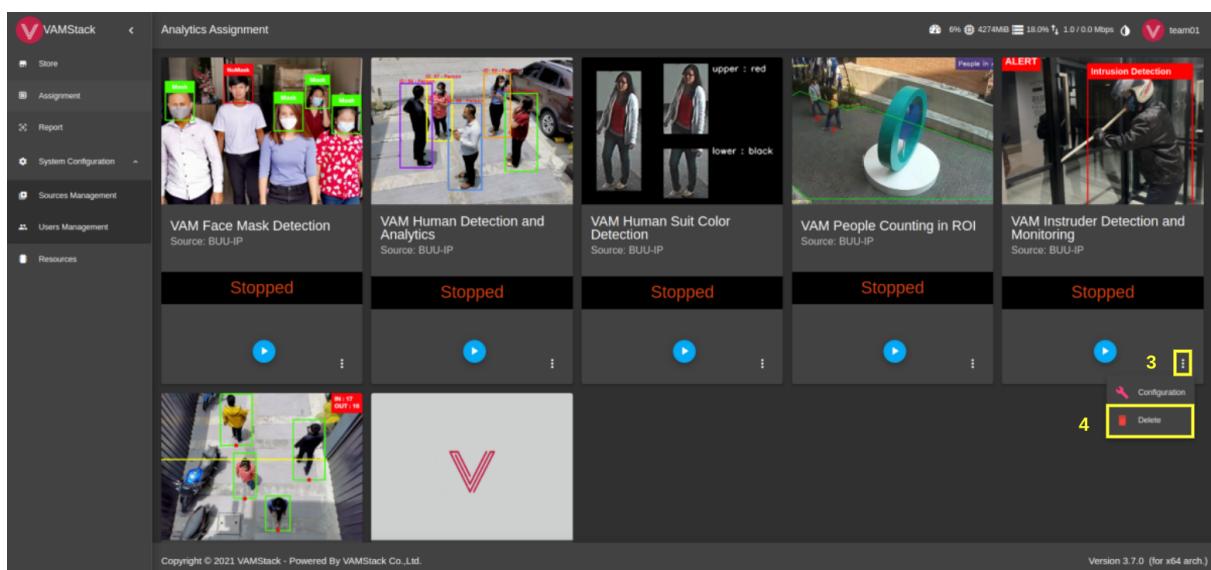
1.4.3 ขั้นตอนการลบโปรแกรมการประมวลผล

1. คลิก “Assignment” ที่แถบเมนูด้านซ้าย
2. ปิดการทำงานของโปรแกรมการประมวลผลทุกครั้งที่ต้องการลบ



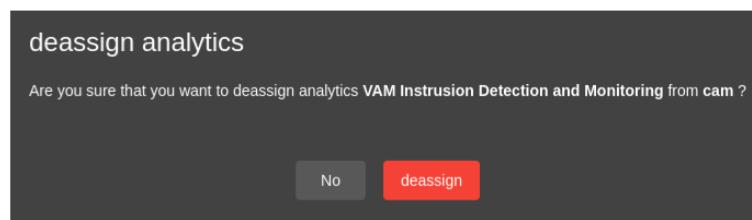
รูปที่ 1.54 ขั้นตอนการลบโปรแกรมการประมวลผล (1)

3. คลิกที่จุด 3 จุด บนโปรแกรมที่ต้องการลบ
4. เลือก “Delete” เพื่อทำการลบ



รูปที่ 1.55 ขั้นตอนการลบโปรแกรมการประมวลผล (2)

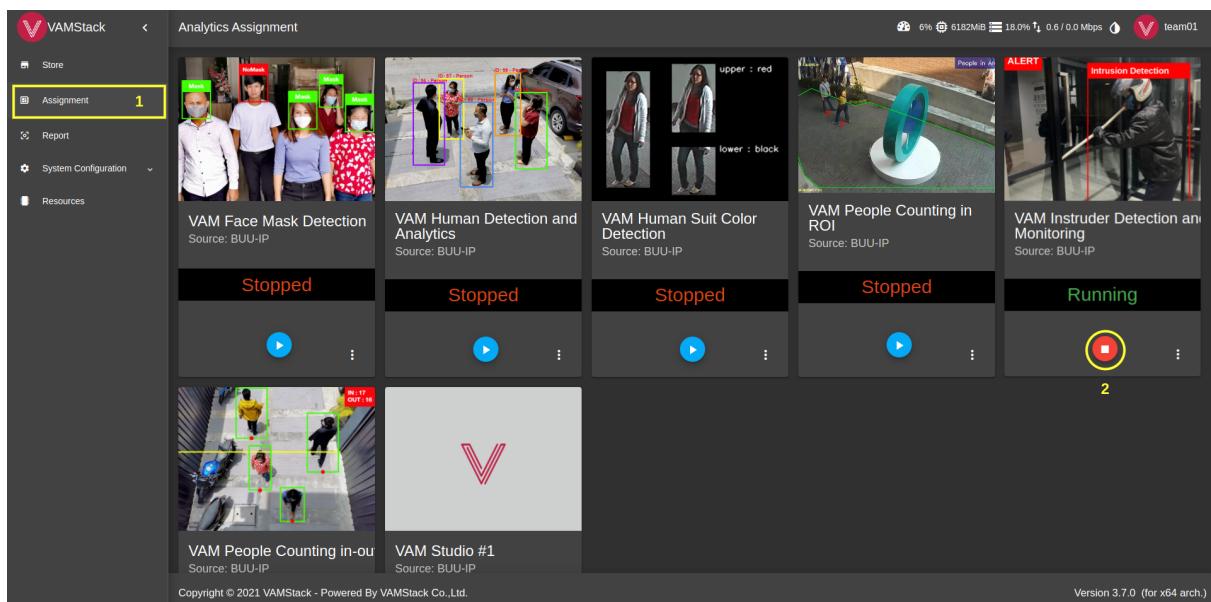
5. คลิก “Deassign” ยืนยันการลบโปรแกรมการประมวลผล



รูปที่ 1.56 ขั้นตอนการลบโปรแกรมการประมวลผล (3)

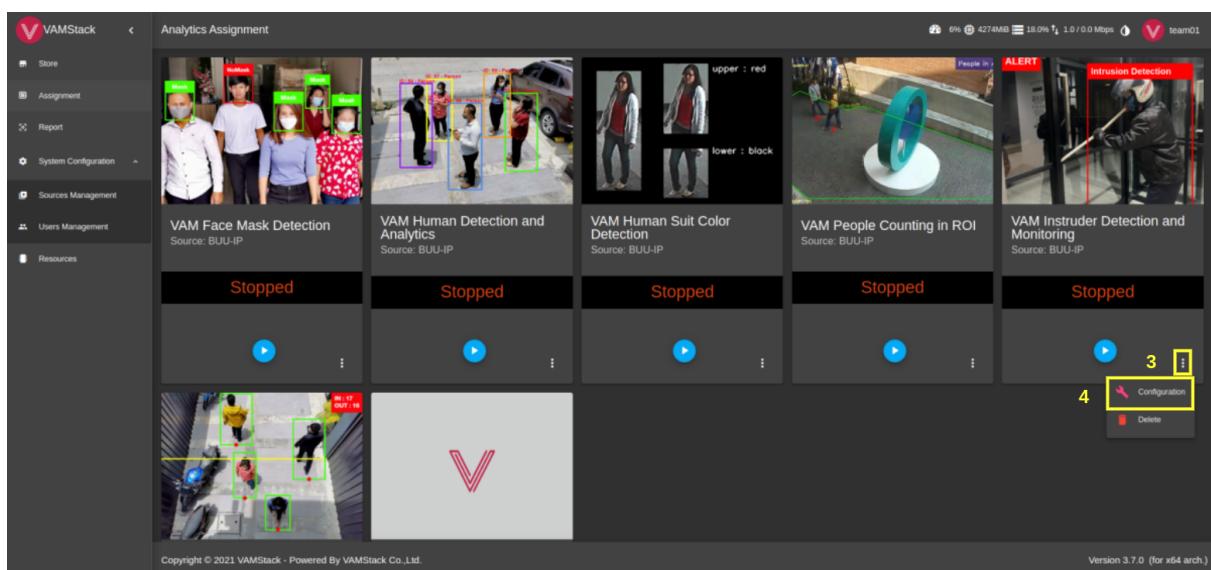
1.4.4 ขั้นตอนการตั้งค่าพื้นที่ที่สินใจ

1. คลิก “Assignment” ที่แถบเมนูด้านซ้าย
2. ปิดการทำงานของโปรแกรมการประมวลผลทุกครั้งที่ต้องการตั้งค่าพื้นที่ที่สินใจ



รูปที่ 1.57 ขั้นตอนการตั้งค่าพื้นที่ที่สินใจ (1)

3. คลิกที่จุด 3 จุด
4. เลือก “Configuration”



รูปที่ 1.58 ขั้นตอนการตั้งค่าพื้นที่ที่สินใจ (2)

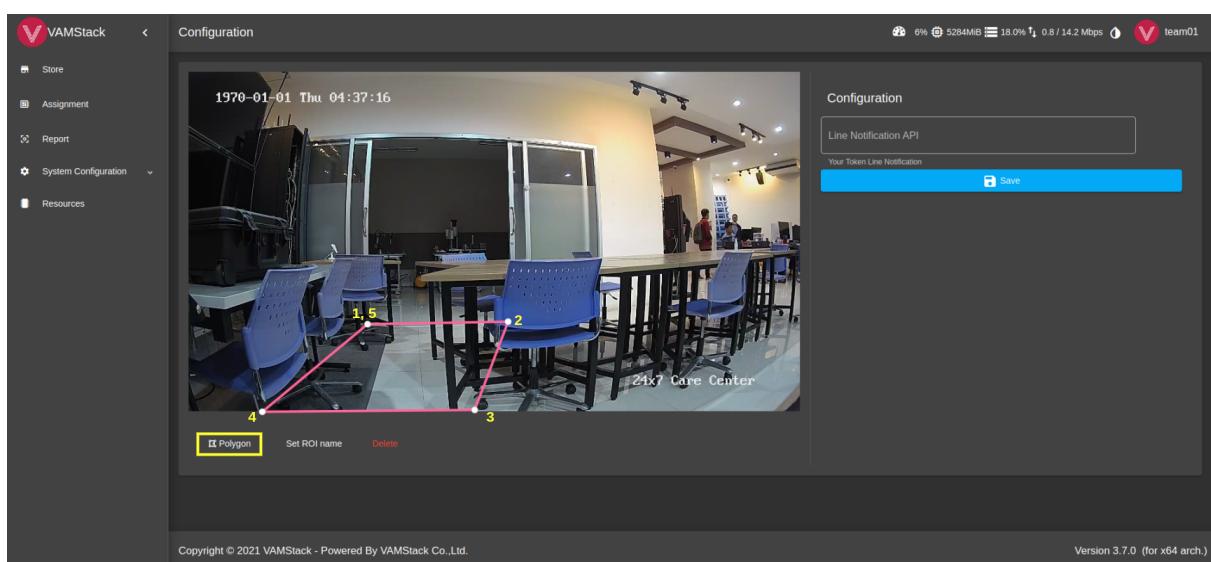
5. คลิก “Polygon” เพื่อเริ่มการวาด ROI ได้ตามต้องการ

ตัวอย่างการวาด ROI รูปสี่เหลี่ยมบริเวณพื้นที่ที่สนใจ

5.1 คลิกจุดใดจุดหนึ่งบนภาพเป็นจุดเริ่มต้นของ ROI (หมายเลข 1)

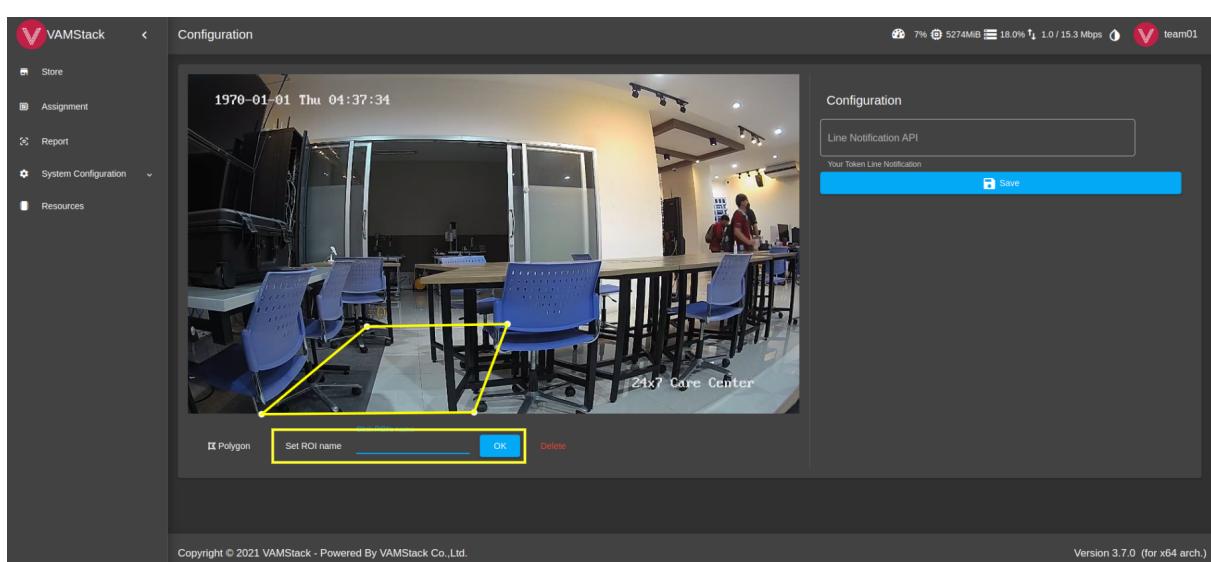
5.2 คลิกเลือกบริเวณพื้นที่ที่ต้องการอีก 3 จุด (หมายเลข 2-4)

5.3 กลับมาคลิกที่จุดเริ่มต้นเพื่อเป็นการยืนยันการวาด ROI (หมายเลข 5)



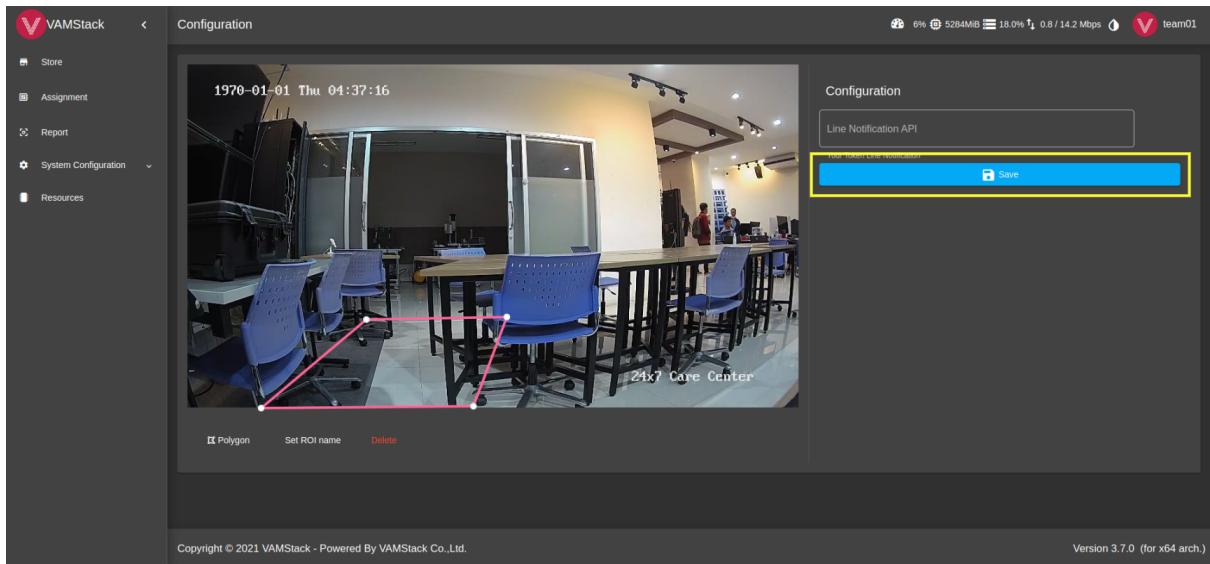
รูปที่ 1.59 ขั้นตอนการตั้งค่าพื้นที่ที่สนใจ (3)

6. หากต้องการตั้งชื่อของ ROI ให้คลิกที่ “Set ROI name”



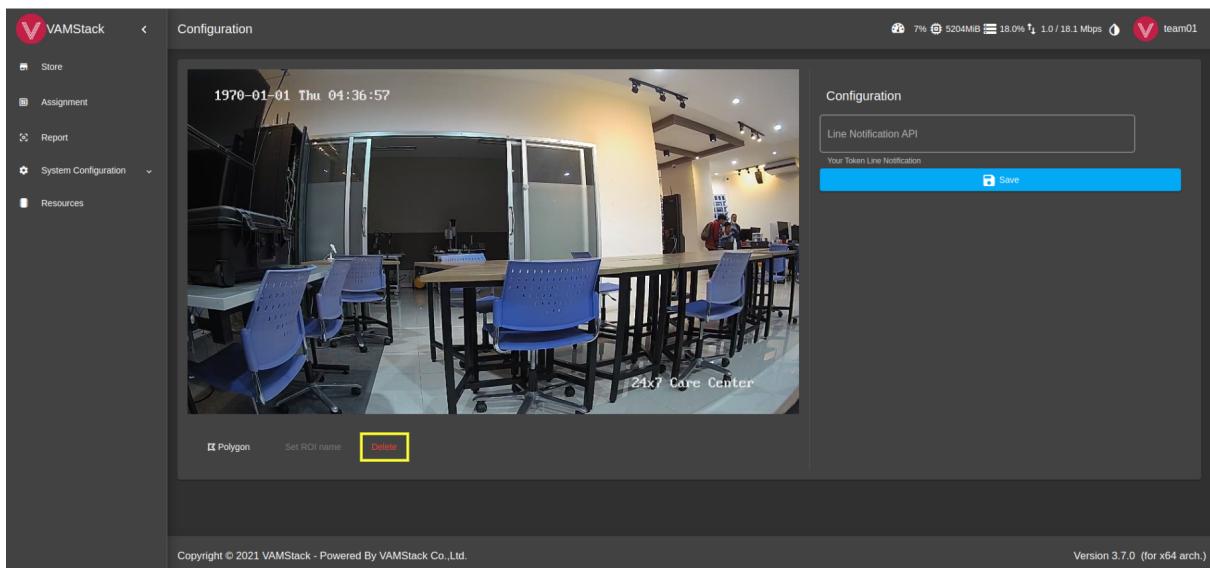
รูปที่ 1.60 ขั้นตอนการตั้งค่าพื้นที่ที่สนใจ (4)

7. เมื่อวัด ROI เสร็จให้คลิกปุ่ม “Save”



รูปที่ 1.61 ขั้นตอนการตั้งค่าพื้นที่ที่สนใจ (5)

8. หากต้องการลบ ROI ให้ทำการคลิก “Delete” และคลิกที่ภาพ 1 ครั้ง ตัวกรอบที่สร้างไว้จะหายไป จากนั้นจึงคลิกปุ่ม “Save”

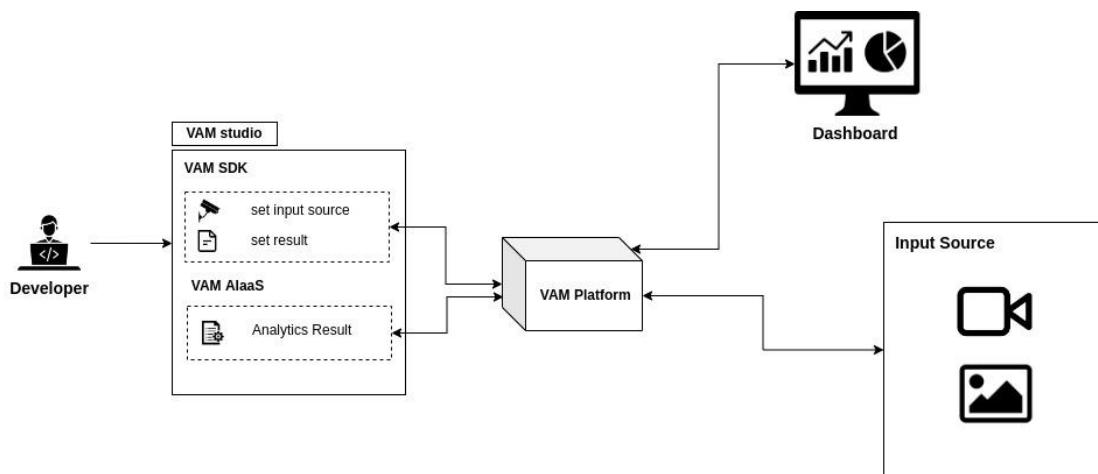


รูปที่ 1.62 ขั้นตอนการตั้งค่าพื้นที่ที่สนใจ (6)

LAB 2 | VAM Studio

VAM Studio

สำหรับให้ผู้ใช้หรือนักพัฒนาสามารถเข้ามาพัฒนาโปรแกรมการประมวลผลและวิเคราะห์ข้อมูลของตนเองในระดับเบื้องต้น รวมถึงสามารถเขียนโปรแกรมเพื่อเชื่อมต่อข้อมูลจาก VAM Platform ผ่านเครื่องมือที่ทางทีมผู้พัฒนาได้จัดเตรียมไว้ให้ใน VAM Studio เช่น VAM SDK ที่เป็นชุดตัวอย่างฟังก์ชันสำหรับใช้รับส่งข้อมูลภาพที่มีอยู่ในแพลตฟอร์มหรือใช้สำหรับการประมวลผลภาพและ VAM AlaaS ที่เป็นเซอร์วิสสำหรับการรับส่งข้อมูลการประมวลผลและวิเคราะห์ภาพเชิงปัญญาประดิษฐ์ และนอกจากนี้ผู้ใช้หรือนักพัฒนาสามารถนำสิ่งที่ตนเองพัฒนาขึ้นใน VAM Studio ไปต่อยอดเป็นแอปพลิเคชันของตนเองร่วมกับแพลตฟอร์มได้อีกด้วย



รูปที่ 2.1 แผนภาพการทำงานของ VAM Studio

LAB 2.1 การเข้าใช้งาน VAM Studio

วัตถุประสงค์

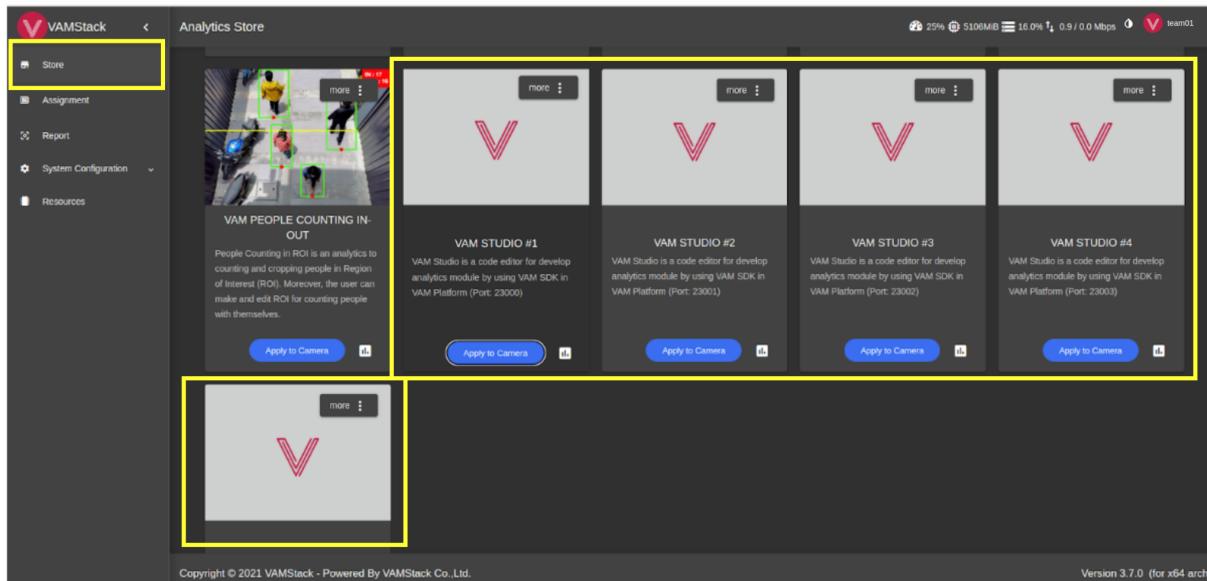
- เพื่อเรียนรู้วิธีการเข้าใช้งาน VAM Studio

อุปกรณ์ที่เกี่ยวข้อง

- ชุดการทดลอง VAM Platform

2.1.1 ขั้นตอนการเข้าใช้งานโปรแกรม VAM Studio

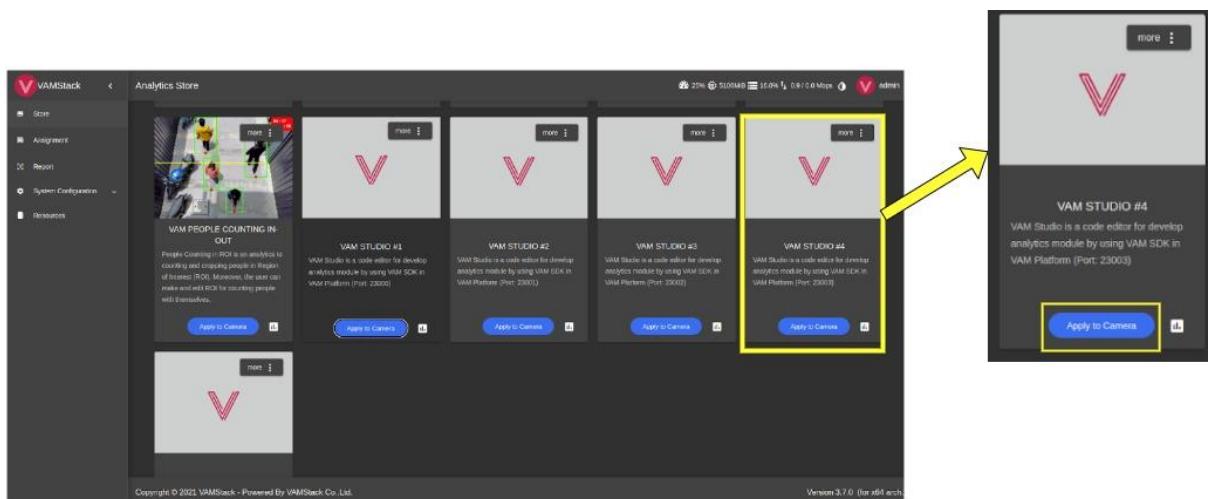
- ตรวจสอบการติดตั้งและประกอบชุดการทดลอง VAM Platform ([LAB 1.1](#))
- เข้าสู่หน้า VAM Platform โดยพิมพ์ “[10.1.1.212](#)” บนเว็บเบราว์เซอร์ และเข้าสู่ระบบของแพลตฟอร์ม (LAB 1.2 หัวข้อ [1.2.1](#))
- หลังเข้าสู่ระบบ ให้คลิก “Store” เพื่อค้นหาการ์ด “VAM STUDIO#X”



รูปที่ 2.2 การ์ด “VAM STUDIO#X”

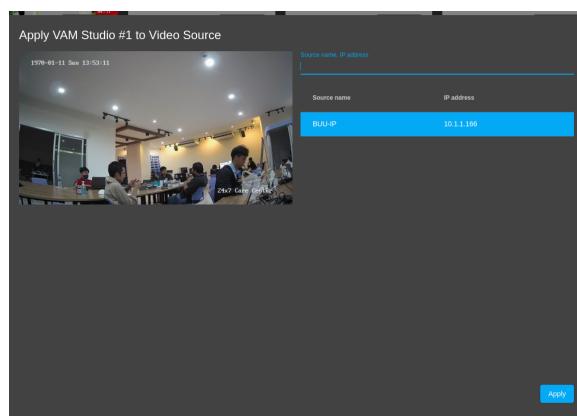
4. คลิกปุ่ม “Apply to Camera” ของการ์ด “VAM STUDIO#X” โดยมีเงื่อนไขดังนี้

- บัญชี team01 ใช้การ์ด “VAM STUDIO#1”
- บัญชี team02 ใช้การ์ด “VAM STUDIO#2”
- บัญชี team03 ใช้การ์ด “VAM STUDIO#3”
- บัญชี team04 ใช้การ์ด “VAM STUDIO#4”
- บัญชี team05 ใช้การ์ด “VAM STUDIO#5”



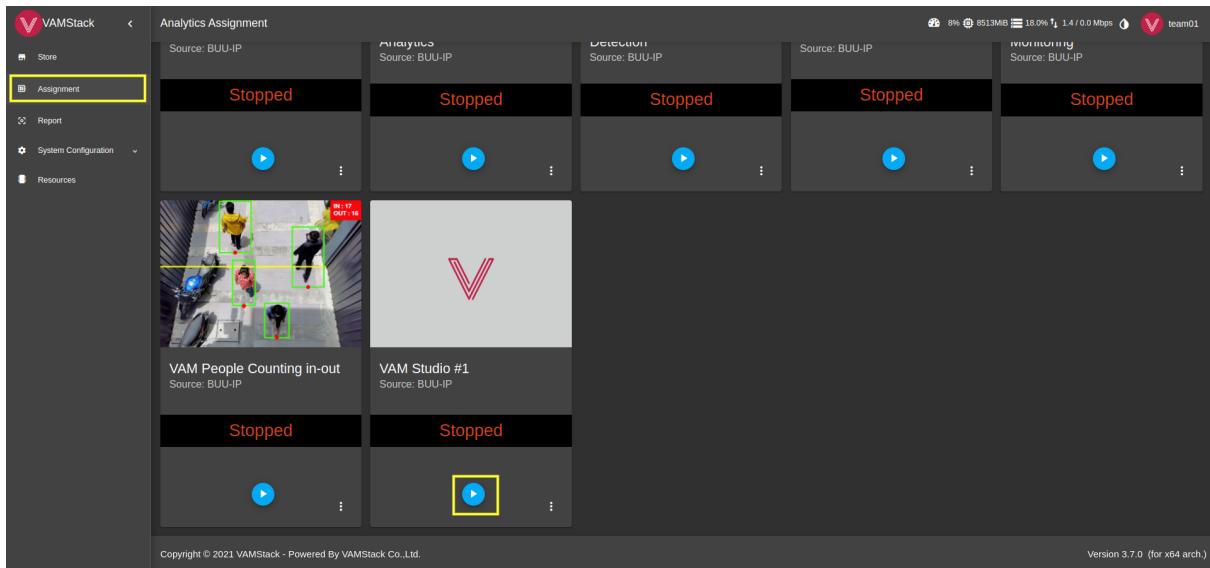
รูปที่ 2.3 เลือกการ์ด “VAM STUDIO#X” ที่ต้องการนำไปใช้

5. เลือกผูกกล้องเข้ากับโปรแกรมและคลิก “Apply”



รูปที่ 2.4 เลือกผูกกล้องเข้ากับโปรแกรม

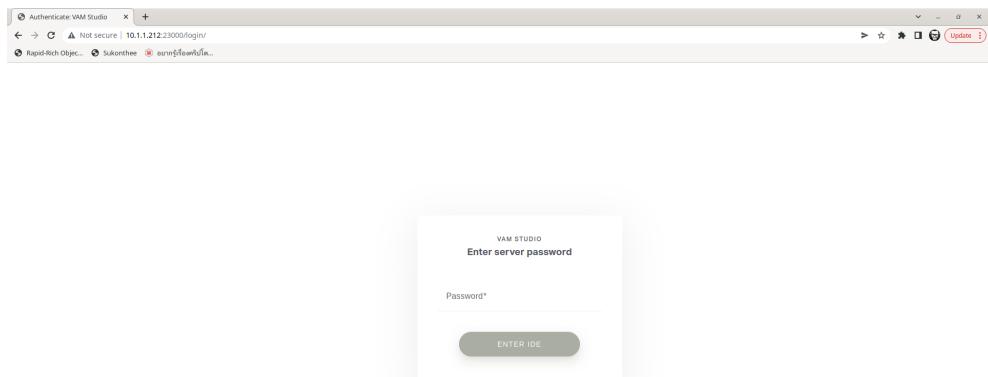
6. คลิกเมนู “Assignment” และคลิกที่ปุ่มการทำงานของ “VAM STUDIO#X” เพื่อเปลี่ยนสถานะการทำงาน



รูปที่ 2.5 เปลี่ยนสถานะการทำงานของ “VAM STUDIO#X”

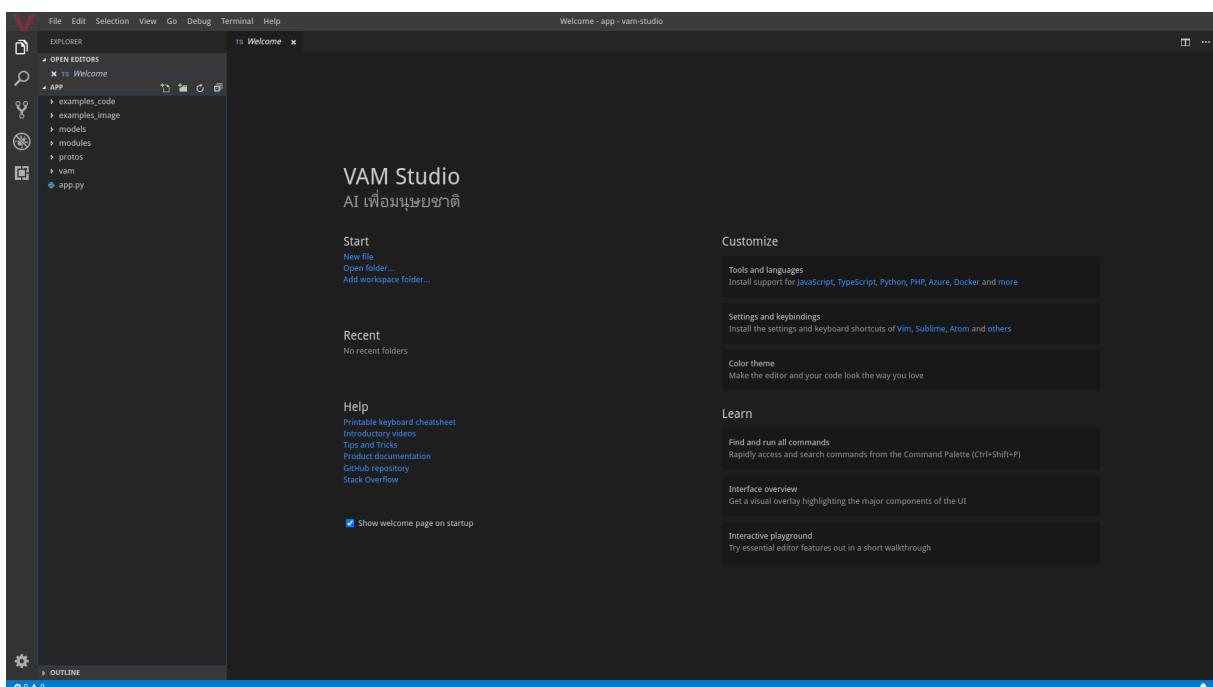
7. เมื่อ “VAM STUDIO#X” เริ่มทำงาน ให้เปิดหน้าต่างในเบราว์เซอร์ใหม่อีก 1 หน้าและทำการพิมพ์ url เพื่อเข้าสู่ระบบ VAM Studio ดังนี้

- การ์ด “VAM STUDIO#1” ใช้ localhost:23000 หรือ 10.1.1.212:23000
- การ์ด “VAM STUDIO#2” ใช้ localhost:23001 หรือ 10.1.1.212:23001
- การ์ด “VAM STUDIO#3” ใช้ localhost:23002 หรือ 10.1.1.212:23002
- การ์ด “VAM STUDIO#4” ใช้ localhost:23003 หรือ 10.1.1.212:23003
- การ์ด “VAM STUDIO#5” ใช้ localhost:23004 หรือ 10.1.1.212:23004



รูปที่ 2.6 หน้าเข้าสู่ระบบของ VAM Studio

8. กรอกรหัสผ่าน P@ssw0rd เพื่อเข้าใช้งาน VAM Studio



รูปที่ 2.7 หน้าเริ่มต้นของ VAM Studio

LAB 2.2 การนำตัวอย่างโปรแกรมเข้ามาใช้งานใน VAM Studio

วัตถุประสงค์

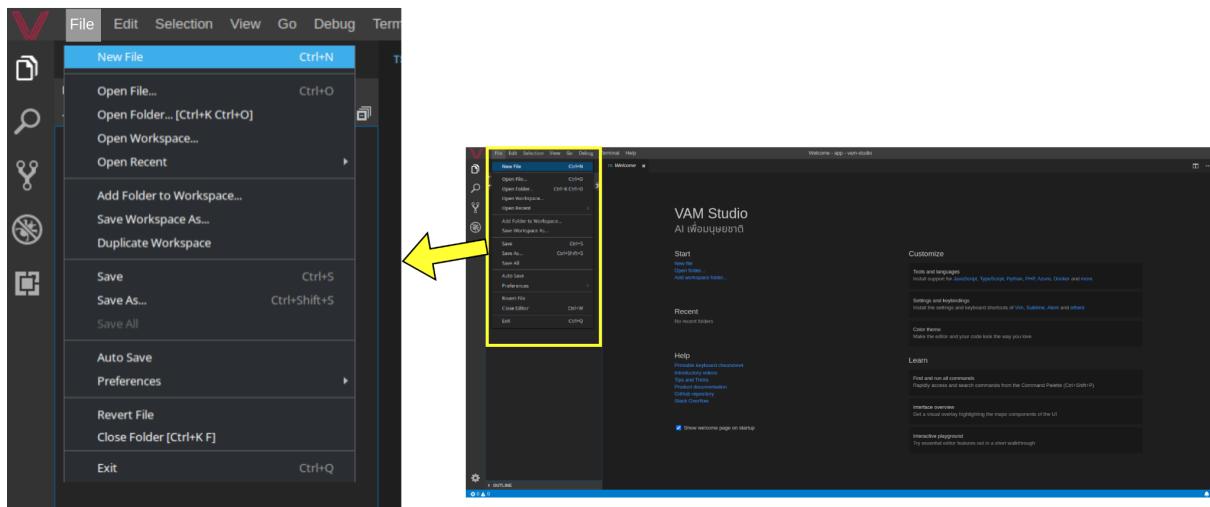
- เพื่อเรียนรู้วิธีการนำตัวอย่างโปรแกรมเข้ามาใช้งานใน VAM Studio

อุปกรณ์ที่เกี่ยวข้อง

- ชุดการทดลอง VAM Platform

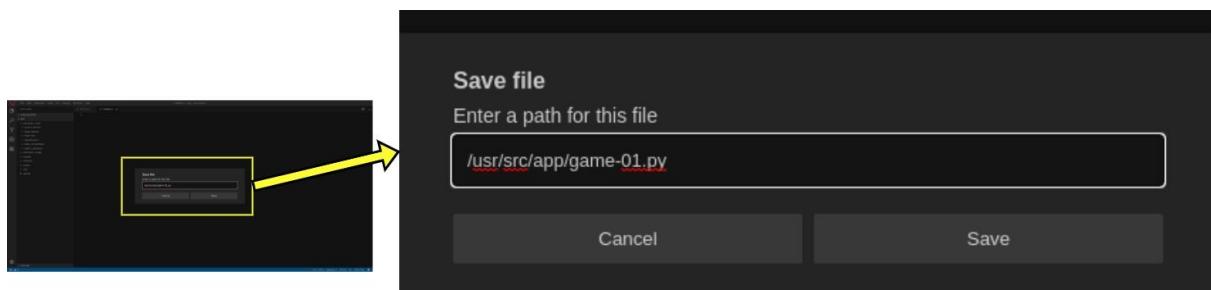
2.2.1 ขั้นตอนการนำตัวอย่างโปรแกรมเข้ามาใช้งานใน VAM Studio

- เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
- สร้างไฟล์สำหรับโปรแกรมเกมไทยตัวเลข 0 - 99 ด้วยการคลิก File > New file



รูปที่ 2.8 สร้างไฟล์ใหม่

- กด **ctrl+s** เพื่อ savefile และตั้งชื่อไฟล์เป็น `game-xx.py` โดยให้แทน `xx` ด้วยหมายเลขกลุ่ม เช่น `game-01.py` และห้ามตั้งชื่อข้ากัน



รูปที่ 2.9 บันทึกและตั้งชื่อไฟล์

4. คัดลอกชุดโค้ดของโปรแกรมลงในไฟล์ที่สร้างไว้

```
# import necessary package
import random

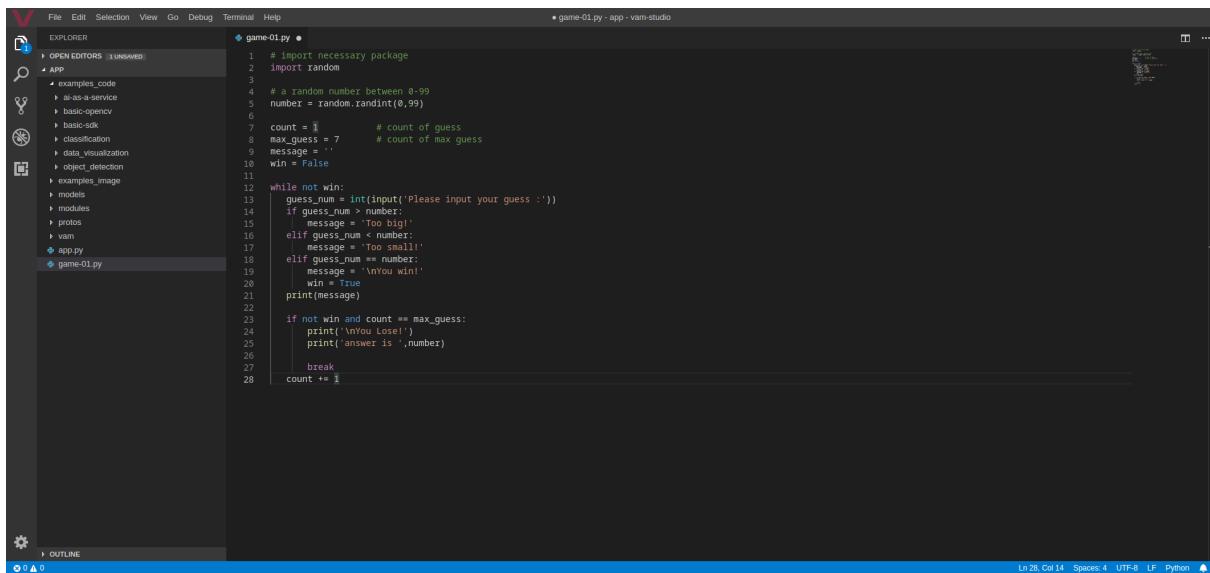
# a random number between 0-99
number = random.randint(0,99)

count = 1          # count of guess
max_guess = 7      # count of max guess
message = ''
win = False

while not win:
    guess_num = int(input('Please input your guess :'))
    if guess_num > number:
        message = 'Too big!'
    elif guess_num < number:
        message = 'Too small!'
    elif guess_num == number:
        message = '\nYou win!'
        win = True
    print(message)

    if not win and count == max_guess:
        print('\nYou Lose!')
        print('answer is ',number)

    break
count += 1
```



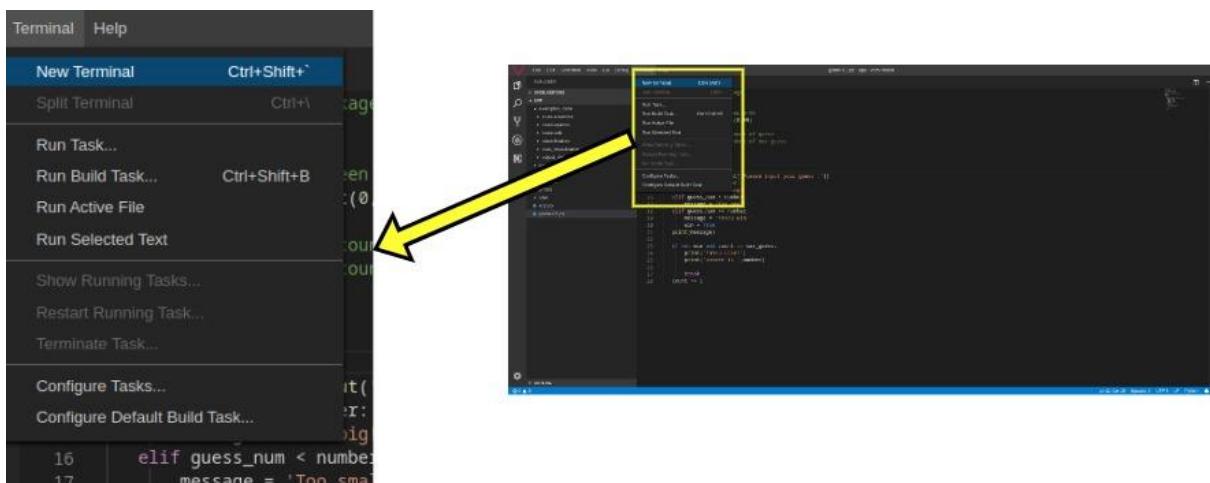
```

File Edit Selection View Go Debug Terminal Help
OPEN EDITORS 1 UNSAVED
EXPLORER game-01.py
examples_code
  app
    examples_code
      al-a-service
        basic-opencv
        basic-sdk
        classification
        data_visualization
        object_detection
      examples_image
        models
        modules
        protos
        vam
      app.py
      game-01.py
game-01.py
1 # import necessary package
2 import random
3
4 # a random number between 0-99
5 number = random.randint(0,99)
6
7 count = 0           # count of guess
8 max_guess = 7       # count of max guess
9 message = ''
10 win = False
11
12 while not win:
13     guess_num = int(input('Please input your guess :'))
14     if guess_num > number:
15         message = 'Too big!'
16     elif guess_num < number:
17         message = 'Too small!'
18     elif guess_num == number:
19         message = '\nYou Win!'
20         win = True
21     print(message)
22
23 if not win and count == max_guess:
24     print('\nYou Lose!')
25     print('answer is ',number)
26
27 break
28 count += 1

```

รูปที่ 2.10 ชุดโค้ดตัวอย่างโปรแกรม

5. บันทึกการเปลี่ยนแปลงชุดโค้ดของโปรแกรมด้วยการกด Ctrl + s ที่คีย์บอร์ด
6. เปิด “terminal” ของ VAM Studio ด้วยการคลิก Terminal > New Terminal



รูปที่ 2.11 เปิด “terminal” ของ VAM Studio

7. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 game-xx.py
# example python3 game-01.py
```

```

game-01.py
1 # import necessary package
2 import random
3
4 # a Random number between 0-99
5 number = random.randint(0,99)
6
7 count = 1           # count of guess
8 max_guess = 7       # count of max guess
9 message = ''
10 win = False
11
12 while not win:
13     guess_num = int(input('Please input your guess :'))
14     if guess_num > number:
15         message = 'Too big!'
16     elif guess_num < number:
17         message = 'Too small!'
18     else:
19         message = 'You Win!'
20         win = True
21     print(message)
22
23 if not win and count == max_guess:
24     print('You Lose!')

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Too big!
root@933f59ec2db:/usr/src/app# python3 game-01.py
Please input your guess :59
Too big!
Please input your guess :40
Too big!
Please input your guess :30
Too big!
Please input your guess :20
Too small!
Please input your guess :25
Too big!
Please input your guess :22

You win!
root@933f59ec2db:/usr/src/app#

รูปที่ 2.12 ผลการรันตัวอย่างโปรแกรมเกมไทยตัวเลข

ผลลัพธ์

```

Please input your guess :50
Too big!
Please input your guess :40
Too big!
Please input your guess :30
Too big!
Please input your guess :20
Too small!
Please input your guess :25
Too big!
Please input your guess :22

```

You win!

หมายเหตุ

ทุกครั้งที่ทำการปิดเครื่องประมวลผล ไฟล์ที่ผู้เรียนสร้างขึ้นใหม่และไฟล์ example ที่ถูกแก้ไขจะถูก Reset กลับเป็นแบบเดิมก่อนการแก้ไข

LAB 2.3 การใช้งานตัวอย่างโปรแกรมพื้นฐานของ OpenCV บน VAM Studio

วัตถุประสงค์

1. เพื่อเรียนรู้วิธีการใช้ตัวอย่างโปรแกรมที่มีใน VAM Studio
2. เพื่อเพิ่มศักยภาพพื้นฐานของไลบารี OpenCV

อุปกรณ์ที่เกี่ยวข้อง

1. ชุดการทดลอง VAM Platform

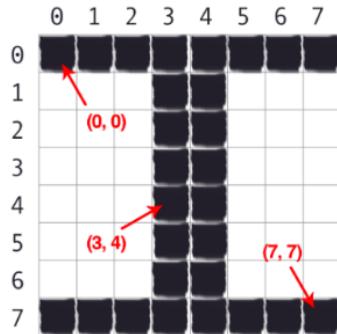
OpenCV

OpenCV คือ ไลบรารีโอเพนซอร์ซ (open-source library) ที่มีอัลกอริทึมสำหรับการประมวลผลรูปภาพและการทำงานของคอมพิวเตอร์วิศวกรรม (Computer Vision) สามารถใช้ในการทำงานต่าง ๆ เช่น การตรวจจับใบหน้า (Face Detection) การตรวจจับวัตถุ (Object Detection) และอื่น ๆ อีกมากมาย รองรับหลายภาษารวมถึง python, java และ C++

พิกเซลและสี

พิกเซล (Pixel) เป็นส่วนประกอบพื้นฐานของรูปภาพ ทุกรูปภาพจะประกอบด้วยชุดพิกเซล โดยปกติพิกเซลถือเป็น "สี" (color) หรือ "ความเข้ม" (intensity) ของแสงที่ปรากฏในตำแหน่งที่กำหนดในรูปภาพ หากเปรียบเทียบรูปภาพเป็นตาราง จุดพิกัดเริ่มต้น (0,0) ของภาพจะอยู่ที่มุมซ้ายบน และจุดพิกัดสุดท้าย (x,y) ของภาพจะอยู่ที่มุมล่างขวาของภาพ ซึ่งจุดพิกัดสุดท้ายจะมีขนาดเท่ากับขนาดของภาพเสมอ และแต่ละช่องในตารางจะมีพิกเซลเดียว

ยกตัวอย่างเช่น รูปภาพมีขนาด 600x450 หมายความว่า รูปภาพกว้าง 600 และสูง 450 พิกเซล ถ้าเทียบเป็นตาราง พิกเซลจะได้ 600 คอลัมน์และ 450 แถว โดยรวมแล้วมี $600 \times 450 = 270,000$ พิกเซล



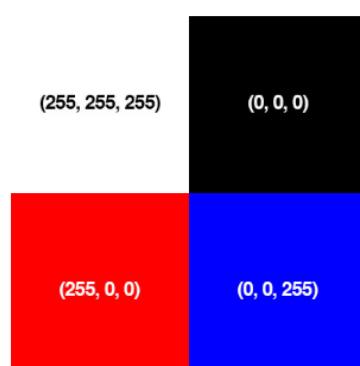
รูปที่ 2.13 เปรียบเทียบรูปภาพเป็นตาราง

พิกเซลส่วนใหญ่แสดงเป็นสองลักษณะ คือ ระดับสีเทา (Gray scale) และสี (RGB) ภาระดับสีเทาแต่ละพิกเซลมีค่าระหว่าง 0 ไปจนถึง 255 โดยที่ค่า 0 จะเป็นสีดำ และค่า 255 จะเป็นสีขาว นอกจากนี้ยังมีภาพใบหน้า ซึ่งเป็นภาพที่แต่ละพิกเซลจะมีค่าเพียง 0 หรือ 255 เท่านั้น

0 255

รูปที่ 2.14 การไล่ระดับสีเทา

ภาพสี (RGB) ในหนึ่งพิกเซลจะประกอบไปด้วย 3 ค่า คือ สีแดง (Red) สีเขียว (Green) และสีน้ำเงิน (Blue) โดยทั้งสามค่าจะมีค่าความเข้มอยู่ในช่วง 0 ถึง 255 เช่นเดียวกับภาระดับสีเทา โดยค่าพิกเซลที่แสดงบนภาพจะเกิดจากการรวมสีของทั้งสามสี



รูปที่ 2.15 ค่าสี RGB

Simple Thresholding

Simple Thresholding คือ การคัดแยกวัตถุที่สนใจจากภาพพื้นหลัง ด้วยการใช้ภาพสีเทามาเปลี่ยนเป็นภาพใบหน้าที่มีค่าพิกเซลเป็น 0 หรือ 255 จากการกำหนดค่าคงที่ขึ้นต่ำมาเทียบกับค่าพิกเซลในแต่ละพื้นที่ของภาพสีเทาเพื่อทำการเปลี่ยนค่าพิกเซล ซึ่งในไลบรารี OpenCV จะมีรูปแบบการเปลี่ยนค่าพิกเซลที่แตกต่างกันดังต่อไปนี้

- THRESH_BINARY เป็นการเปลี่ยนค่าของพิกเซลใหม่เป็น 0 ถ้าค่าของพิกเซลน้อยกว่าค่าคงที่ แต่ถ้าค่าของพิกเซลมากกว่าค่าคงที่ก็จะเปลี่ยนค่าของพิกเซลใหม่เป็น 255

$$\begin{aligned} dst(x, y) &= 0 \quad \text{if } src(x, y) < thresh \\ dst(x, y) &= 255 \quad \text{otherwise} \end{aligned} \quad (2.1)$$

เมื่อ $dst(x, y)$ คือ ค่าพิกเซลใหม่ ณ จุดคู่อันดับ (x, y)

$src(x, y)$ คือ ค่าของพิกเซล ณ จุดคู่อันดับ (x, y)

thresh คือ ค่าคงที่ที่กำหนดขึ้นในการ Thresholding



รูปที่ 2.16 ภาพใบหน้าแบบ THRESH_BINARY ที่กำหนดค่าคงที่เท่ากับ 120

- THRESH_BINARY_INV เป็นการเปลี่ยนค่าของพิกเซลใหม่เป็น 1 ถ้าค่าของพิกเซลน้อยกว่าค่าคงที่ แต่ถ้าค่าของพิกเซลมากกว่าค่าคงที่ก็จะเปลี่ยนค่าของพิกเซลใหม่เป็น 0 ดังรูปที่ 2.15 และสามารถเขียนความสัมพันธ์ได้ดังสมการที่ 2.2

$$dst(x, y) = 0 \quad \text{otherwise}$$

$$dst(x, y) = 255 \text{ if } src(x, y) > thresh \quad (2.2)$$

เมื่อ $dst(x, y)$ คือ ค่าพิกเซลใหม่ ณ จุดคู่อันดับ (x, y)

$src(x, y)$ คือ ค่าของพิกเซล ณ จุดคู่อันดับ (x, y)

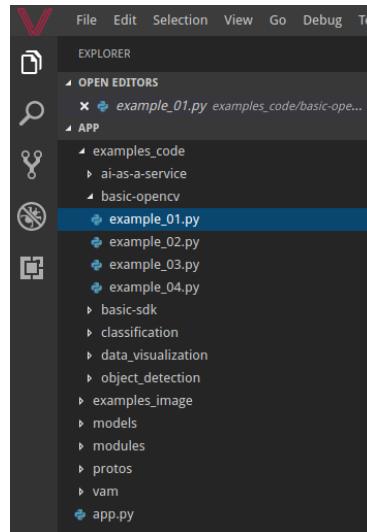
$thresh$ คือ ค่าคงที่ที่กำหนดขึ้นในการ Thresholding



รูปที่ 2.17 ภาพใบนาเรียบแบบ THRESH_BINARY_INV ที่กำหนดค่าคงที่เท่ากับ 120

2.3.1 ขั้นตอนการใช้งานตัวอย่างโปรแกรมการอ่านและบันทึกรูปภาพ

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > basic_opencv > example_01.py ที่ແຄบເມນຸດ້ານຂ້າຍ



ຮູບທີ 2.18 ການເຂົ້າໃຈໄຟລ໌ຕ້ວອຍ່າງໂປຣແກຣມກາຮຳອ່ານແລະບັນທຶກຮູບປາພ

ຕ້ວອຍ່າງໂປຣແກຣມກາຮຳອ່ານແລະບັນທຶກຮູບປາພ (Read & Write Image)

```
# import library
import cv2
import numpy as np

# select image
IMAGE_PATH="../../examples_image/coins.png"

# calling opencv imread function
image = cv2.imread(IMAGE_PATH)

# print image values
print(image)

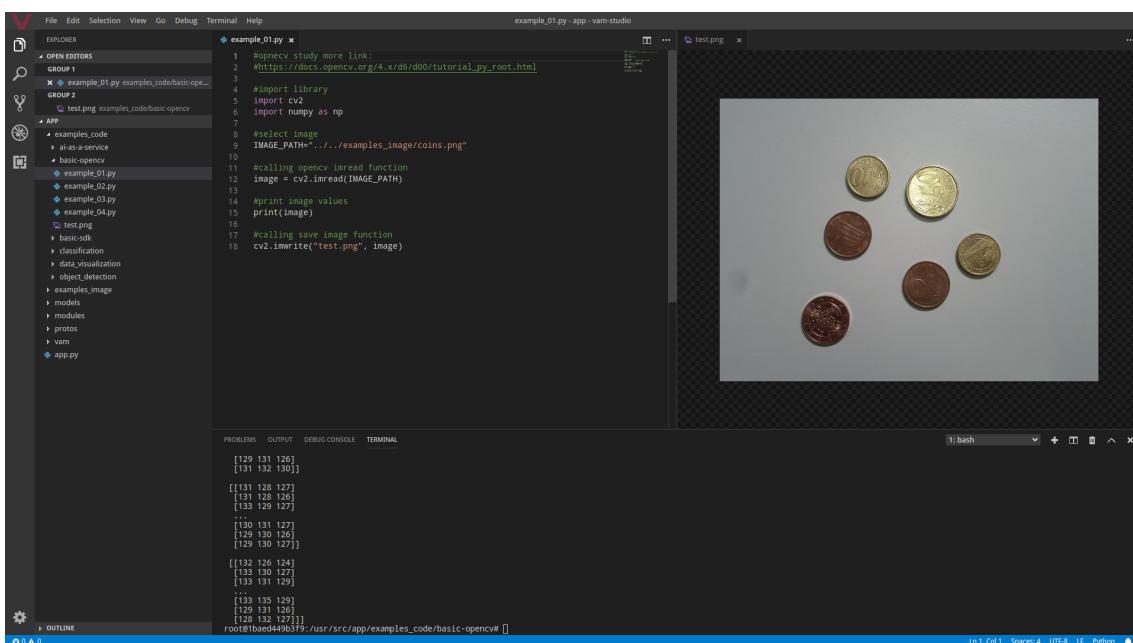
# calling save image function
cv2.imwrite("test.png", image)
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์เดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/basic-opencv
```

4. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_01.py
```



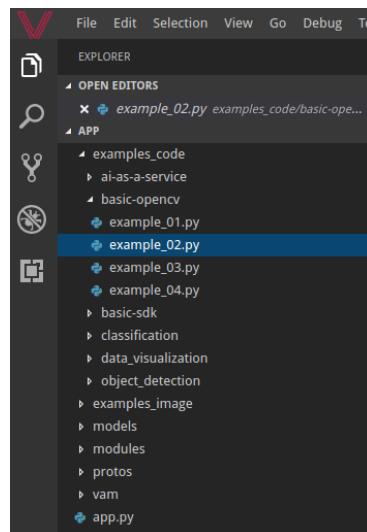
รูปที่ 2.19 ผลการรันตัวอย่างโปรแกรมการอ่านและบันทึกรูปภาพบน VAM Studio

ผลลัพธ์

test.png	image values (array)
	<pre> [[[146 143 135] [145 141 134] [144 140 135] ... [161 160 156] [161 160 156] [160 158 157]]] [[145 142 137] [145 142 136] [145 142 137] ... [133 135 129] [129 131 126] [128 132 127]]] </pre>

2.3.2 ขั้นตอนการใช้งานตัวอย่างโปรแกรมการปรับภาพสีเทา

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > basic_opencv > example_02.py ที่ແນບເມັນດ້ານຊ້າຍ



ຮູບທີ 2.20 ການເຂົ້າຄືໄຟລ໌ຕ້ວອຍ່າງໂປຣແກຣມການປັບປຸງພິບສະບັບພິບສະບັບ

ตัวอย่างโปรแกรมการปรับภาพสีเทา (Gray Scale Image)

```
# import library
import cv2
import numpy as np

# select image
IMAGE_PATH= "../examples_image/coins.png"

# calling opencv imread function
image = cv2.imread(IMAGE_PATH)
print(image)

# calling grayscale image function
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
print(gray)

# calling save image function
cv2.imwrite("test.png", gray)
```

- ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงโฟลเดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/basic-opencv
```

- ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_02.py
```

The screenshot shows the VAM Studio interface with the following details:

- File Explorer:** Shows the project structure with files like example_02.py, test.png, and various OpenCV examples.
- Code Editor:** Displays the Python script `example_02.py` which reads an image of coins and saves it as a grayscale image.
- Output View:** Shows the terminal output with the command `cv2.imwrite("test.png", gray)` and the resulting array values for the image.
- Image Preview:** A preview window shows the original image of several coins on a light background.
- Terminal:** Shows the command `cv2.imwrite("test.png", gray)` and the resulting array values for the image.

รูปที่ 2.21 ผลการรันตัวอย่างโปรแกรมการปรับภาพสีเทาบน VAM Studio

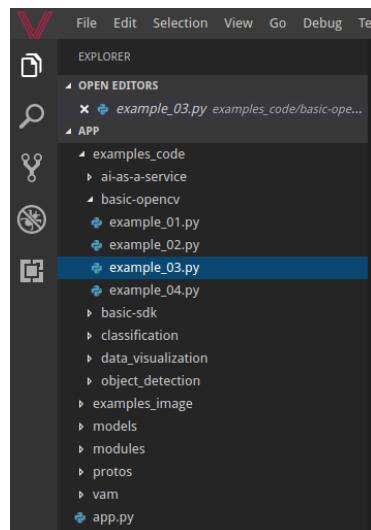


```

    ...
[133 135 129]
[129 131 126]
[128 132 127]]]
```

2.3.3 ขั้นตอนการใช้งานตัวอย่างโปรแกรมการตัดภาพเฉพาะส่วน

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > basic_opencv > example_03.py ที่แถบเมนูด้านซ้าย



รูปที่ 2.22 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการตัดภาพเฉพาะส่วน

ตัวอย่างโปรแกรมการตัดภาพเฉพาะส่วน (Crop Image)

```
# import library
import cv2
import numpy as np

# select image
IMAGE_PATH="../../examples_image/coins.png"

# calling opencv imread function
image = cv2.imread(IMAGE_PATH)
print(image.shape)

# cropped interesting roi in image
```

```
# image[y1:y2, x1:x2]

# where (x1, y1) is topleft of boundingbox
# where (x2, y2) is bottomright of boundingbox
cropped_image = image[80:280, 150:330]
print(cropped_image.shape)

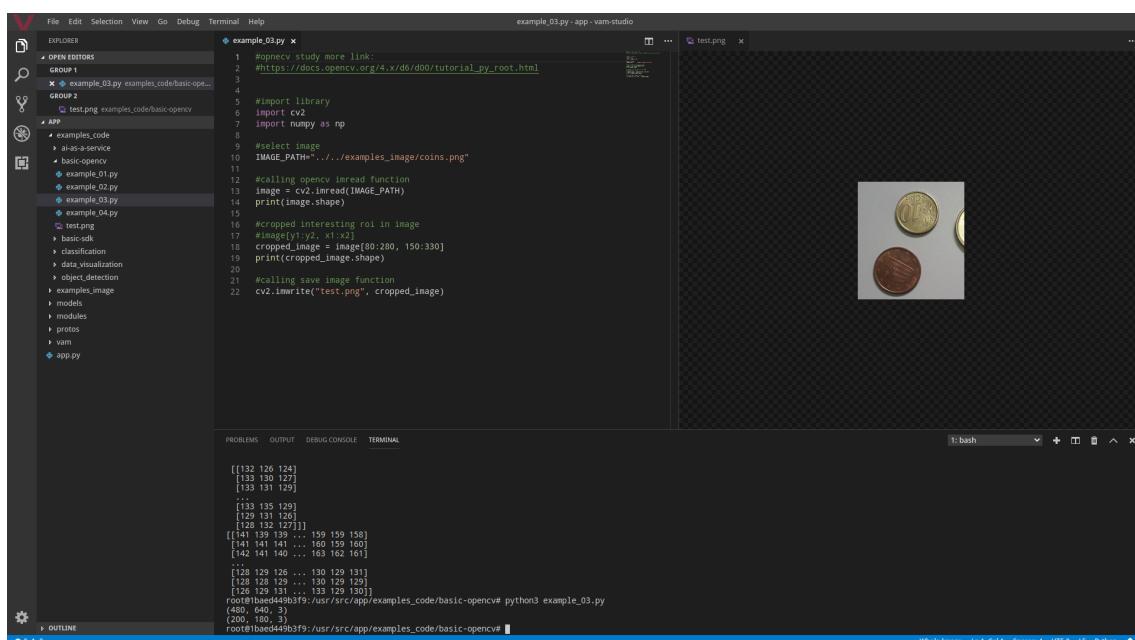
# calling save image function
cv2.imwrite("test.png", cropped_image)
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงโฟลเดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/basic-opencv
```

4. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_03.py
```



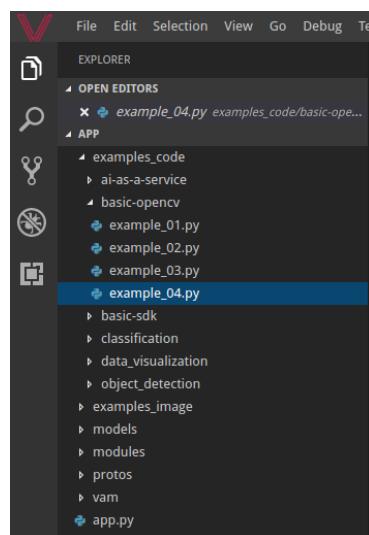
รูปที่ 2.23 ผลการรันตัวอย่างโปรแกรมการตัดภาพเฉพาะส่วนบน VAM Studio

ผลลัพธ์

test.png	image.shape (h,w,chanals)	cropped_image.shape (h,w,chanals)
	(480, 640, 3)	(200, 180, 3)

2.3.4 ขั้นตอนการใช้งานตัวอย่างโปรแกรมการปรับภาพใบหน้า

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > basic_opencv > example_04.py ที่แท็บเมนูด้านซ้าย



รูปที่ 2.24 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการปรับภาพใบหน้า

ตัวอย่างโปรแกรมการปรับภาพใบหน้ารี (Binary Image)

```
# import library
import cv2
import numpy as np

# select image
IMAGE_PATH= "../../examples_image/coins.png"

# calling opencv imread function
image = cv2.imread(IMAGE_PATH)
print(image.shape)

# calling grayscale image function
gray = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)

# calling binary image function
TRESHOLD_VALUE =200
(T, thresh) = cv2.threshold(gray , TRESHOLD_VALUE , 255,
cv2.THRESH_BINARY)

# calling save image function
cv2.imwrite("test.png", thresh)
```

- ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงโฟลเดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/basic-opencv
```

- ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_04.py
```

```

File Edit Selection View Go Debug Terminal Help
OPEN EDITORS
GROUP 1
example_04.py examples_code/basic-opencv
GROUP 2
test.png examples_code/basic-opencv
APP
examples_code
ai-as-a-service
basic-opencv
example_01.py
example_02.py
example_03.py
example_04.py
test.png
basic-sdk
classification
data_visualization
object_detection
examples_image
models
modules
protos
vam
app.py

example_04.py x
example 04.py - app - vam-studio
example_04.py
1 #opencv study more link:
2 https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html
3
4 #import library
5 import cv2
6 import numpy as np
7
8 #select image
9 IMAGE_PATH='../../examples_image/coins.png'
10
11 #calling opencv imread function
12 image = cv2.imread(IMAGE_PATH)
13 print(image)
14
15 #calling grayscale image function
16 gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
17
18 #calling binary image function
19 TRESHOLD_VALUE=200
20 (T, thresh) = cv2.threshold(gray, TRESHOLD_VALUE, 255, cv2.THRESH_BINARY)
21 print(thresh)
22
23 #calling save image function
24 cv2.imwrite("test.png", thresh)

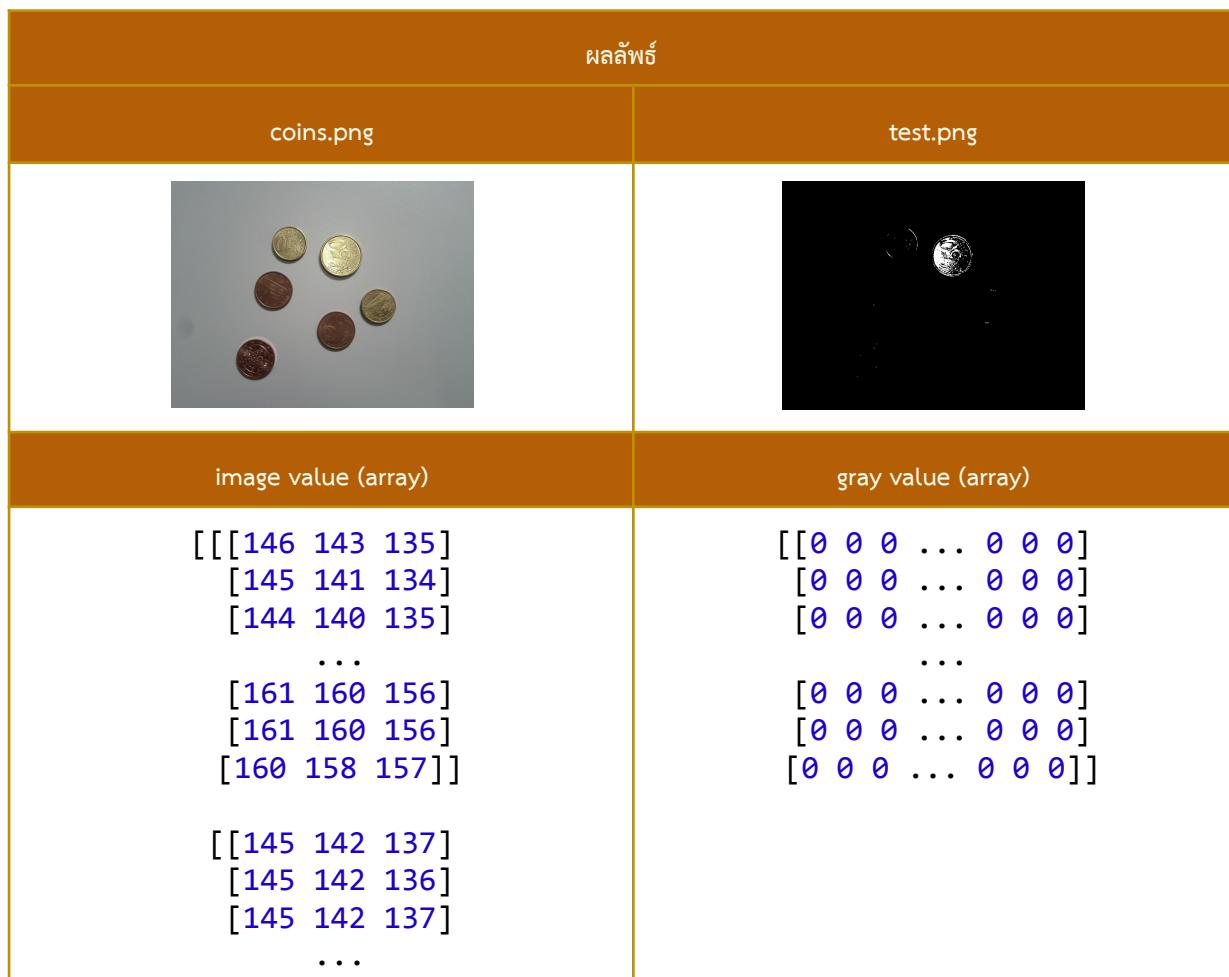
[130 131 127]
[129 130 126]
[129 130 127]

[[132 126 124]
[133 130 127]
[131 131 129]
...
[133 130 129]
[133 130 125]
[128 132 127]]
[[0 0 0 ... 0 0 0]
[0 0 0 ... 0 0 0]
[0 0 0 ... 0 0 0]
...
[0 0 0 ... 0 0 0]
[0 0 0 ... 0 0 0]
[0 0 0 ... 0 0 0]]

[130 131 127]
[129 130 126]
[129 130 127]

[[132 126 124]
[133 130 127]
[131 131 129]
...
[133 130 129]
[133 130 125]
[128 132 127]]
[[0 0 0 ... 0 0 0]
[0 0 0 ... 0 0 0]
[0 0 0 ... 0 0 0]
...
[0 0 0 ... 0 0 0]
[0 0 0 ... 0 0 0]
[0 0 0 ... 0 0 0]]
```

รูปที่ 2.25 ผลการรันตัวอย่างโปรแกรมการปรับภาพใบนาใน VAM Studio



[133 135 129]	
[129 131 126]	
[128 132 127]]]	

โจทย์

1. จากตัวอย่างโปรแกรมการตัดภาพเฉพาะส่วน ให้ผู้เรียนทำการเปลี่ยนรูปเครื่องเป็นรูปภาพใบหน้า และปรับตัวเลขให้ตัดภาพเฉพาะส่วนใบหน้า

สรุปและบันทึกผล

.....
.....
.....
.....
.....
.....
.....
.....

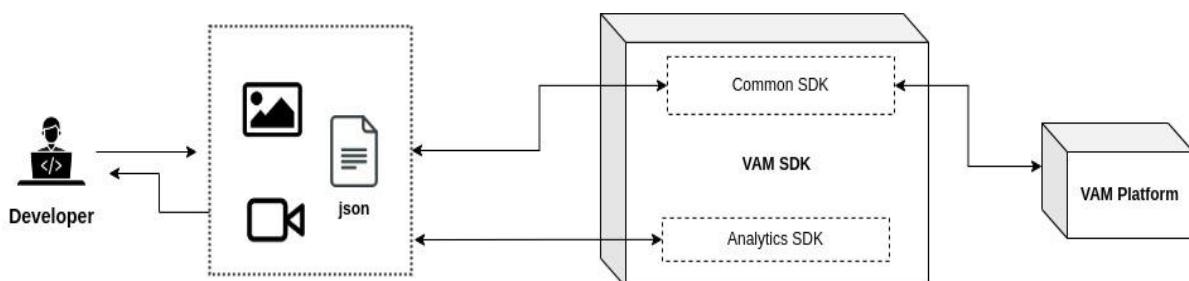
LAB 3 | VAM SDK

วัตถุประสงค์

1. เพื่อทำความรู้จักและเข้าใจกระบวนการทำงานของ VAM SDK เป็นต้น
2. เพื่อให้ผู้เรียนสามารถนำความรู้จาก VAM SDK ไปต่อยอดงานของตนเอง

VAM SDK

Video Analytics Management Software Development Toolkit (VAM SDK) คือ เครื่องมือที่ช่วยให้ผู้ใช้หรือนักพัฒนาสามารถพัฒนาโปรแกรมการประมวลผลและวิเคราะห์ข้อมูลบน VAM Studio โดยจะมีฟังก์ชันพื้นฐานสำหรับการประมวลผลทั่วไป (Analytics SDK) และฟังก์ชันสำหรับการรับ-ส่งข้อมูลระหว่างผู้ใช้และ VAM Platform (Common SDK) ให้ผู้ใช้หรือนักพัฒนานำมาสร้างหรือพัฒนาต่อยอดเป็นโปรแกรมประมวลผลและวิเคราะห์ข้อมูลตัวใหม่บนแพลตฟอร์ม



รูปที่ 3.1 แผนภาพการทำงานของ VAM SDK

LAB 3.1 พังก์ชันสำหรับการรับข้อมูลกล้องจาก VAM Platform

วัตถุประสงค์

- เพื่อเรียนรู้วิธีการใช้ VAM SDK ที่เป็นพังก์ชันสำหรับการรับข้อมูลจาก VAM Platform

อุปกรณ์ที่เกี่ยวข้อง

- ชุดการทดลอง VAM Platform

3.2.1 พังก์ชันสำหรับการรับข้อมูลของกล้องที่อยู่ใน VAM Platform

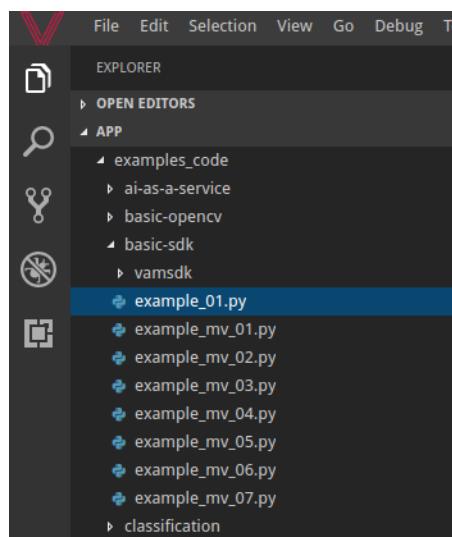
พังก์ชันสำหรับการรับข้อมูลของกล้องที่อยู่ใน VAM Platform จะทำหน้าที่ในการดึงข้อมูลกล้องทั้งหมดที่ลงทะเบียนใน VAM Platform ส่งกลับไปยังผู้ใช้ ซึ่งข้อมูลที่ผู้ใช้ได้รับจะถูกจัดเก็บอยู่ในรูปแบบของข้อมูล Json

พังก์ชันที่เกี่ยวข้อง

- listSource(machine_ip)
 - machine_ip : รับพารามิเตอร์ประเภทตัวอักษรที่เป็น IP address ของเครื่องที่ติดตั้ง VAM Platform
 - ส่งคืนข้อมูลกล้องที่ลงทะเบียนใน VAM Platform ในรูปแบบของข้อมูล Json

ขั้นตอนการใช้ตัวอย่างโปรแกรมการรับข้อมูลของกล้อง

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > basic-sdk > example_01.py ที่แสดงเมนูด้านซ้าย



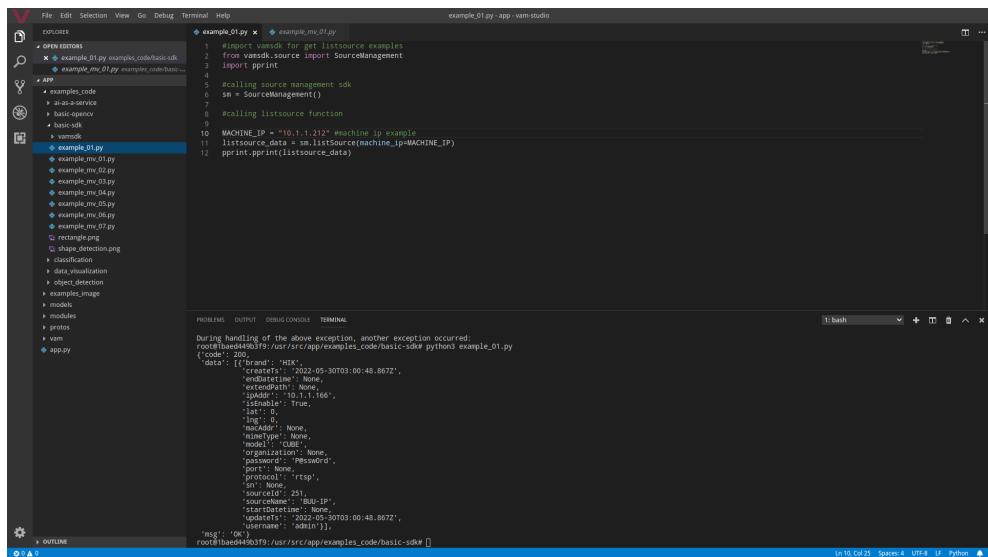
รูปที่ 3.2 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการรับข้อมูลของกล้อง

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงโฟลเดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/basic-sdk
```

4. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_01.py
```



รูปที่ 3.3 ผลการรันตัวอย่างโปรแกรมการรับข้อมูลของกล้องบน VAM Studio

ผลลัพธ์

```
{  
  "code": 200,  
  "msg": "OK",  
  "data": [  
    {  
      "sourceId": 250,  
      "sourceName": "BDH-VAM",  
      "brand": "HIK",  
      "model": "DOME",  
      "ipAddr": "10.9.0.30",  
      "macAddr": null,  
      "sn": null,  
      "protocol": "rtsp",  
      "port": null,  
      "extendPath": null,  
      "mimeType": null,  
      "startDatetime": null,  
      "endDatetime": null,  
      "isEnabled": true,  
      "lat": 0,  
      "lng": 0,  
      "id": 1  
    }  
  ]  
}
```

```

    "username": "admin",
    "password": "vamstack_admin",
    "createTs": "2022-05-19T10:33:16.413Z",
    "updateTs": "2022-05-19T10:33:16.413Z",
    "organization": null
  }
]
}

```

3.2.2 พังก์ชันสำหรับการรับภาพจากกล้องที่อยู่ใน VAM Platform

พังก์ชันสำหรับการรับภาพจากกล้องที่อยู่ใน VAM Platform ทำหน้าที่ดึงข้อมูลภาพเพียง 1 ภาพ (Snapshot) จากกล้องที่ผูกอยู่กับการต์ “VAM STUDIO#X” ในหน้า “Assignment” ที่ผู้เรียนใช้งานอยู่และสามารถนำภาพที่ดึงออกมาไปประมวลผลข้อมูลในส่วนต่อได้ ด้วยวิธีนี้ นำข้อมูลภาพมาตรวจสอบจับบุคคล ตรวจนับจำนวนบุคคลภายในภาพ เป็นต้น

พังก์ชันที่เกี่ยวข้อง

1. `get_vamsnapshot()`
 - 1.1. พังก์ชันนี้จะไม่มีการรับพารามิเตอร์
 - 1.2. ส่งคืนข้อมูลรูปภาพที่ได้จากการรับภาพเป็นแพลตฟอร์มเป็นข้อมูลประเภท `numpy.ndarray`
 - 1.3. ส่งคืนข้อมูลประเภทตัวเลขที่เป็นลำดับของการตั้งค่า “Assignment” ที่ใช้งาน
 - 1.4. ส่งคืนข้อมูลประเภทตัวเลขที่เป็นลำดับเลขลงท้ายของกล้องที่ใช้งาน
 - 1.5. ส่งคืนข้อมูลประเภท `timestamp` ที่เป็นเวลาที่ทำการดึงภาพ

ขั้นตอนการนำตัวอย่างโปรแกรมการรับภาพจากกล้องมาใช้ใน VAM Studio

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. สร้างไฟล์แล้วตั้งชื่อเป็น `snapshot-xx.py` โดยให้แทน `xx` ด้วยหมายเลขกลุ่ม เช่น `snapshot-01.py` และห้ามตั้งชื่อซ้ำกัน ([LAB 2.2](#))
3. คัดลอกชุดโค้ดของโปรแกรมลงในไฟล์ที่สร้างไว้

ตัวอย่างโปรแกรมการรับภาพจากกล้อง

```
# import vamsdk for get snapshot examples
```

```

from vam.vamsdk import get_vamsnapshot
import cv2

# SDK: Platform get camera source
(img, assignmentId, sourceId, ts) = get_vamsnapshot()

# return result from get vam snapshot sdk
cv2.imwrite("results.jpg", img)

```

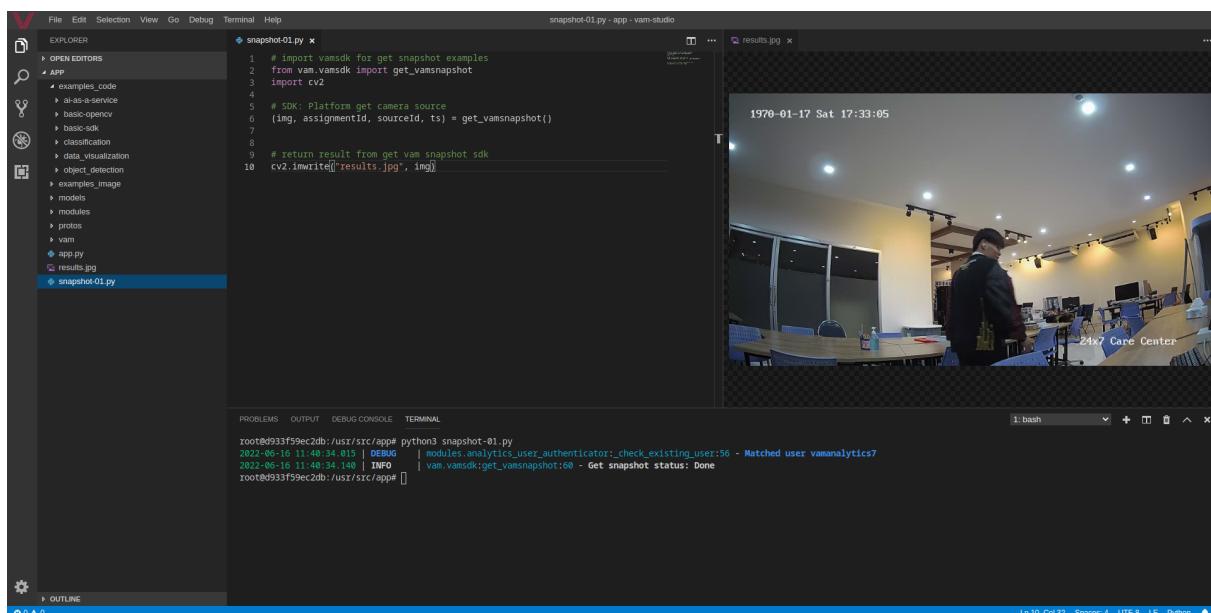
4. บันทึกการเปลี่ยนแปลงชุดโค้ดของโปรแกรมด้วยการกด Ctrl + s ที่คีย์บอร์ด

5. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```

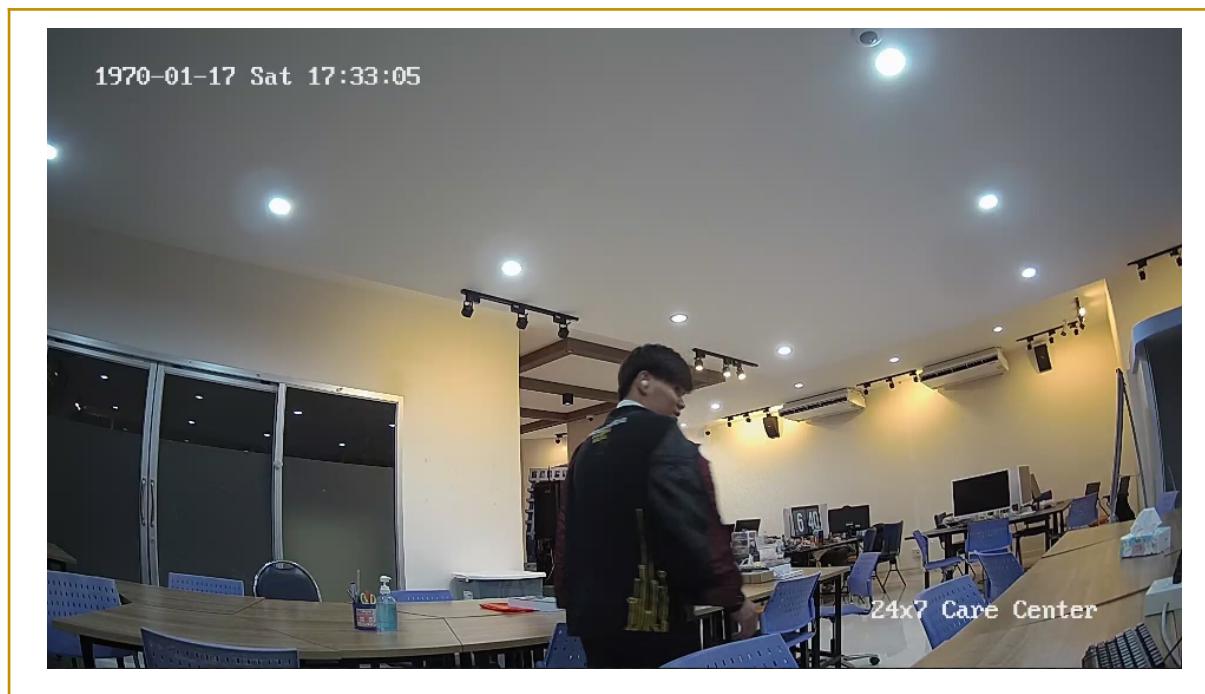
python3 snapshot-xx.py
# example python3 snapshot-01.py

```



รูปที่ 3.4 ผลการรันตัวอย่างโปรแกรมการรับข้อมูลของกล้องบน VAM Studio

ผลลัพธ์



3.2.3 พังก์ชันสำหรับการรับข้อมูลพื้นที่ที่ส่งมาจาก VAM Platform

พังก์ชันสำหรับการรับข้อมูลพื้นที่ที่ส่งมาจาก VAM Platform ทำหน้าที่ในการดึงค่าพื้นที่ที่สนใจ (Region Of Interest: ROI) ที่ถูกตั้งค่าไว้จากกล้องที่ผูกอยู่กับการ์ด “VAM STUDIO#X” ในหน้า “Assignment” (LAB 1.4 หัวข้อ [1.4.4](#)) ที่ผู้เรียนใช้งานอุปกรณ์ในรูปแบบของ json และสามารถนำผลลัพธ์ที่ได้ไปใช้งานต่อได้ เช่น นำพิกัดที่ได้จาก SDK มาทำการวาดพื้นที่ที่ส่งใจให้ผ่านฟังก์ชัน draw_roi ได้

พังก์ชันที่เกี่ยวข้อง

1. `get_vamroi(assignmentId)`
 - 1.1. `assignmentId` : รับพารามิเตอร์ประเภทตัวเลขที่เป็นลำดับของการ์ดใน “Assignment” ที่ใช้งาน
 - 1.2. ส่งคืนค่าสเตตัสประเภท Boolean (True or False)
 - 1.3. ส่งคืนข้อมูลประเภท Json ที่เป็นรายละเอียดของกรอบพื้นที่ที่สนใจ
2. `draw_roi(image,RoiList)`
 - 2.1. `image` : รับพารามิเตอร์เป็นรูปภาพประเภท numpy.ndarray
 - 2.2. `RoiList` : รับพารามิเตอร์ข้อมูลประเภท Json ที่เป็นรายละเอียดของกรอบพื้นที่ที่ส่งมาจากฟังก์ชัน `get_vamroi`
 - 2.3. ส่งคืนข้อมูลรูปภาพที่ได้จากการกรอบ ROI สำเร็จเป็นข้อมูลประเภท numpy.ndarray

ขั้นตอนการนำตัวอย่างโปรแกรมการรับข้อมูลพื้นที่ที่สินใจมาใช้ใน VAM Studio

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. สร้างไฟล์แล้วตั้งชื่อเป็น drawROI-xx.py โดยให้แทน xx ด้วยหมายเลขคุณ เช่น drawROI-01.py และห้ามตั้งชื่อซ้ำกัน ([LAB 2.2](#))
3. คัดลอกชุดโค้ดของโปรแกรมลงในไฟล์ที่สร้างไว้

ตัวอย่างโปรแกรมการรับข้อมูลพื้นที่ที่สินใจ

```
# import vamsdk for get roi examples
from vam.vamsdk import get_vamsnapshot
from vam.vamsdk import get_vamroi
from vam.vamsdk import draw_roi

import cv2
import os

# SDK: Platform get camera source
(img, assignmentId, sourceId, ts) = get_vamsnapshot()

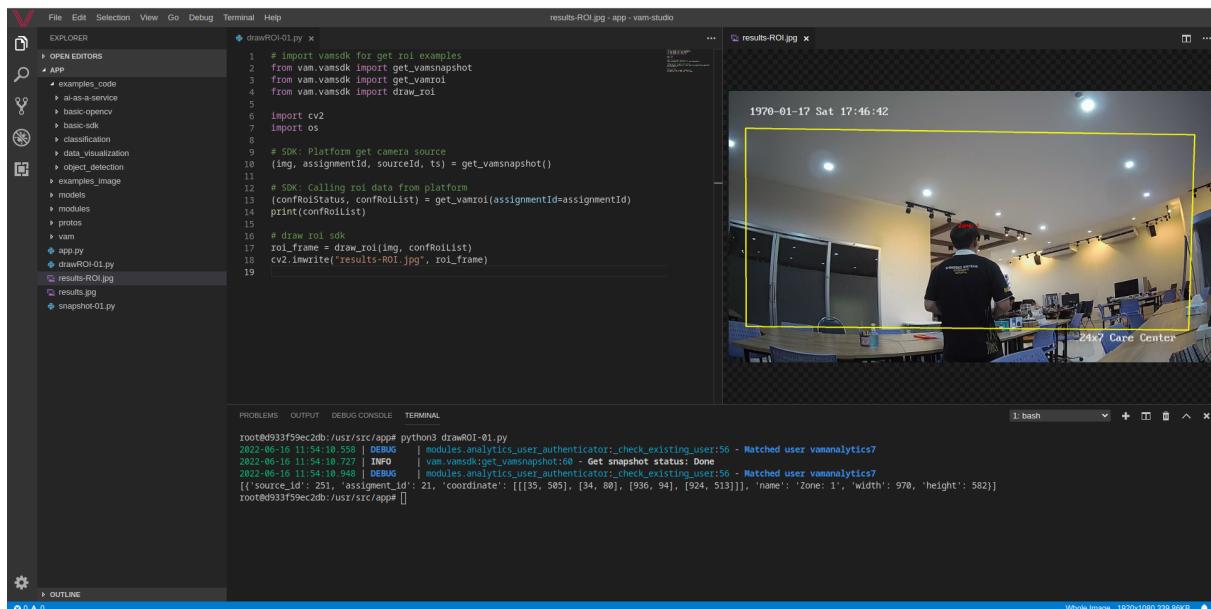
# SDK: Calling roi data from platform
(confRoiStatus, confRoiList) =
get_vamroi(assignmentId=assignmentId)
print(confRoiList)

# draw roi sdk
roi_frame = draw_roi(img, confRoiList)
cv2.imwrite("results-ROI.jpg", roi_frame)
```

4. บันทึกการเปลี่ยนแปลงชุดโค้ดของโปรแกรมด้วยการกด Ctrl + s ที่คีย์บอร์ด

5. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 drawROI-xx.py
# example python3 drawROI-01.py
```



รูปที่ 3.5 ผลการรันตัวอย่างโปรแกรมการรับข้อมูลพื้นที่ที่สนใจบน VAM Studio

ผลลัพธ์	
results-ROI.jpg	confRoiList
	[{"source_id": 251, "assignment_id": 21, "coordinate": [[[35, 505], [34, 80], [936, 94], [924, 513]]], "name": "Zone: 1", "width": 970, "height": 582}]

LAB 3.2 พังก์ชันพื้นฐานสำหรับการประมวลผลและวิเคราะห์ข้อมูล

วัตถุประสงค์

- เพื่อเรียนรู้วิธีการใช้ VAM SDK ที่เป็นพังก์ชันพื้นฐานสำหรับการประมวลผลและวิเคราะห์ข้อมูล

อุปกรณ์ที่เกี่ยวข้อง

- ชุดการทดลอง VAM Platform

3.2.1 พังก์ชันการตรวจจับบุคคล

พังก์ชันการตรวจจับบุคคล (People Detection) ทำหน้าที่ในการประมวลผลและวิเคราะห์ข้อมูลภาพ เพื่อค้นหาและตัวจับบุคคลภายในภาพ

พังก์ชันที่เกี่ยวข้อง

- `people_detection(frame,conf)`
 - `frame` : รับพารามิเตอร์เป็นรูปภาพประเภท numpy.ndarray
 - `conf` : รับพารามิเตอร์เป็นตัวเลขค่าความมั่นใจขั้นต่ำที่ยอมรับได้ว่าวัตถุที่ตรวจพบเป็นมนุษย์ มีค่าระหว่าง 0 ถึง 1
 - ส่งคืนข้อมูลตำแหน่งกรอบตรวจจับวัตถุและค่าความน่าจะเป็นที่ได้
 - ส่งคืนข้อมูลรูปภาพจากการประมวลผลเป็นข้อมูลประเภท numpy.ndarray

ขั้นตอนการนำตัวอย่างโปรแกรมการตรวจจับบุคคลมาใช้ใน VAM Studio

- เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
- สร้างไฟล์แล้วตั้งชื่อเป็น `people-detection-xx.py` โดยให้แทน xx ด้วยหมายเลขกลุ่ม เช่น `people-detection-01.py` และห้ามตั้งชื่อซ้ำกัน ([LAB 2.2](#))
- คัดลอกชุดโค้ดของโปรแกรมลงในไฟล์ที่สร้างไว้

ตัวอย่างโปรแกรมการตรวจจับบุคคล

```
# import vamsdk for set get analytics results examples
from vam.vamsdk import people_detection

import cv2
# select image
IMAGE_PATH = "examples_image/person.png"
image = cv2.imread(IMAGE_PATH)

# calling detection people function
people_bboxes, results_frame_people = people_detection(frame=image,
conf=0.6)
print(people_bboxes)

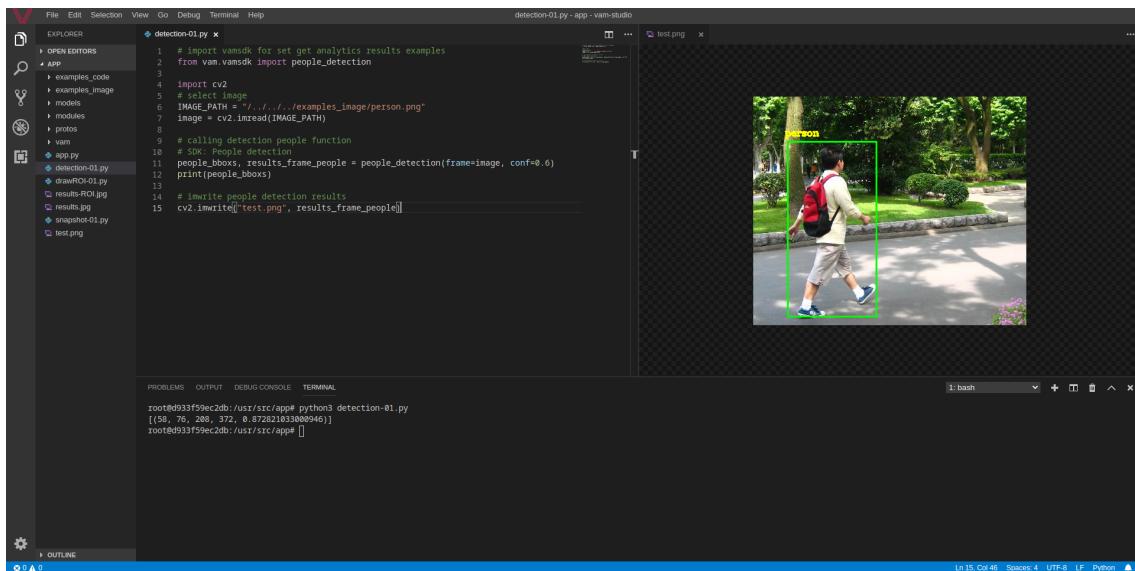
# returning people_bboxes
# [(x1, y1, x2, y2, conf)]

# where (x1, y1) is topleft of boundingbox
# where (x2, y2) is bottomright of boundingbox
# where conf is confident of an object

# imwrite people detection results
cv2.imwrite("test.png", results_frame_people)
```

4. บันทึกการเปลี่ยนแปลงชุดโค้ดของโปรแกรมด้วยการกด Ctrl + s ที่คีย์บอร์ด
5. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 people-detection-xx.py
# example python3 people-detection-01.py
```



รูปที่ 3.6 ผลการรันตัวอย่างโปรแกรมการตรวจจับบุคคลบน VAM Studio

ผลลัพธ์	
test.png	people_bboxes
	[(58, 76, 208, 372, 0.872821033000946)]

3.2.2 พังก์ชันการตรวจนับบุคคล

พังก์ชันการตรวจนับบุคคล (People counting) เป็นพังก์ชันที่ต่อยอดมาจากการใช้พังก์ชันตรวจจับบุคคล โดยพังก์ชันนี้จะทำหน้าที่ในการนับข้อมูลที่ได้จากพังก์ชันตรวจจับบุคคล

พังก์ชันที่เกี่ยวข้อง

1. people_counting(people_bboxes)
 - 1.1. people_bboxes : รับพารามิเตอร์ข้อมูลการตรวจจับบุคคลจากพังก์ชันการตรวจจับบุคคล
 - 1.2. ส่งคืนค่าเป็นตัวเลขจำนวนบุคคลที่ตรวจพบได้

ขั้นตอนการนำตัวอย่างโปรแกรมการนับบุคคลมาใช้ใน VAM Studio

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. สร้างไฟล์แล้วตั้งชื่อเป็น people-cnt-xx.py โดยให้แทน xx ด้วยหมายเลขกลุ่ม เช่น people-cnt-01.py และห้ามตั้งชื่อซ้ำกัน ([LAB 2.2](#))
3. คัดลอกชุดโค้ดของโปรแกรมลงในไฟล์ที่สร้างไว้

ตัวอย่างโปรแกรมการนับบุคคล

```
# import vamsdk for set get analytics results examples
from vam.vamsdk import people_detection
from vam.vamsdk import objects_counting
import cv2

# select image
IMAGE_PATH = "examples_image/person.png"
image = cv2.imread(IMAGE_PATH)

# SDK: People detection
people_bboxes, results_frame_people = people_detection(frame=image,
conf=0.6)

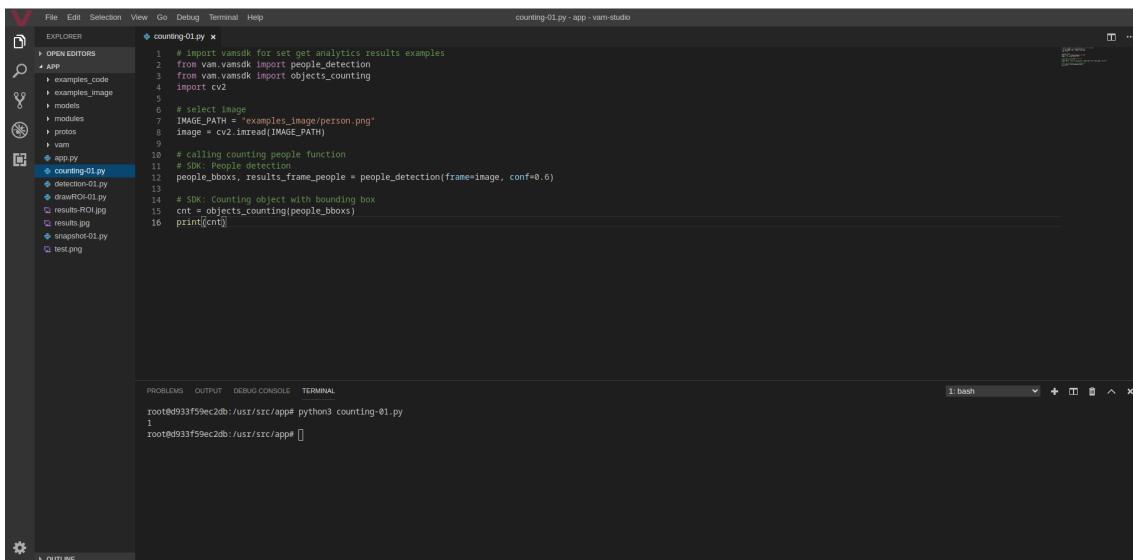
# imwrite people detection results
cv2.imwrite("test.png", results_frame_people)

# SDK: Counting object with bounding box
```

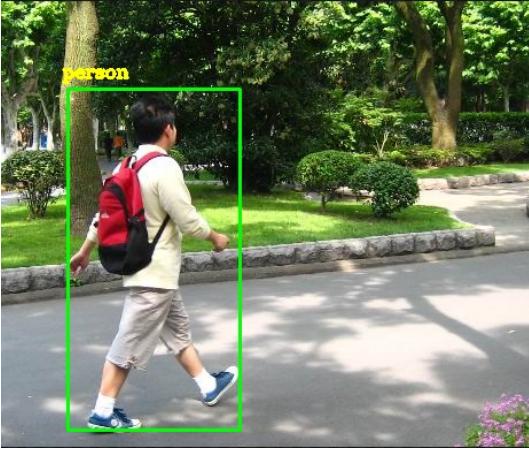
```
cnt = objects_counting(people_bboxs)
print(cnt)
```

4. บันทึกการเปลี่ยนแปลงชุดโค้ดของโปรแกรมด้วยการกด Ctrl + s ที่สีฟอร์ด
5. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 people-counting-xx.py
# example python3 people-counting-01.py
```



รูปที่ 3.7 ผลการรันตัวอย่างโปรแกรมการตรวจนับบุคคลบน VAM Studio

ผลลัพธ์	
test.png	cnt
 A photograph of a person walking away from the camera on a paved path in a park-like setting. A green rectangular bounding box surrounds the person's body, and the word "person" is written in yellow above it.	1

LAB 3.3 พัฒนาสำหรับการส่งข้อมูลที่ประมวลผลเข้า VAM Platform

วัตถุประสงค์

- เพื่อเรียนรู้วิธีการใช้ VAM SDK ที่เป็นพัฒนาสำหรับการส่งข้อมูลไปยัง VAM Platform

อุปกรณ์ที่เกี่ยวข้อง

- ชุดการทดลอง VAM Platform

3.3.1 พัฒนาการส่งภาพผลลัพธ์การประมวลผลไปที่ VAM Platform

พัฒนาการส่งภาพผลลัพธ์การประมวลผลไปที่ VAM Platform ทำหน้าที่ส่งภาพผลลัพธ์ที่ผ่านการประมวลผลและวิเคราะห์ข้อมูลหลังจากสามารถดึงข้อมูลภาพจาก VAM Platform กลับมาแสดงผลในหน้าการทำงานของกราฟ “VAM STUDIO#X” ของเมนู “Assignment” (LAB 1.4 หัวข้อ [1.4.2](#)) ที่ผู้เรียนใช้งาน

พัฒนาที่เกี่ยวข้อง

- set_vamlivefigure(frame,assignmentId)
 - frame : รับพารามิเตอร์เป็นรูปภาพประเภท numpy.ndarray
 - assignmentId : รับพารามิเตอร์ประเภทตัวเลขที่เป็นลำดับของการ์ดใน “Assignment” ที่ใช้อยู่
 - ฟังก์ชันนี้ไม่มีการส่งคืนค่าใด ๆ

ขั้นตอนการนำตัวอย่างโปรแกรมการส่งภาพผลลัพธ์มาใช้ใน VAM Studio

- เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
- สร้างไฟล์แล้วตั้งชื่อเป็น vamlivefigure-xx.py โดยให้แทน xx ด้วยหมายเลขกลุ่ม เช่น vamlivefigure-01.py และห้ามตั้งชื่อซ้ำกัน ([LAB 2.2](#))
- คัดลอกชุดโค้ดของโปรแกรมลงในไฟล์ที่สร้างไว้

ตัวอย่างโปรแกรมการส่งภาพผลลัพธ์การประมวลผลไปที่ VAM Platform

```
# import vamsdk for set analytics live figure examples
```

```

from vam.vamsdk import get_vamsnapshot
from vam.vamsdk import people_detection
from vam.vamsdk import set_vamlivefigure
import cv2

# SDK: Platform get camera source
(img, assignmentId, sourceId, ts) = get_vamsnapshot()

# SDK: People detection
people_bboxes, results_frame_people = people_detection(frame=img,
conf=0.6)

# SDK: Platform live result
set_vamlivefigure(frame=results_frame_people,
assignmentId=assignmentId)

```

4. บันทึกการเปลี่ยนแปลงชุดโค้ดของโปรแกรมด้วยการกด Ctrl + s ที่คีย์บอร์ด
5. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```

python3 vamlivefigure-xx.py
# example python3 vamlivefigure-01.py

```

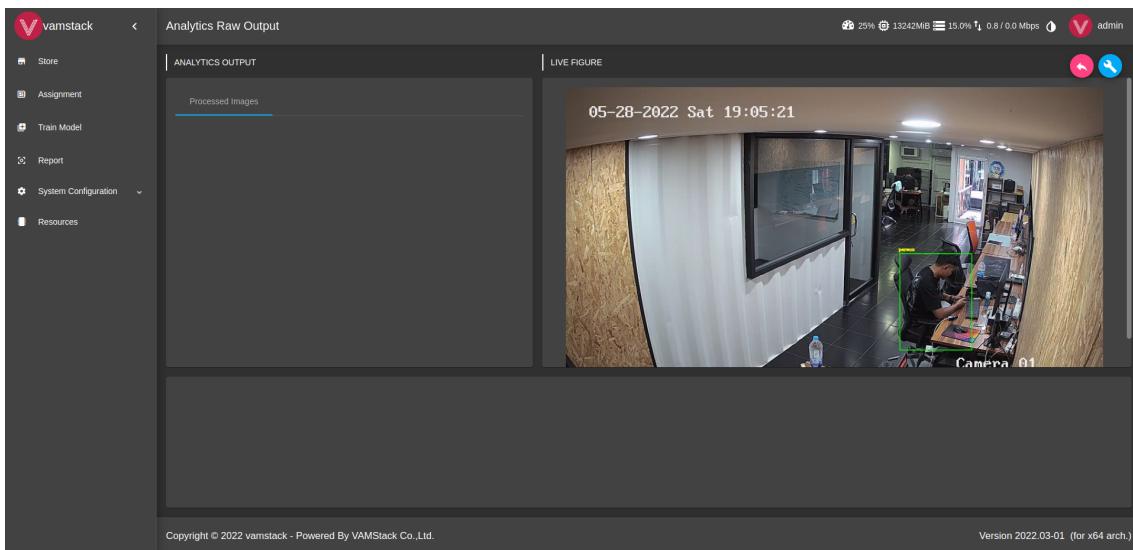
The screenshot shows the VAM Studio interface. The code editor displays the Python script 'vamlivefigure-01.py'. The terminal window at the bottom shows the command 'python3 vamlivefigure-01.py' being run, followed by several lines of log output indicating the program's execution and successful connection to the VAM system.

```

root@933f59ec2d0:/usr/src/app# python3 vamlivefigure-01.py
2022-06-16 14:25:04.772 | DEBUG | modules.analytics_user_authenticator:_check_existing_user:56 - Matched user vamanalytics7
2022-06-16 14:25:04.945 | INFO | van.vamsdk:get_vamsnapshot:60 - Get snapshot status: Done
2022-06-16 14:25:05.331 | DEBUG | modules.analytics_user_authenticator:_check_existing_user:56 - Matched user vamanalytics7
2022-06-16 14:25:05.420 | INFO | van.vamsdk:set_vamlivefigure:64 - Live figure status: OK
root@933f59ec2d0:/usr/src/app#

```

รูปที่ 3.8 ผลการรันตัวอย่างโปรแกรมการส่งภาพผลลัพธ์บน VAM Studio



รูปที่ 3.9 ผลการส่งภาพผลลัพธ์การประมวลผลไปที่ VAM Platform

3.3.2 พังก์ชันการส่งข้อมูลผลลัพธ์การประมวลผลไปที่ VAM Platform

พังก์ชันการส่งข้อมูลผลลัพธ์การประมวลผลไปที่ VAM Platform ทำหน้าที่ส่งข้อมูลผลลัพธ์ที่ผ่านการประมวลผลและวิเคราะห์ข้อมูลที่ถูกจัดเก็บในรูปแบบของ Json ไปแสดงผลในหน้าการทำงานของกรณี “VAM STUDIO#X” ของเมนู “Assignment” (LAB 1.4 หัวข้อ [1.4.2](#)) ที่ผู้เรียนใช้งาน

พังก์ชันที่เกี่ยวข้อง

1. set_data2vamMetadata(assignmentId, sourceId, ts, image, bboxs, objectClass, cnt=cnt)
 - 1.1. assignmentId: รับพารามิเตอร์ประเภทตัวเลขที่เป็นลำดับของการ์ดใน “Assignment” ที่ใช้งาน
 - 1.2. sourceId: รับพารามิเตอร์ประเภทตัวเลขที่เป็นลำดับเลขลงทะเบียนของกล้องที่ใช้งาน
 - 1.3. ts: รับพารามิเตอร์ข้อมูลประเภท timestamp ที่เป็นเวลาที่ทำการดึงภาพมาใช้งาน
 - 1.4. image: รับพารามิเตอร์เป็นรูปภาพประเภท numpy.ndarray
 - 1.5. bboxs: รับพารามิเตอร์ข้อมูลของกรอบตรวจจับวัตถุ
 - 1.6. objectClass: รับพารามิเตอร์ประเภทตัวอักษรเป็นการระบุว่าวัตถุที่ตรวจจับได้คืออะไร
 - 1.7. cnt: รับพารามิเตอร์ประเภทตัวเลขที่เป็นจำนวนของวัตถุที่ตรวจจับ
 - 1.8. พังก์ชันนี้ไม่มีการส่งคืนค่าใด ๆ

ขั้นตอนการนำตัวอย่างโปรแกรมการส่งผลลัพธ์มาใช้ใน VAM Studio

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))

2. สร้างไฟล์แล้วตั้งชื่อเป็น set-data-xx.py โดยให้แทน xx ด้วยหมายเลขกลุ่ม เช่น set-data-01.py และห้ามตั้งชื่อซ้ำกัน ([LAB 2.2](#))
3. คัดลอกชุดโค้ดของโปรแกรมลงในไฟล์ที่สร้างไว้

ตัวอย่างโปรแกรมการส่งผลลัพธ์การประมวลผลข้อมูล

```
# import vamsdk for send data to platform examples
from vam.vamsdk import get_vamsnapshot
from vam.vamsdk import people_detection
from vam.vamsdk import set_data2vamMetadata
import cv2
import os

ANALYTICS_ID = int(os.environ.get("ANALYTICS_ID"))

# SDK: Platform get camera source
(img, assignmentId, sourceId, ts) = get_vamsnapshot()
print(ts)

# SDK: People detection
(people_bboxs, results_frame_people) = people_detection(frame=img,
conf=0.6)

# SDK: Counting object with bounding box
cnt = objects_counting(people_bboxs)
print(cnt)

# SDK: Send analyzed data to metadata API
# ***If not have data to input in the argument parameter. Please
set None to that argument.
set_data2vamMetadata(assignmentId=assignmentId, sourceId=sourceId,
ts=ts, image=results_frame_people, bboxs=people_bboxs,
objectClass="person", cnt=cnt)
```

4. บันทึกการเปลี่ยนแปลงชุดโค้ดของโปรแกรมด้วยการกด Ctrl + s ที่คีย์บอร์ด
5. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 set-data-xx.py
# example python3 set-data-01.py
```

The screenshot shows the VAM Studio interface. In the top navigation bar, there are tabs for File, Edit, Selection, View, Go, Debug, Terminal, Help, and OPEN EDITORS. The OPEN EDITORS tab is active, displaying two files: 'vanvlefure-01.py' and 'set-data-01.py'. Below the tabs is the Explorer sidebar, which lists various project components like APP, examples, code, ai-as-a-service, basic-opencv, basic-sdk, classification, data-visualization, object-detection, and others. The main area contains the code for 'set-data-01.py' and a terminal window showing command-line logs. The terminal output includes Python code for interacting with the VAM SDK and log messages from the server, such as 'Matched user vamanalytics' and 'Metadata status: Success'.

```

set-data-01.py - app - vam-studio

File Edit Selection View Go Debug Terminal Help
OPEN EDITORS
APP
examples code
ai-as-a-service
basic-opencv
basic-sdk
classification
data-visualization
object-detection
examples image
models
modules
protos
yaml
app.py
counting-01.py
detection-01.py
drawROI-01.py
results-01.jpg
set-data-01.py
snapshot-01.py
test.png
vanvlefure-01.py

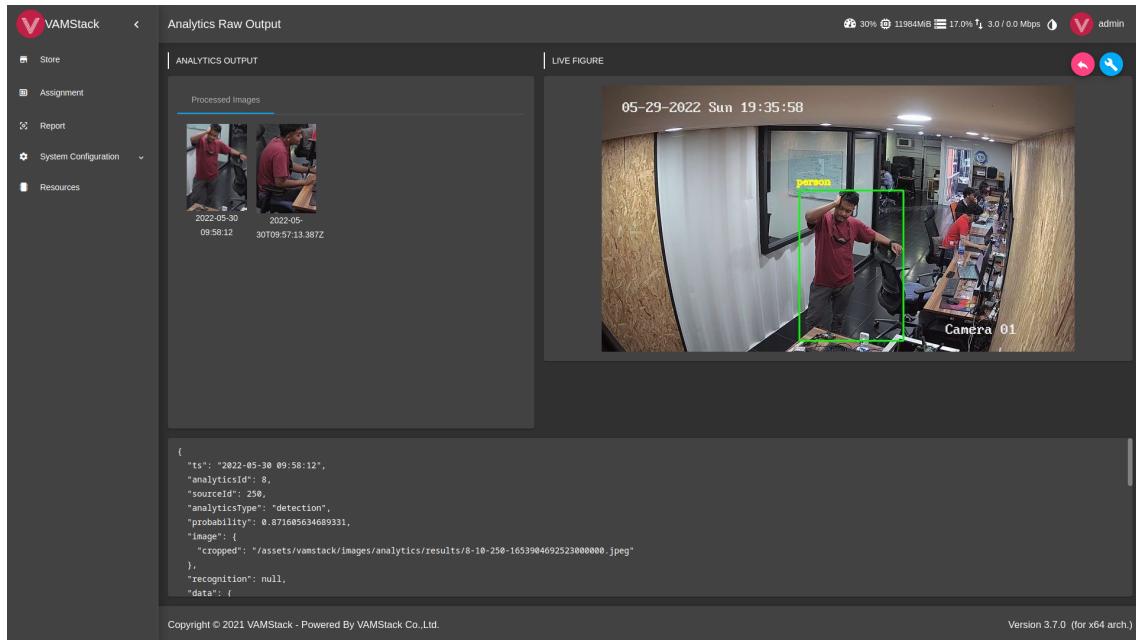
set-data-01.py x counting-01.py

1 # Import vamsdk for send data to platform examples
2 from vam.vamsdk import get_vamsnapshot
3 from vam.vamsdk import people_detection
4 from vam.vamsdk import objects_counting
5 from vam.vamsdk import set_data2vmMetadata
6 import cv2
7 import os
8
9 ANALYTICS_ID = int(os.environ.get("ANALYTICS_ID"))
10
11 # SDK - Platform get camera source
12 (img, assignmentId, sourceId, ts) = get_vamsnapshot()
13 print(ts)
14
15 # SDK - People detection
16 (people_bboxes, results_frame_people) = people_detection(frame=img, conf=0.6)
17
18 # SDK - Counting object with bounding box
19 cnt = objects_counting(people_bboxes)
20 print(cnt)
21
22 # SDK - Send analyzed data to metadata API
23
24 # ***If not have data to input in the argument parameter. Please set None to that argument.
25 set_data2vmMetadata(assignmentId=assignmentId, sourceId=sourceId, ts=ts, image=results_frame_people, bboxes=people_bboxes, objectClass="person", cnt=cnt)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
root@b933f59ec2db:/usr/src/app# python3 set-data-01.py
2022-05-16 14:50:39.497 | DEBUG | modules.analytics.user_authenticator:check_existing_user:56 - Matched user vamanalytics
2022-05-16 14:50:39.498 | INFO | vam.vamsdk:get_vamsnapshot():60 - Get snapshot status: Done
1653908940000000
2022-05-16 14:50:39.816 | DEBUG | modules.analytics.user_authenticator:check_existing_user:56 - Matched user vamanalytics7
2022-05-16 14:50:39.875 | DEBUG | modules.vamServiceAPI:serviceDataInfo:56 - analytics user <modules.analytics.user_authenticator.AnalyticsUser object at 0x7f9fafeacac>
2022-05-16 14:50:39.911 | INFO | modules.vamServiceAPI:metadataSendInfo:136 - Metadata API: 200
2022-05-16 14:50:39.912 | INFO | modules.vamServiceAPI:metadataSendInfo:139 - Metadata API: 200
2022-05-16 14:50:39.913 | INFO | modules.vamServiceAPI:metadataSendInfo:139 - Metadata status: Success
root@b933f59ec2db:/usr/src/app#

```

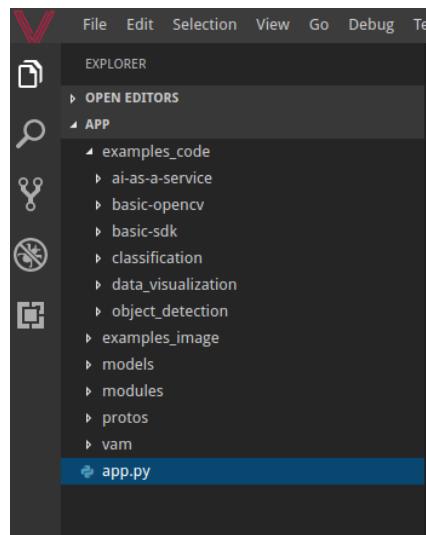
รูปที่ 3.10 ผลการรันตัวอย่างโปรแกรมการส่งข้อมูลผลลัพธ์บน VAM Studio



รูปที่ 3.11 ผลการส่งข้อมูลผลลัพธ์การประมวลผลไปที่ VAM Platform

โจทย์

- ให้ผู้เรียนทำการรัน app.py โดยนำคอมเมนต์ "#" ออกพร้อมทั้งสังเกตการทำงานของโปรแกรมในไฟล์ดังกล่าวว่ามีลำดับการทำงานอย่างไรและบันทึกผล



สรุปและบันทึกผล

LAB 4 | VAM AlaaS

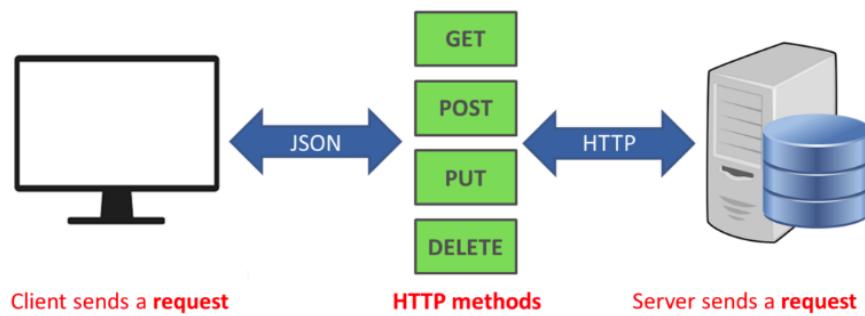
RESTful API

RESTful API เป็นอินเทอร์เฟซที่ระบบคอมพิวเตอร์สองระบบใช้เพื่อแลกเปลี่ยนข้อมูลผ่านอินเทอร์เน็ตได้อย่างปลอดภัย และใช้อ้างถึง Web Service ที่ใช้สถาปัตยกรรม REST

การทำงานของ RESTful API จะอนุญาตให้คิลเอนต์ (Client) ส่งคำขอ (Request) และเข้าถึงข้อมูลหรือทรัพยากร (Resource) บนเซิร์ฟเวอร์ (Server) ด้วยการใช้ API ที่กำหนดให้เซิร์ฟเวอร์ทราบถึงสิ่งที่ต้องทำกับข้อมูลหรือทรัพยากรและส่งผลตอบรับ (Response) กลับมาเป็น Payload ในรูปแบบ HTML, XML, JSON หรือ format อื่น ๆ โดยที่การเต็ตอบทองระบบที่ใช้สถาปัตยกรรม REST จะอยู่บนพื้นฐานของ Hypertext Transfer Protocol (HTTP) ทั้งนี้ การส่งคำขอและผลตอบรับจากเซิร์ฟเวอร์จะแตกต่างกันไป จากการออกแบบของนักพัฒนา

ชุดคำสั่งพื้นฐานของ HTTP มีดังต่อไปนี้

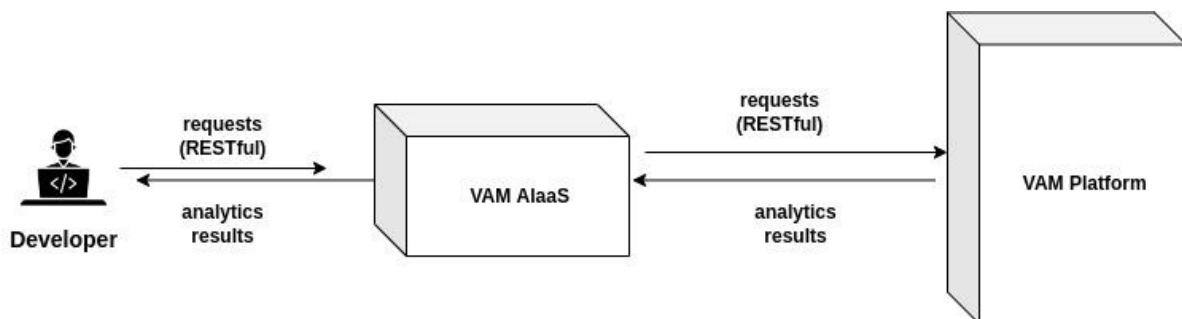
1. GET ใช้สำหรับดึงข้อมูลที่อยู่บนเซิร์ฟเวอร์
2. POST ใช้สำหรับส่งข้อมูลหรือสร้างข้อมูลใหม่บนเซิร์ฟเวอร์
3. PUT ใช้สำหรับอัปเดตหรือแก้ไขข้อมูลที่มีอยู่บนเซิร์ฟเวอร์
4. DELETE ใช้สำหรับลบข้อมูลที่มีอยู่บนเซิร์ฟเวอร์



รูปที่ 4.1 การเชื่อมต่อข้อมูลระหว่าง client และ server ด้วยหลักการ RESTful API

VAM AlaaS

Video Analytics Management AI as a Service (VAM AlaaS) จะทำหน้าที่เป็นตัวกลางการรับ-ส่งข้อมูลระหว่างผู้ใช้และ VAM Platform โดยใช้องค์ความรู้ของ RESTful ทำให้ผู้ใช้สามารถรับส่งข้อมูลและเรียกใช้โมเดลการวิเคราะห์ข้อมูลสำหรับเอาไปใช้ในการพัฒนาต่ออยอดในส่วนของโปรแกรมประมวลผลและวิเคราะห์ข้อมูลได้รวดเร็วมากขึ้น



รูปที่ 4.2 แผนภาพการทำงานของ VAM AlaaS

LAB 4.1 การติดตั้งและประกอบชุดทดลอง VAM Platform สำหรับ AlaaS

วัสดุประสงค์

1. เพื่อก่อตัวถึงอุปกรณ์ที่ใช้ในการการติดตั้งและประกอบชุดการทดลอง
2. เพื่อเรียนรู้วิธีการติดตั้งและประกอบชุดการทดลอง VAM Platform สำหรับ AlaaS

4.1.1 วิธีที่ 1 การติดตั้งและประกอบชุดทดลองเข้ากับอุปกรณ์เสริม

วิธีที่ 1 นี้จะแนะนำวิธีการติดตั้งและประกอบชุดทดลอง VAM Platform สำหรับ AlaaS ให้กับผู้ที่ต้องการทดลองใช้งาน แต่ไม่ต้องการติดตั้งซอฟต์แวร์เพิ่มเติม แต่ต้องมีอุปกรณ์ที่รองรับการเชื่อมต่ออินเทอร์เน็ต เช่น สมาร์ทโฟน แท็บเล็ต หรือคอมพิวเตอร์ที่มีพอร์ต USB หรือ HDMI สำหรับการต่อจอภาพ

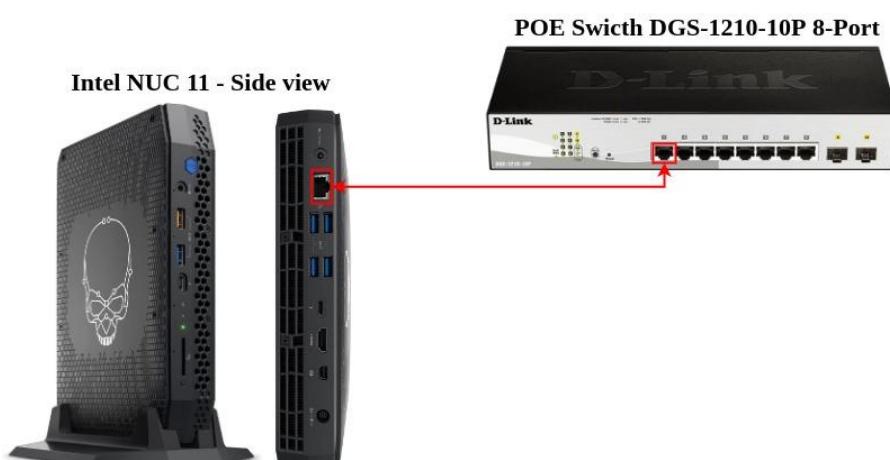
อุปกรณ์ที่เกี่ยวข้อง

1. อุปกรณ์หลัก
 - 1.1. Intel NUC 11 enthusiast mini pc kit 1 เครื่อง
 - 1.2. LITEON AC Adapter 1 ตัว
 - 1.2.1. Input AC 100-240V ~ 50-60Hz 3.5A

1.2.2.	Output 19.5V 11.8A 230.0W		
1.3.	iWave Development Board	1	บอร์ด
1.4.	SINPRO Switching Power supply	1	ตัว
1.4.1.	Input 100-240V ~ 47-63Hz 1.45A		
1.4.2.	Output 12V 5A max		
1.5.	POE Switch: D-Link DGS-1210-10P 8-Port	1	เครื่อง
1.6.	LEADER ELETRONICS Power supply	1	ตัว
1.6.1.	Input 100-240V ~ 50/60Hz 1.2A		
1.6.2.	Output 54.0V 1.67A 90.0W		
1.7.	Hikvision H.265+ Exir Fixed Cube Network Camera	1	ตัว
1.8.	สาย LAN	2	เส้น
2.	อุปกรณ์เสริม		
2.1.	Azus ZenScreen	1	จอ
2.2.	สาย USB Display type-C to Type-C	1	เส้น
2.3.	คีย์บอร์ด	1	ตัว
2.4.	เม้าส์	1	ตัว

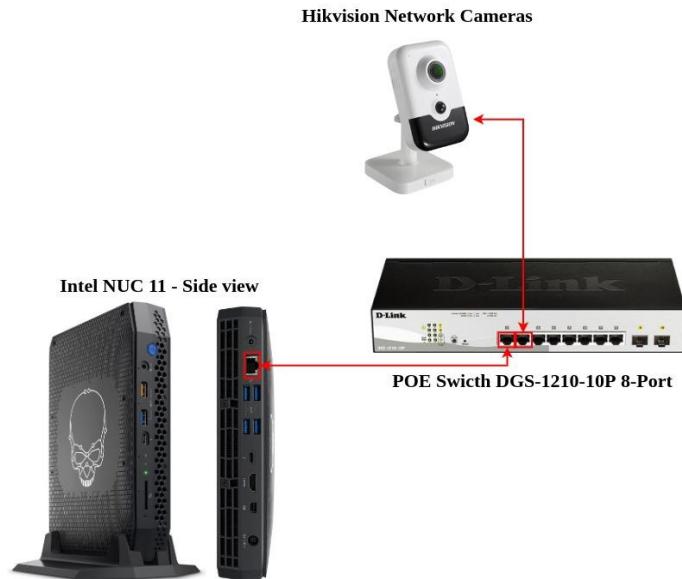
ขั้นตอนการติดตั้งและประกอบชุดการทดลอง

- เชื่อมต่อเครื่อง NUC และ POE Switch ด้วยสาย LAN



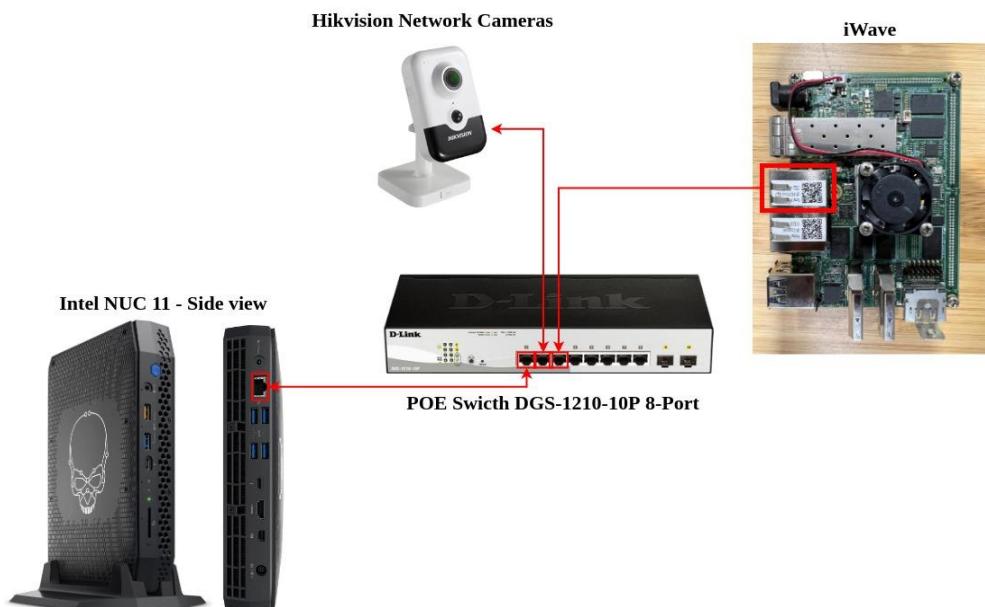
รูปที่ 4.3 การเชื่อมต่อเครื่อง NUC และ POE Switch

- เชื่อมต่อกล้องและ POE Switch ด้วยสาย LAN



รูปที่ 4.4 การเชื่อมต่อกล้องและ POE Switch

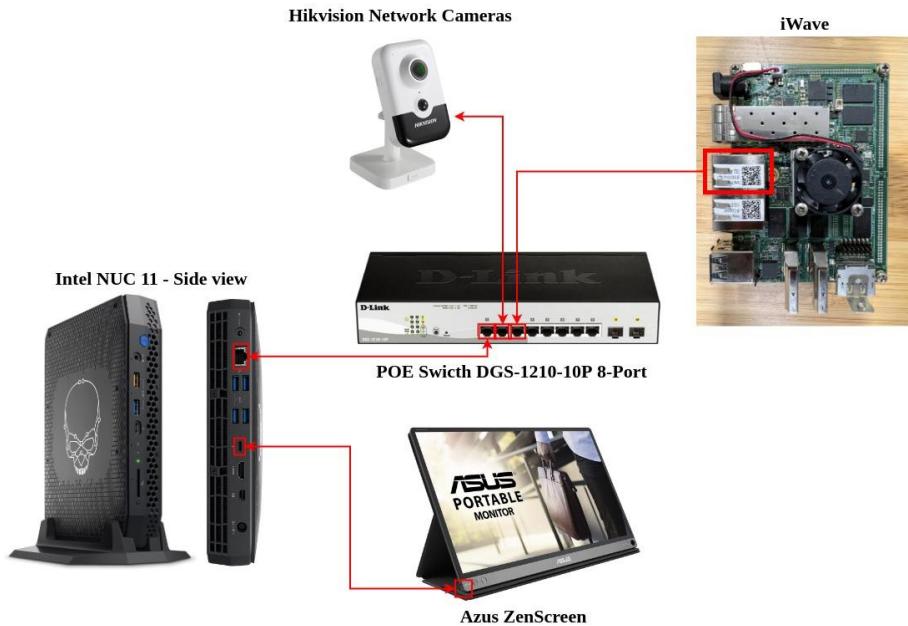
3. เชื่อมต่อบอร์ด iWave และ POE Switch ด้วยสาย LAN



รูปที่ 4.5 การเชื่อมต่อบอร์ด iWave และ POE Switch

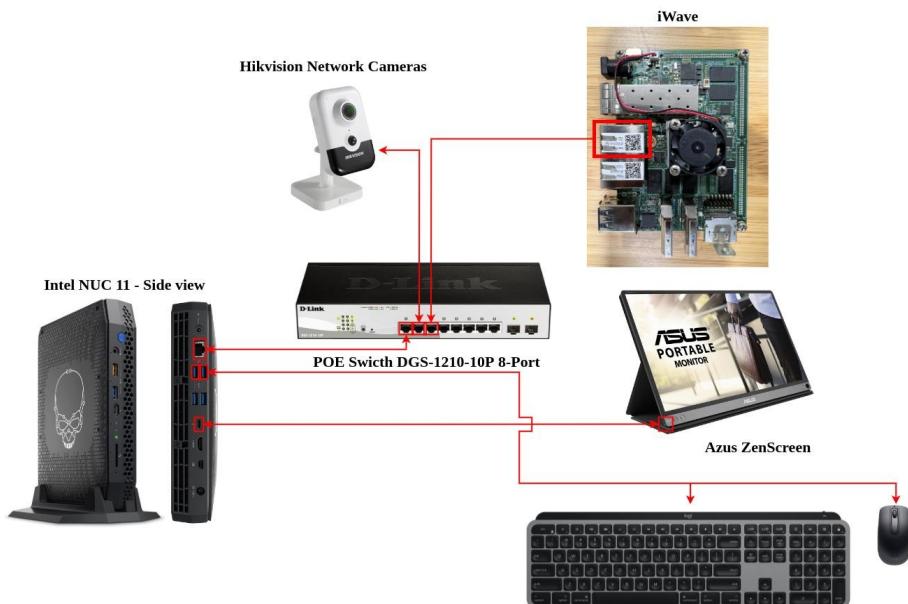
หมายเหตุ ช่องสาย LAN ของบอร์ด iWave บังคับให้ใช้เพียงช่องเดียวตามรูปที่ 4.4

4. เชื่อมต่อเครื่อง NUC และจอ ZenScreen ด้วยสาย USB Display type-C to Type-C



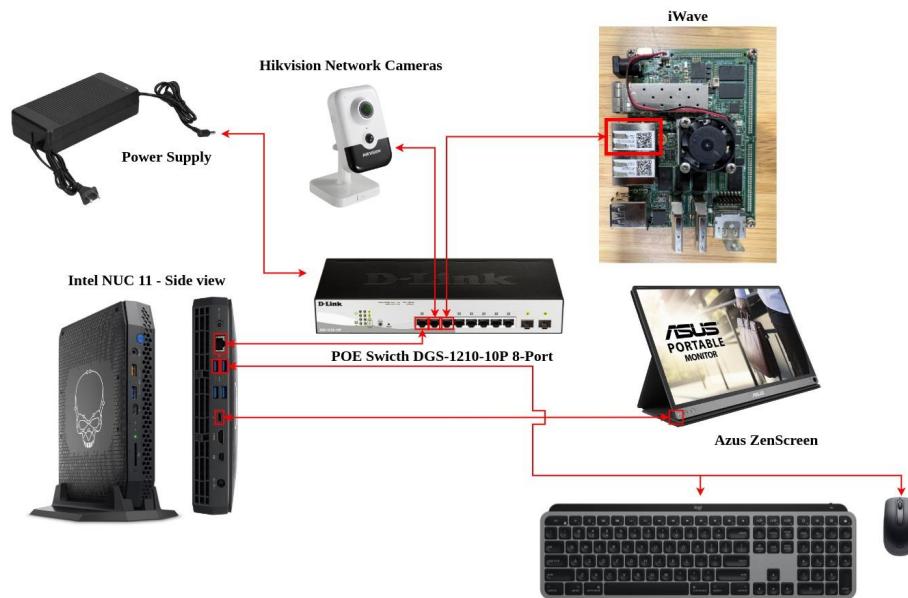
รูปที่ 4.6 การเชื่อมต่อเครื่อง NUC และจอ ZenScreen

5. เชื่อมต่อเครื่อง NUC คีย์บอร์ดและเมาส์



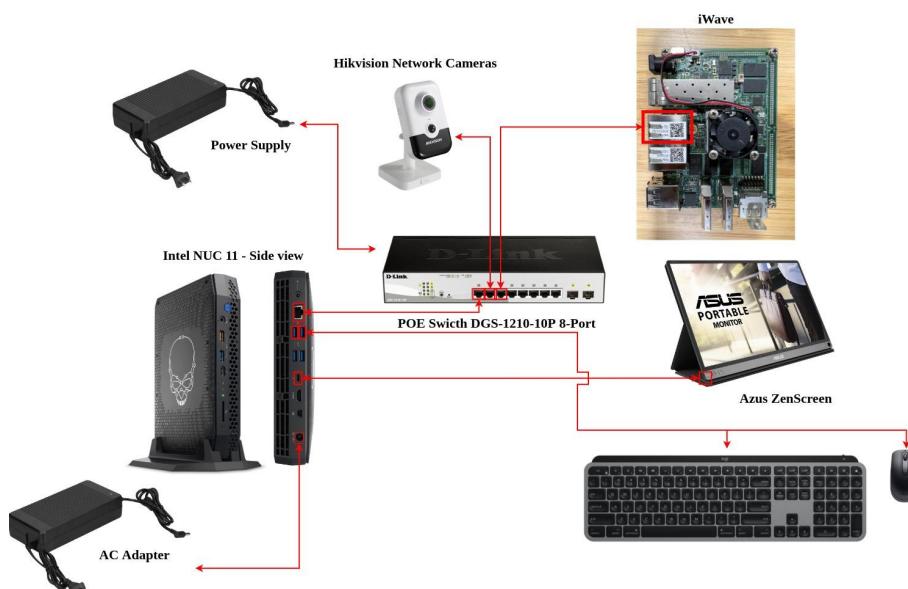
รูปที่ 4.7 การเชื่อมต่อเครื่อง NUC คีย์บอร์ดและเมาส์

6. เชื่อมต่อแหล่งจ่ายไฟจาก LEADER ELETRONICS Power supply เข้า POE Switch



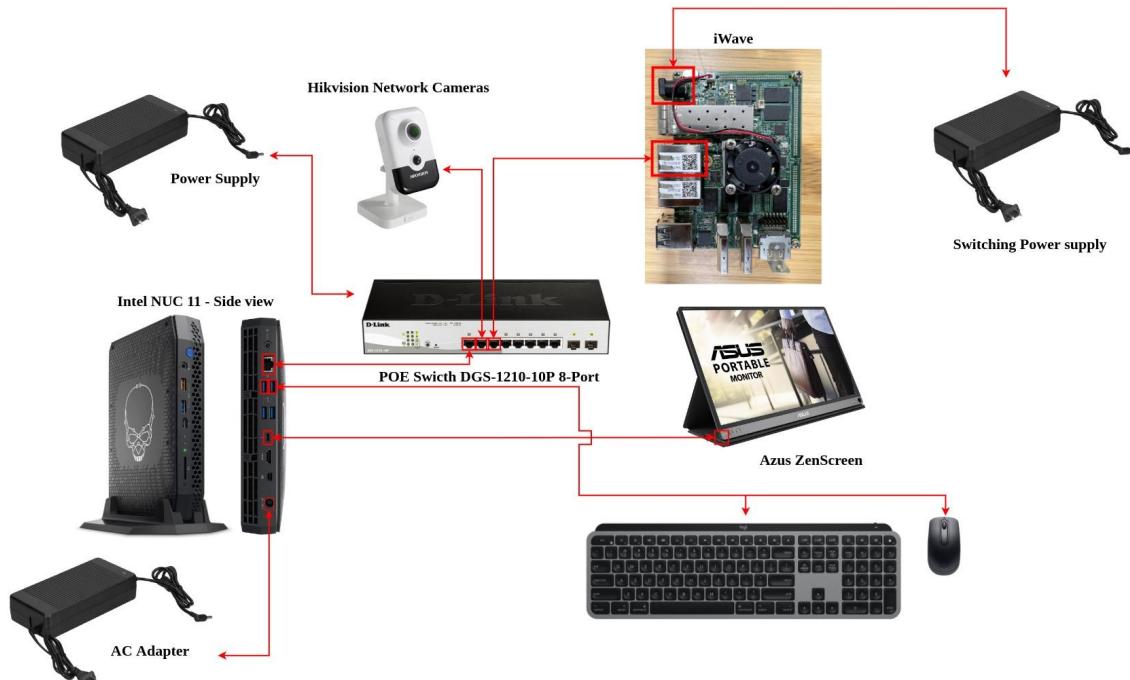
รูปที่ 4.8 การเชื่อมต่อแหล่งจ่ายไฟจาก Power supply เข้า POE Switch

7. เชื่อมต่อแหล่งจ่ายไฟจาก LITEON AC Adapter เข้าเครื่อง NUC



รูปที่ 4.9 การเชื่อมต่อแหล่งจ่ายไฟจาก AC Adapter เข้าเครื่อง NUC

8. เชื่อมต่อแหล่งจ่ายไฟจาก SINPRO Switching Power supply เข้าบอร์ด iWave



รูปที่ 4.10 การเชื่อมต่อแหล่งจ่ายไฟจาก Switching Power supply เข้าบอร์ด iWave

- ตรวจสอบการเชื่อมต่ออุปกรณ์ให้เรียบร้อย แล้วจึงเปิดเครื่อง NCU เปิดจอ ZenScreen และดันสวิตซ์ของบอร์ด iWave ไปที่ ON เพื่อเปิดเครื่อง



รูปที่ 4.11 เปิดเครื่อง NUC จอ ZenScreen และบอร์ด iWave

- เมื่อเปิดเครื่อง NUC ให้ผู้เรียนทำการกรอกรหัส “eslab” เพื่อเข้าสู่ระบบของเครื่อง NUC

4.1.2 วิธีที่ 2 การติดตั้งและประกอบชุดทดลองเข้ากับเครื่องของผู้เรียน

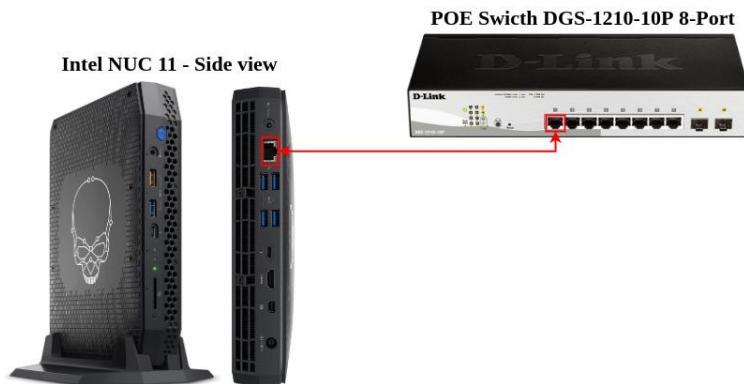
วิธีนี้จะมีขั้นตอนการติดตั้งคล้ายกับวิธีที่ 1 แตกต่างกันที่เครื่องประมวลผลไม่ต้องต่อเข้ากับอุปกรณ์เสริม แต่จะเชื่อมต่อเข้ากับคอมพิวเตอร์ของผู้เรียน ทำให้ผู้เรียนสามารถใช้งานแพลตฟอร์มบนเครื่องของผู้เรียนได้โดยเครื่องพร้อมกัน ซึ่งวิธีการติดตั้งจะคล้ายกับ LAB 1.1.2 แต่จะเพิ่มและปรับลำดับขั้นตอนที่ 3 และ 8 สำหรับการเชื่อมต่อบอร์ด iWave ที่ใช้สำหรับประมวลผลไมเดลในหน้า

อุปกรณ์ที่เกี่ยวข้อง

1. อุปกรณ์หลัก			
1.1. Intel NUC 11 enthusiast mini pc kit	1	เครื่อง	
1.2. LITEON AC Adapter	1	ตัว	
1.2.1. Input AC 100-240V ~ 50-60Hz 3.5A			
1.2.2. Output 19.5V 11.8A 230.0W			
1.3. iWave Development Board	1	บอร์ด	
1.4. SINPRO Switching Power supply	1	ตัว	
1.4.1. Input 100-240V ~ 47-63Hz 1.45A			
1.4.2. Output 12V 5A max			
1.5. POE Switch: D-Link DGS-1210-10P 8-Port	1	เครื่อง	
1.6. LEADER ELETRONICS Power supply	1	ตัว	
1.6.1. Input 100-240V ~ 50/60Hz 1.2A			
1.6.2. Output 54.0V 1.67A 90.0W			
1.7. สาย LAN			
1.7.1. สำหรับต่อเข้าอุปกรณ์หลัก	2	เส้น	
1.7.2. สำหรับต่อเข้าคอมพิวเตอร์ของผู้เรียน	1	เส้น/เครื่อง	
2. เครื่องคอมพิวเตอร์ของผู้เรียน			
2.1. ระบบปฏิบัติการ Windows 10 ขึ้นไป			
2.2. Ubuntu 18.04 หรือ 20.04			
2.3. MacOS			

ขั้นตอนการติดตั้งและประกอบชุดการทดลอง

1. เชื่อมต่อเครื่อง NUC และ POE Switch ด้วยสาย LAN



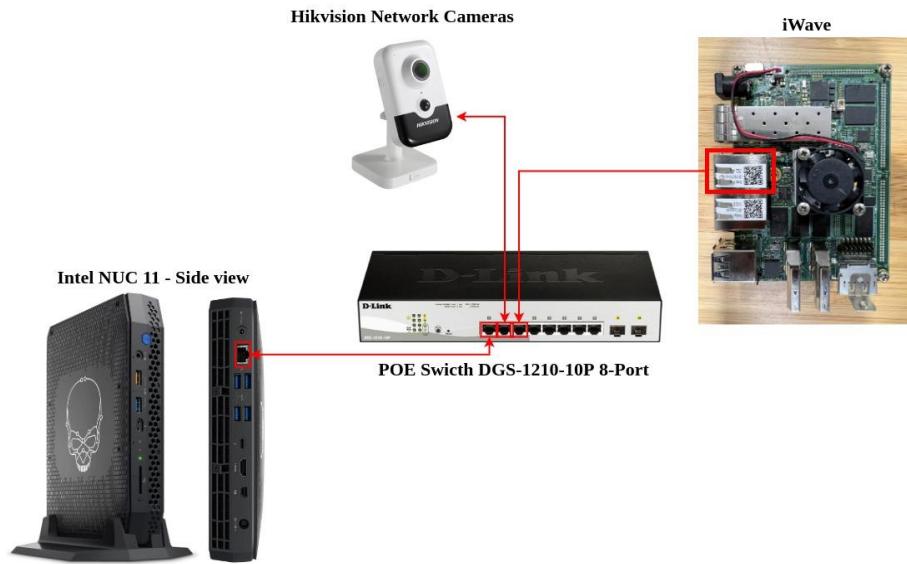
รูปที่ 4.12 การเชื่อมต่อเครื่อง NUC และ POE Switch

2. เชื่อมต่อกล้องและ POE Switch ด้วยสาย LAN



รูปที่ 4.13 การเชื่อมต่อกล้องและ POE Switch

3. เชื่อมต่อบอร์ด iWave และ POE Switch ด้วยสาย LAN

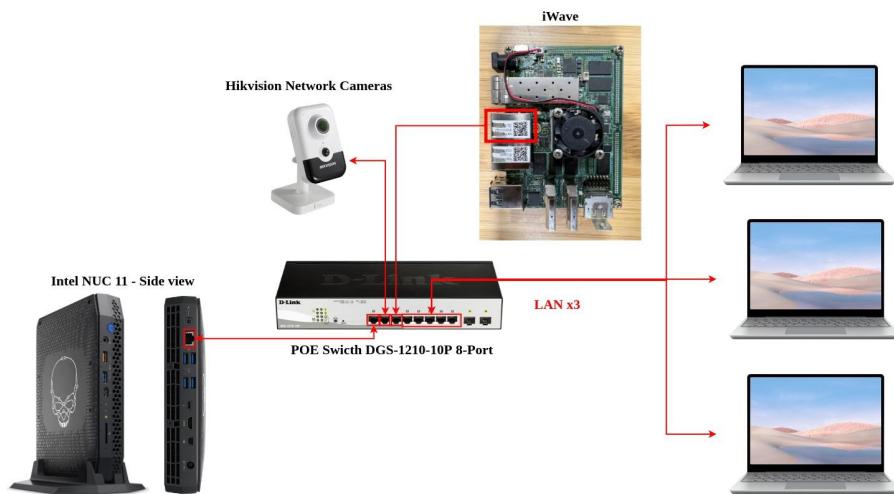


รูปที่ 4.14 การเชื่อมต่อบอร์ด iWave และ POE Switch

หมายเหตุ

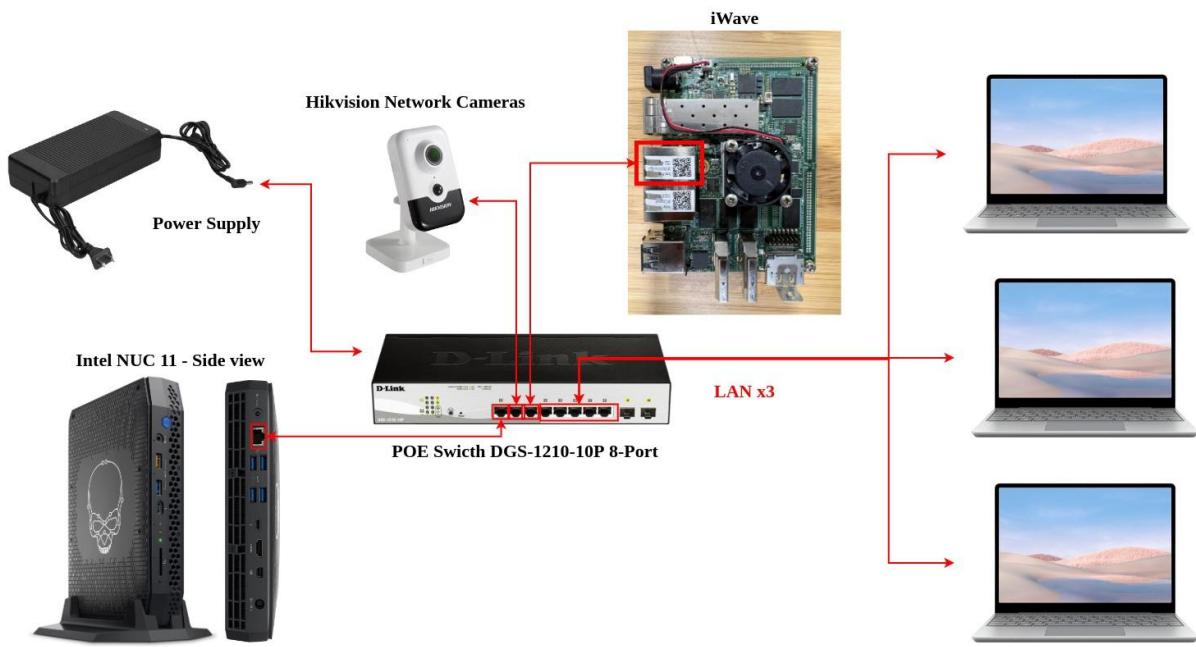
ช่องสาย LAN ของบอร์ด iWave บังคับให้ใช้เพียงช่องเดียวตามรูปที่ 4.14

4. เชื่อมต่อคอมพิวเตอร์ของผู้เรียนและ POE Switch ด้วยสาย LAN



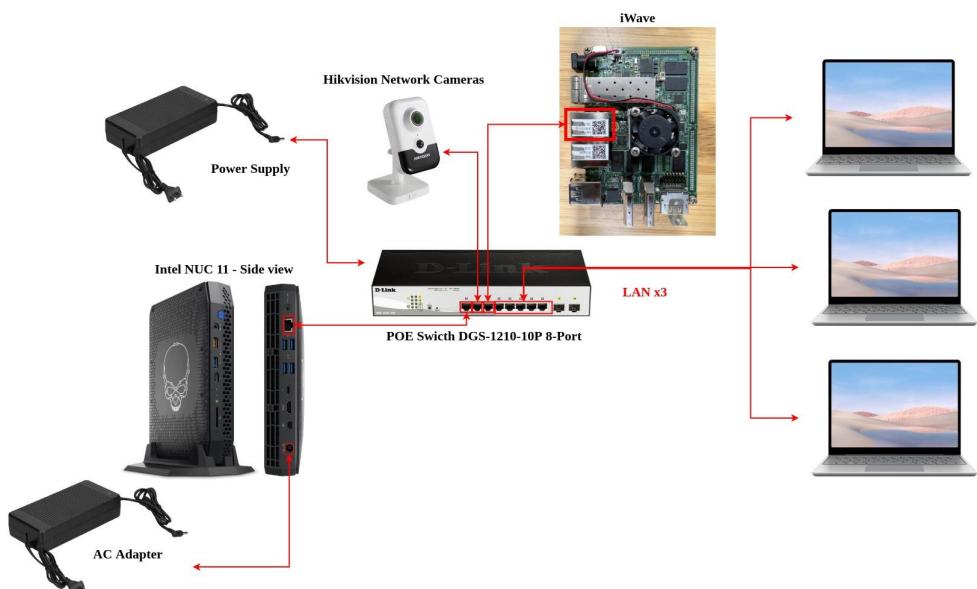
รูปที่ 4.15 การเชื่อมต่อแหล่งจ่ายไฟจาก AC Adapter เข้า POE Switch

5. เชื่อมต่อแหล่งจ่ายไฟจาก LEADER ELETRONICS Power supply เข้า POE Switch



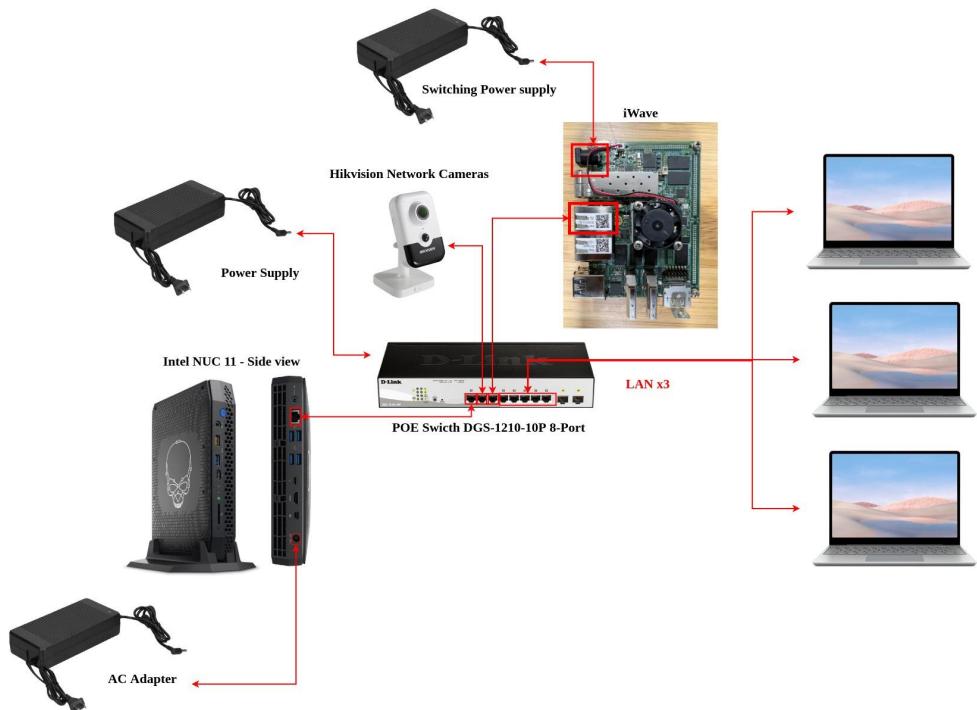
รูปที่ 4.16 การเชื่อมต่อแหล่งจ่ายไฟจาก Power supply เข้า POE Switch

6. เชื่อมต่อแหล่งจ่ายไฟจาก LITEON AC Adapter เข้าเครื่อง NUC



รูปที่ 4.17 การเชื่อมต่อแหล่งจ่ายไฟจาก AC Adapter เข้าเครื่อง NUC

7. เชื่อมต่อแหล่งจ่ายไฟจาก SINPRO Switching Power supply เข้าบอร์ด iWave



รูปที่ 4.18 การเชื่อมต่อแหล่งจ่ายไฟจาก Switching Power supply เข้าบอร์ด iWave

8. ตรวจสอบการเชื่อมต่ออุปกรณ์ให้เรียบร้อย และวิ่งเปิดเครื่อง NCU และดันสวิตช์ของบอร์ด iWave ไปที่ ON เพื่อเปิดเครื่อง



รูปที่ 4.19 ปุ่มเปิดเครื่อง NUC และปุ่มเปิดจอ ZenScreen

9. ตั้งค่าเครือข่ายอินเทอร์เน็ตภายในเครื่องคอมพิวเตอร์ของผู้เรียน ([หัวข้อ 1.1.2](#))

LAB 4.2 โปรแกรมการลงทะเบียนภาพใบหน้า (Face Registration)

วัตถุประสงค์

1. เพื่อเรียนรู้วิธีการใช้เซอร์วิสสำหรับการลงทะเบียนภาพใบหน้า
2. เพื่อศึกษาหลักการส่งข้อมูลผ่าน url

อุปกรณ์ที่เกี่ยวข้อง

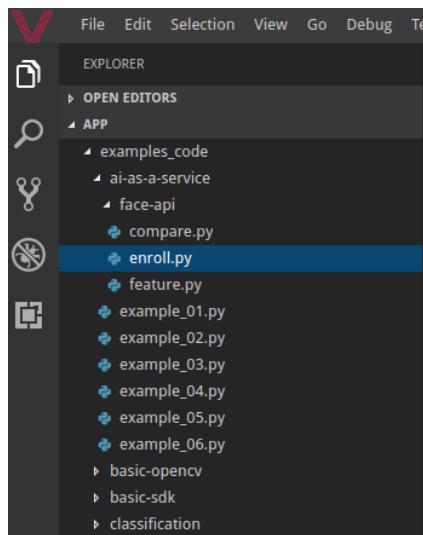
1. ชุดการทดลอง VAM Platform สำหรับ AlaaS

4.2.1 EnrollFace

EnrollFace คือ เซอร์วิสสำหรับใช้ในการลงทะเบียนภาพใบหน้าเข้าสู่ฐานข้อมูล โดยจะทำการรับข้อมูลภาพใบหน้ามาแปลงเป็นข้อมูลไบต์ (bytes) และรับหมายเลขลำดับการลงทะเบียนจากผู้ใช้ส่งไปยังฐานข้อมูล หลังจากทำการส่งข้อมูล เซอร์วิสจะส่งเลขสถานะการทำงานและข้อความจากทำงานกลับมาให้ผู้ใช้

ขั้นตอนการใช้ตัวอย่างโปรแกรมการลงทะเบียนใบหน้า

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > ai-as-a-service > face-api > enroll.py ที่แถบเมนูด้านซ้าย



รูปที่ 4.20 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการลงทะเบียนใบหน้า

ตัวอย่างโปรแกรมการลงทะเบียนใบหน้า

```
# import necessary library
import requests
import pprint
import json

# parameters for calling ai service
HOST_IP = "10.1.1.212"
PORT = "8500"
IMAGE_PATH = "../../examples_image/face02.jpg"
endpoint = "/EnrollFace"
memberId = "<insert member_id here>" #insert member id

# imagepath
request_form = {"Image": open(IMAGE_PATH, "rb")}

# member id data
member_data = {"MemberId": memberId}

# calling ai service function
response = requests.post(f'http://{{HOST_IP}}:{{PORT}}{{endpoint}}',
files=request_form, data=member_data)

# response and results in json format
```

```
print(response.status_code)
pprint.pprint(response.json())
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์เดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/ai-as-a-service/face-api
```

4. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 enroll.py
```

The screenshot shows the VAM Studio interface. On the left, the file tree displays the project structure with files like 'enroll.py', 'feature.py', and various 'example' files. The main area shows the content of 'enroll.py':

```

# import necessary library
import requests
import pprint
import json

# parameters for calling ai service
HOST_IP = "192.168.1.1.212"
PORT = "8880"
IMAGE_PATH = "../../../../examples/image/face02.jpg"
endpoint = "/EnrollFace"
memberId = "test-face" #insert member id

# imagepath
request_form = {
    "image": open(IMAGE_PATH, "rb")
}

# member id data
member_data = {
    "MemberID": memberId
}

# calling ai service function
response = requests.post("http://{}:{}{}".format(HOST_IP, PORT, endpoint), files=request_form, data=member_data)

# response and results in json format
print(response.status_code)
pprint.pprint(response.json())

```

The terminal window at the bottom shows the command being run and its output:

```

root@febd0ad82b42d:/usr/src/app/examples_code# ls
ai-as-a-service basic-opency basic-sdk classification data_visualization object_detection
root@febd0ad82b42d:/usr/src/app/examples_code# cd ai-as-a-service/
root@febd0ad82b42d:/usr/src/app/examples_code/ai-as-a-service# ./face-api/python3 enroll.py
root@febd0ad82b42d:/usr/src/app/examples_code/ai-as-a-service/face-api# python3 enroll.py
2021-07-20 10:45:43,646 INFO     [__main__.enroll] EnrollFaceSuccessful: success=True
{'message': 'EnrollFaceSuccessful', 'success': True}
root@febd0ad82b42d:/usr/src/app/examples_code/ai-as-a-service/face-api#

```

รูปที่ 4.21 ผลการรันตัวอย่างโปรแกรมการลงทะเบียนใบหน้าบน VAM Studio

ผลลัพธ์

200
 {'message': 'EnrollFaceSuccessful', 'success': True}

LAB 4.3 โปรแกรมการจดจำใบหน้า (Face Recognition)

วัตถุประสงค์

1. เพื่อเรียนรู้วิธีการใช้เซอร์วิสสำหรับหาคุณลักษณะบนใบหน้า
2. เพื่อศึกษาหลักการส่งข้อมูลผ่าน url

อุปกรณ์ที่เกี่ยวข้อง

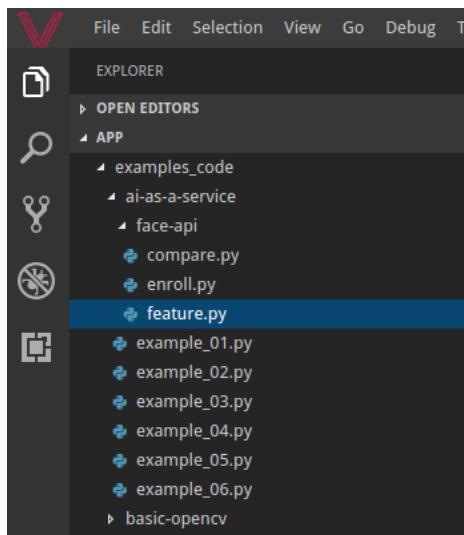
1. ชุดการทดลอง VAM Platform สำหรับ AlaaS

4.3.1 FaceFeature

FaceFeature คือ เซอร์วิสสำหรับใช้ในการหาคุณลักษณะบนใบหน้า โดยจะทำการรับข้อมูลภาพใบหน้ามาแปลงเป็นข้อมูลไบต์ (bytes) และส่งไปประมวลผลและวิเคราะห์ข้อมูลเพื่อหาคุณลักษณะบนใบหน้า หลังจากเสร็จสิ้นการประมวลผล เซอร์วิสจะส่งเลขสถานะการทำงานและข้อมูล Json ที่ได้จากการประมวลผลและวิเคราะห์ข้อมูลกลับมาให้ผู้ใช้

ขั้นตอนการใช้ตัวอย่างโปรแกรมการจดจำใบหน้า

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > ai-as-a-service > face-api > feature.py ที่แถบเมนูด้านซ้าย



รูปที่ 4.22 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการหาคุณลักษณะของใบหน้า

ตัวอย่างโปรแกรมการหาคุณลักษณะของใบหน้า

```
# import necessary library
import requests
import pprint
import json

# parameters for calling ai service
HOST_IP = "10.1.1.212"
PORT = "8500"
IMAGE_PATH = "../../examples_image/face-compare01.jpg"
endpoint = "/FaceFeature"

# imagepath
request_form = {"Image": open(IMAGE_PATH, "rb")}

# calling ai service function
response = requests.post(f'http://{HOST_IP}:{PORT}{endpoint}',
files=request_form)

# response and results in json format
print(response.status_code)
pprint.pprint(response.json())
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงโผลเดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/ai-as-a-service/face-api
```

4. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 feature.py
```

The screenshot shows the VAM Studio interface. The code editor displays the `feature.py` file, which contains Python code for calling an AI service to extract face features from an image. The terminal window below shows the command `python3 feature.py` being run, and it displays a long list of numerical values representing the extracted face features.

```

feature.py x
File Edit Selection View Go Debug Terminal Help
OPEN EDITORS feature.py example_code/ai-as-a-service...
APP
example_code
  - ai-as-service
    - face-api
      - compare.py
      - enroll.py
      - feature.py
    - example_01.py
    - example_02.py
    - example_03.py
    - example_04.py
    - example_05.py
    - example_06.py
  - basic-spency
  - basic-sdk
  - classification
  - face-detection
  - object-detection
  - example_image
  - models
  - modules
  - protos
  - vam
  - app.py
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
feature.py - app - vam-studio
feature.py x
1 # import necessary library
2 import requests
3 import json
4 import json
5
6 # parameters for calling ai service
7 HOST_IP = "10.1.1.212"
8 PORT = "8500"
9 IMAGE_PATH = "../../../../examples_image/face-compare01.jpg"
10 endpoint = "/FaceFeature"
11
12 # imagepath
13 request_form = {
14     "image": open(IMAGE_PATH, "rb")
15 }
16
0.928125,
-0.265625,
-0.57125,
-0.59375,
-1.0,
-0.46875,
0.03125,
-0.59375,
-0.46875,
-0.875,
0.375,
0.15625,
1.15625,
0.25,
0.53125,
-0.015625,
-0.34375,
0.75,
-1.21875,
0.03125,
0.609375,
-1.15625,
-0.375,
-1.0625,
-0.640625,
1.0,
-1.140625,
0.609375,
-0.40625,
-0.390625,
0.4375,
-0.5625,
-0.05625,
'quality5p': {'p1': {'x': 74.1333392822266, 'y': 175.72499084472656},
p2: {'x': 208.5, 'y': 181.0500030517578},
p3: {'x': 139.0, 'y': 276.8999938964844},
p4: {'x': 83.4000015258789, 'y': 335.4750061035156},
p5: {'x': 185.3334350585938, 'y': 346.125},
p6: {'x': 47.231218218809401, 'y': 188.0625},
p7: {'x': 102.0, 'y': 256.0}},
Ln 1, Col 1 Spaces: 2 UTF-8 LF Python
root@febad0d2b42:/user/app/example_code/ai-as-a-service/face-api

```

รูปที่ 4.23 ผลการรันตัวอย่างโปรแกรมการหาคุณลักษณะของใบหน้าบน VAM Studio

ผลลัพธ์

```
200
{'face_feature': [1.984375,
                  -0.3125,
                  ... ,
                  -0.5625,
                  -0.65625],
 'quality5p': {'p1': {'x': 74.13333892822266, 'y': 175.72499084472656},
                'p2': {'x': 208.5, 'y': 181.0500030517578},
                'p3': {'x': 139.0, 'y': 276.8999938964844},
                'p4': {'x': 83.4000015258789, 'y': 335.4750061035156},
```

```
'p5': {'x': 185.33334350585938, 'y': 346.125},  
'quality': 0.7231218218803406}}
```

LAB 4.4 โปรแกรมการตรวจจับใบหน้าและการตรวจสอบคุณลักษณะ (Face Detection and Quality Verification)

วัตถุประสงค์

1. เพื่อเรียนรู้วิธีการใช้ชอร์ติวิสการลงทะเบียนภาพใบหน้า
2. เพื่อศึกษาหลักการส่งข้อมูลผ่าน url

อุปกรณ์ที่เกี่ยวข้อง

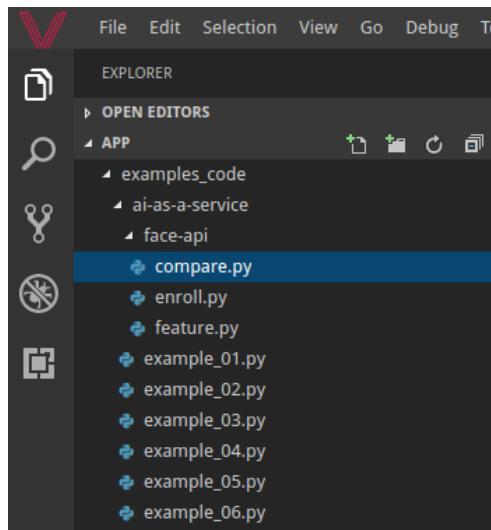
1. ชุดการทดลอง VAM Platform สำหรับ AlaaS

4.4.1 CompareFace

เซอร์วิสสำหรับใช้ในการเปรียบเทียบความคล้ายคลึงระหว่าง 2 ใบหน้า โดยจะทำการรับข้อมูลภาพใบหน้าทั้ง 2 ภาพมาแปลงเป็นข้อมูลไบต์ (bytes) และส่งไปประมวลผลและวิเคราะห์ข้อมูลหาคุณลักษณะบนใบหน้าของทั้ง 2 ภาพมาเทียบความคล้ายคลึงกัน หลังจากเสร็จสิ้นการประมวลผล เซอร์วิสจะส่งผลสถานะการทำงานและค่าหมายความคล้ายคลึงระหว่าง 2 ใบหน้ามาให้ผู้ใช้

ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับใบหน้าและการตรวจสอบคุณลักษณะ

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > ai-as-a-service > face-api > compare.py ที่ແນບເມນູດ້ານໜ້າຍ



รูปที่ 4.24 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการเปรียบเทียบความคล้ายคลึงของใบหน้า

ตัวอย่างโปรแกรมการเปรียบเทียบความคล้ายคลึงของใบหน้า

```
# import necessary library
import requests
import pprint
import json

# parameters for calling ai service
HOST_IP = "10.1.1.212"
PORT = "8500"
image01Path = "../ ../../examples_image/face-compare01.jpg"
image02Path = "../ ../../examples_image/face-compare01.jpg"
endpoint = "/CompareFace"

request_form = {
    "ImageA": open(image01Path, "rb"),
    "ImageB": open(image02Path, "rb")
}

# calling ai service function
response = requests.post(f'http://{HOST_IP}:{PORT}{endpoint}',
files=request_form)

# response and results in json format
print(response.status_code)
pprint.pprint(response.json())
```

- ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์เดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/ai-as-a-service/face-api
```

- ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 compare.py
```

```

File Edit Selection View Go Debug Terminal Help
compare.py - app - vam-studio
OPEN EDITORS
EXPLORER
APP
examples_code
ai-as-a-service
face-api
compare.py
image0.jpg
feature.py
example_01.py
example_02.py
example_03.py
example_04.py
example_05.py
example_06.py
basic_ai
basic_cv
classification
data_visualization
object_detection
examples_image
image0.jpg
modules
protos
vam
app.py

compare.py
1 # import necessary library
2 import requests
3 import pprint
4 import json
5
6 # parameters for calling ai service
7 HOST_IP = "10.1.1.212"
8 PORT = 5000
9 image0Path = "./././examples_images/face-compared0.jpg"
10 image0Path = "./././examples_images/face-compared0.jpg"
11 endpoint = "/CompareFace"
12
13 request_form = {
14     "ImageA": open(image0Path, "rb"),
15     "ImageB": open(image0Path, "rb")
16 }
17
18 # calling ai service function
19 response = requests.post("http://{}:{}{}".format(HOST_IP, PORT, endpoint), files=request_form)
20
21 # response and results in json format
22 print(response.status_code)
23 pprint.pprint(response.json())

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```

root@fe80:0:0:2b24%2d:/user/src/app# cd examples_code/ai-as-a-service/face-api/
root@fe80:0:0:2b24%2d:/user/src/app/examples_code/ai-as-a-service/face-api# python3 compare.py
{'score': 99.9989999845351, 'success': True}
root@fe80:0:0:2b24%2d:/user/src/app/examples_code/ai-as-a-service/face-api#

```

รูปที่ 4.25 ผลการรันตัวอย่างโปรแกรมการเปรียบเทียบความคล้ายคลึงของใบหน้าบน VAM Studio



โจทย์

- ให้ผู้เรียนปรับแก้โปรแกรมการตรวจภาพใบหน้าและการตรวจสอบคุณลักษณะ (Face Detection and Quality Verification) โดยใช้ภาพใบหน้าของผู้เรียนและเพื่อนอีก 1 คนที่ได้จากกล้อง IP camera มาลงทะเบียน

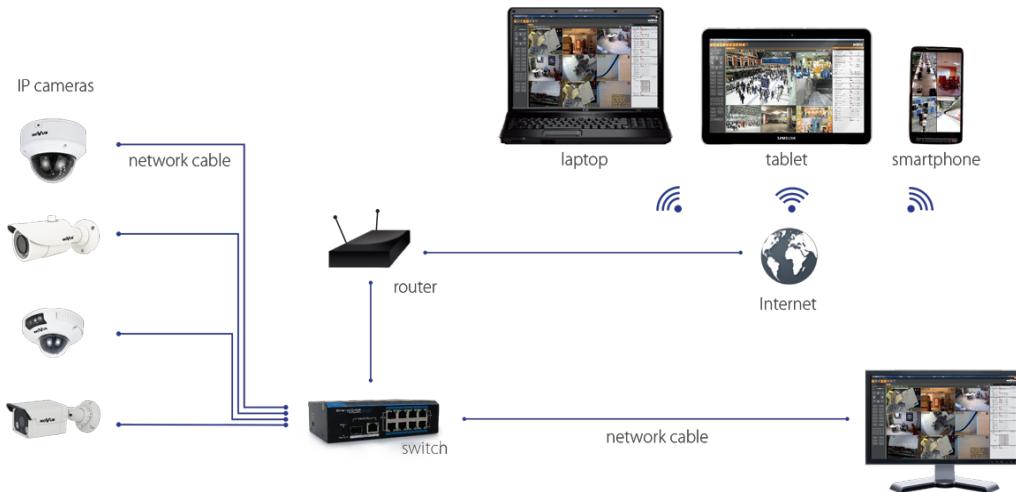
สรุปและบันทึกผล

.....
.....
.....
.....
.....
.....
.....
.....

LAB 5 | VAM Security & Surveillance

IP Camera Network Architecture

โครงสร้างสถาปัตยกรรมการเชื่อมต่อกล้อง CCTV IP Camera ที่ใช้ในระบบป้องกันและระวังภัย จะมีการเชื่อมต่อเข้ากับวงแลนทั้งแบบออนไลน์และออฟไลน์ โดยการเชื่อมต่อ IP Camera เข้ากับตัว switch ด้วยสายแลนเพื่อทำให้มีไฟจ่ายไปที่ตัวกล้อง ต่อมานำเครื่องประมวลผลเข้ามาติดตั้งโดยเสียบสายแลนเข้ากับตัว switch เดียวกันและเซตงแลนให้เป็นวงเดียวกับตัวกล้องในกรณีแบบออฟไลน์ ส่วนในกรณีออนไลน์ให้ผู้เรียนสามารถต่อ router internet เข้ามาใน switch และเชื่อมแลนตัว router ให้เป็นวงเดียว กับกล้องและเครื่องประมวลผล เป็นต้น



รูปที่ 5.1 แผนภาพการทำงานของระบบป้องกันและระวังภัยโดยการใช้ IP Camera

(ที่มา: www.novuscctv.com/en/world-ip/)

VAM Security & Surveillance

VAM Security & Surveillance คือ เป็น VAM AlaaS สำหรับใช้งานในการประมวลผลและวิเคราะห์ข้อมูลที่จะเข้ามาช่วยเหลือและดูแลในส่วนของการป้องกันและระวังภัย เช่น การตรวจจับบุคคล การตรวจจับผู้บุกรุก และการตรวจจับจำนวนบุคคลเข้า-ออกในพื้นที่ที่กำหนด เป็นต้น

LAB 5.1 โปรแกรมตรวจจับผู้บุกรุก (Intrusion Detection)

วัตถุประสงค์

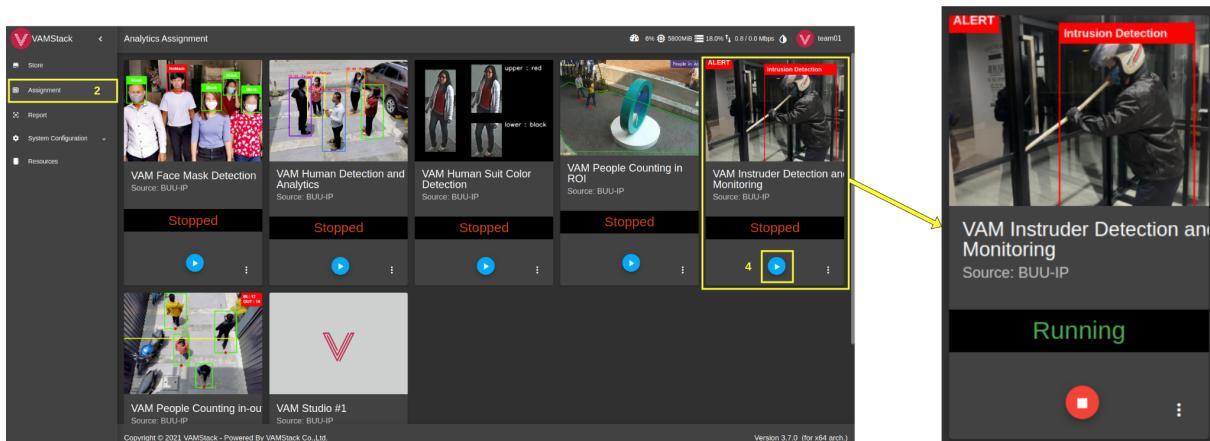
- เพื่อเรียนรู้วิธีการใช้โปรแกรมการตรวจจับผู้บุกรุกบน VAM Platform
- เพื่อเรียนรู้วิธีการใช้ตัวอย่างโปรแกรมการตรวจจับผู้บุกรุกบน VAM Studio

อุปกรณ์ที่เกี่ยวข้อง

- ชุดการทดลอง VAM Platform

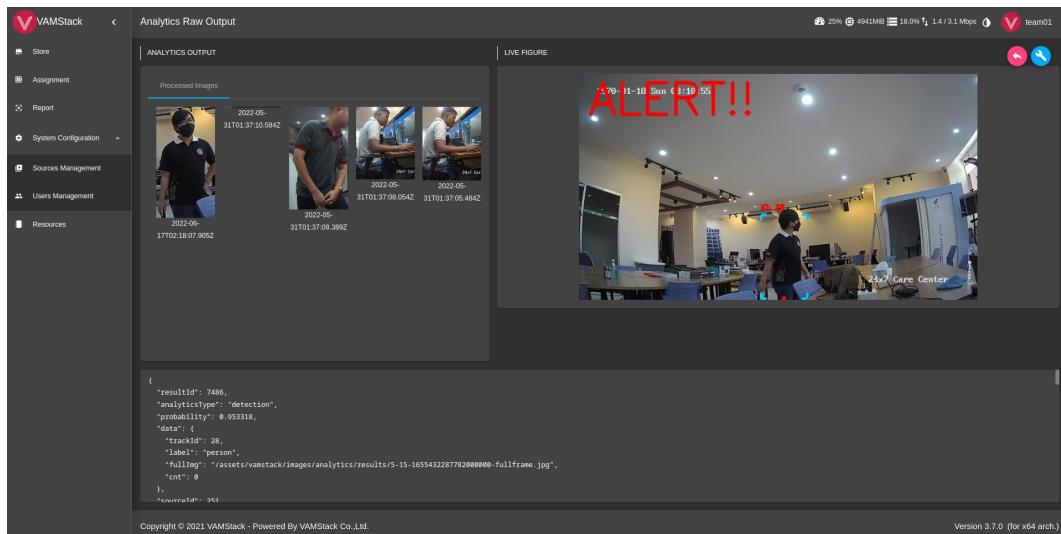
5.1.1 ขั้นตอนการใช้โปรแกรมการตรวจจับผู้บุกรุกบน VAM Platform

- เข้าสู่หน้า VAM Platform โดยพิมพ์ “[10.1.1.212](#)” บนเว็บเบราว์เซอร์ และเข้าสู่ระบบของแพลตฟอร์ม (LAB 1.2 หัวข้อ [1.2.1](#))
- คลิก “Assignment” ที่แถบเมนูด้านซ้าย
- หากต้องการตรวจจับผู้บุกรุกในพื้นที่ที่ต้องการ ผู้ใช้สามารถตั้งค่าพื้นที่ที่สนใจได้ (LAB 1.4 หัวข้อ [1.4.4](#))
- คลิกปุ่มเปลี่ยนสถานะของ “VAM Intruder Detection” ให้เป็น “Running”



รูปที่ 5.2 ลำดับการเข้าสู่โปรแกรมการตรวจจับผู้บุกรุกบน VAM Platform

5. คลิกที่การ์ด “VAM Intrusion Detection” เพื่อดูการประมวลผลและวิเคราะห์ข้อมูล



รูปที่ 5.3 ผลการรันโปรแกรมการตรวจจับผู้บุกรุกบน VAM Platform

5.1.2 intrusion-detection

intrusion-detection คือ เซอร์วิสสำหรับใช้ตรวจสอบผู้บุกรุก โดยจะทำการดึงข้อมูลที่ถูกประมวลผลและวิเคราะห์ข้อมูลมา จาก VAM Platform ที่ถูกจัดเก็บไว้ในฐานข้อมูลมาแสดงผลในรูปแบบของ json บน VAM Studio ซึ่งผู้เรียนสามารถกำหนดจำนวนข้อมูลที่ต้องการรับได้ กรณีที่ไม่มีข้อมูลถูกส่งกลับมา ผู้เรียนจะต้องทำการตัวชี้วัด 5.1.1 เพื่อทำการสร้างข้อมูลวิเคราะห์และประมวลผลขึ้น

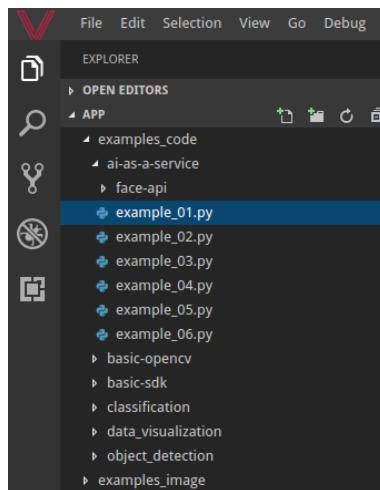
ค่าที่เกี่ยวข้อง

- machine_ip: หมายเลข IP Address ของเครื่องประมวลผลที่ติดตั้ง VAM Platform
- limit_data: ค่ากำหนดจำนวนกลุ่มข้อมูล Json ที่เซอร์วิสทำการดึงมาแสดงผล

ตัวอย่าง กำหนดค่า limit_data คือ 1 หมายความว่า ข้อมูลที่ถูกดึงมาแสดงผลจะมีเพียง 1 ชุดข้อมูลเท่านั้น

ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับผู้บุกรุกบน VAM Studio

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > ai-as-a-service > example_01.py ที่แถบเมนูด้านซ้าย



รูปที่ 5.4 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการตรวจจับผู้บุกรุก

ตัวอย่างโปรแกรมการตรวจจับผู้บุกรุก

```
# import necessary library
import requests
import pprint
import json

# parameters for calling ai service
HOST_IP = "10.1.1.212"
PORT = "20000"
endpoint = "/intrusion-detection"

# parameter for requests intrusion detection ai services
request_form = {
    "machine_ip": "10.1.1.212",
    "limit_data": 1
}

# calling ai service function
response = requests.get(f'http://{{HOST_IP}}:{{PORT}}{{endpoint}}',
```

```
params=request_form)

# response and results in json format
# print(response.status_code)
pprint.pprint(response.json())
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์เดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/ai-as-a-service
```

4. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_01.py
```

The screenshot shows the VAM Studio interface. In the top navigation bar, the file 'example_01.py' is selected. The left sidebar shows a tree view of project files under 'examples_code' and 'ai-as-a-service'. The main area displays the Python code for 'example_01.py'. The code imports requests, pprint, and json, sets parameters for calling an AI service, and makes a GET request to 'http://HOST_IP:PORT/endpoint'. It then prints the status code and pretty prints the JSON response. Below the code editor is a terminal window titled 'example_01.py - app - vam-studio'. It shows the command 'python3 example_01.py' being run and the resulting JSON output. The output includes a timestamp, a data object with an ID, and a 'analyticsResult' field containing a 'classification' object with a confidence of 1.0, a probability of 0.650128936767578, and a bounding box with coordinates [10, 10, 1500, 1420]. The assignment ID is 4, and the full image path is 'assets/vamstack/images/analytics/results/5-4-1653360000740000000.jpeg'. The source name is 'VAM-VAM'.

```
1 # Import necessary library
2 import requests
3 import pprint
4 import json
5
6 # parameters for calling ai service
7 HOST_IP = "10.1.1.212"
8 PORT = "20000"
9 endpoint = "/intrusion-detection"
10
11 # parameter for requests intrusion detection ai services
12 request_form = {
13     "machine_ip": "10.1.1.212",
14     "limit_data": 5
15 }
16
17 # calling ai service function
18 response = requests.get("http://{}:{}/{}".format(HOST_IP, PORT, endpoint), params=request_form)
19
20 # response and results in json format
21 print(response.status_code)
22 pprint.pprint(response.json())
23
```

```
[{"compute_name": "VAM Intruder Detection and Monitor", "data": {"analytic_id": 5, "analytics_result": {"classification": {"confidence": 1.0, "obj_info": {"cropping": "/assets/vamstack/images/analytics/results/5-4-1653360000740000000.jpeg", "probability": 0.650128936767578, "rollname": None, "rollsize": 10, "x1": 10, "x2": 1500, "y1": 1420, "y2": 1500}, "assignment_id": 4, "fullimg": "/assets/vamstack/images/analytics/results/5-4-1653360000740000000-fullframe.jpg", "imgdim": {"height": 1000, "width": 1920}, "source_name": "VAM-VAM"}, "origin_name": "VAMSTACK-UNPREMISED", "time": "2022-05-24T02:41:20.908594+00:00"}}]
```

รูปที่ 5.5 ผลการรันตัวอย่างโปรแกรมการตรวจจับผู้บุกรุกบน VAM Studio

ผลลัพธ์

```
[{"time": "2022-05-24T02:41:20.908594+00:00", "data": {"imgDim": {"width": 1920,
```

```
        "height": 1080
    },
    "fullImg":
"/assets/vamstack/images/analytics/results/5-4-165336008074000000-fullframe.jpg",
    "sourceId": 250,
    "sourceName": "BDH-VAM",
    "analyticsId": 5,
    "assignmentId": 4,
    "analyticsResult": {
        "cnt": 0,
        "objsInfo": {
            "x1": 1300,
            "x2": 1420,
            "y1": 422,
            "y2": 604,
            "label": "person",
            "roiName": null,
            "trackId": 10,
            "croppedImg":
"/assets/vamstack/images/analytics/results/5-4-250-165336008074000000.jpeg",
                "probability": 0.6050128936767578
            },
            "notification": true
        }
    },
    "origin_name": "VAMSTACK-ONPREMISED",
    "compute_name": "VAM Intruder Detection and Monitor"
}
]
```

LAB 5.2 โปรแกรมการตรวจจับบุคคล (Human Detection)

วัตถุประสงค์

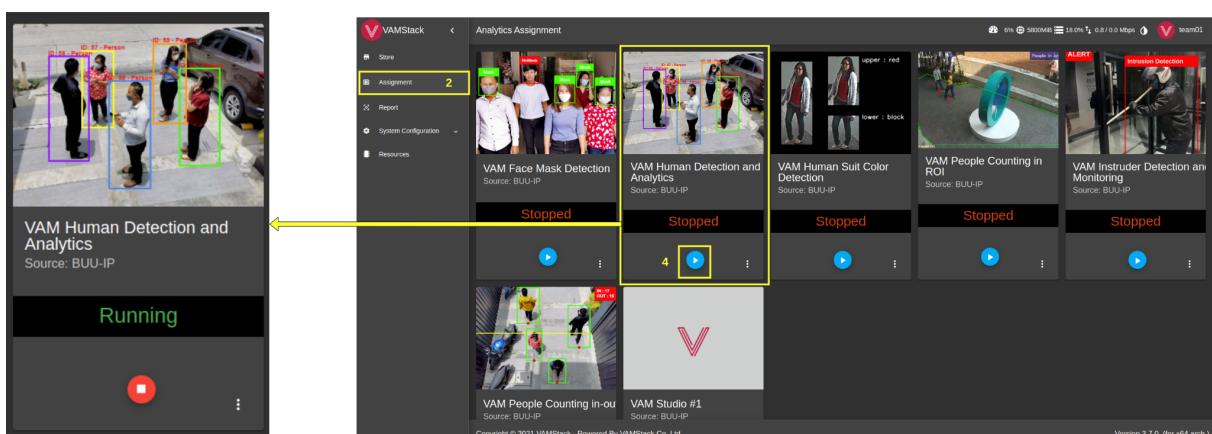
1. เพื่อเรียนรู้วิธีการใช้โปรแกรมการตรวจจับบุคคลบน VAM Platform
2. เพื่อเรียนรู้วิธีการใช้ตัวอย่างโปรแกรมการตรวจจับบุคคลบน VAM Studio

อุปกรณ์ที่เกี่ยวข้อง

1. ชุดการทดลอง VAM Platform

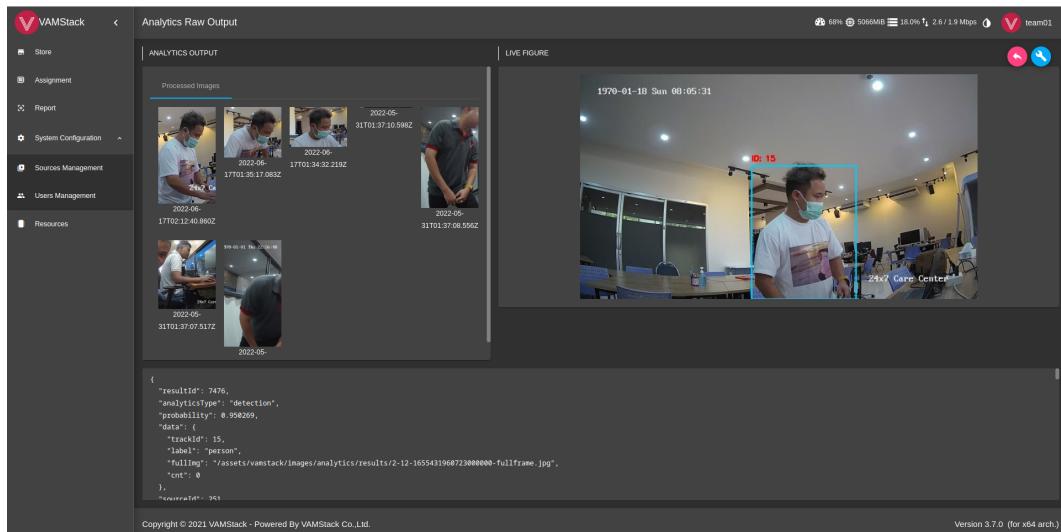
5.2.1 ขั้นตอนการใช้โปรแกรมการตรวจจับบุคคลบน VAM Platform

1. เข้าสู่หน้า VAM Platform โดยพิมพ์ “[10.1.1.212](#)” บนเบราว์เซอร์ และเข้าสู่ระบบของแพลตฟอร์ม (LAB 1.2 หัวข้อ [1.2.1](#))
2. คลิก “Assignment” ที่แนบมาด้านข้าง
3. หากต้องการตรวจจับบุคคลในพื้นที่ที่ต้องการ ผู้ใช้สามารถตั้งค่าพื้นที่ที่สนใจได้ (LAB 1.4 หัวข้อ [1.4.4](#))
4. คลิกปุ่มเปลี่ยนสถานะของ “VAM Human Detection” ให้เป็น “Running”



รูปที่ 5.6 ลำดับการเข้าถึงโปรแกรมการตรวจจับบุคคลบน VAM Platform

5. คลิกที่การ์ด “VAM Human Detection” เพื่อดูการประมวลผลและวิเคราะห์ข้อมูล



รูปที่ 5.7 ผลการรันโปรแกรมการตรวจจับบุคคลบน VAM Studio

5.2.2 human-detection

human detection คือ เซอร์วิสการประมวลผลและวิเคราะห์ข้อมูลสำหรับการตรวจจับบุคคล จากรูปภาพของกล้องวงจรปิดต่าง ๆ โดยจะทำการดึงข้อมูลที่ถูกประมวลผลและวิเคราะห์ข้อมูลมาจาก VAM Platform ที่ถูกจัดเก็บไว้ในฐานข้อมูลมาแสดงผลในรูปแบบของ json บน VAM Studio ซึ่งผู้เรียนสามารถกำหนดจำนวนข้อมูลที่ต้องการรับได้ กรณีที่ไม่มีข้อมูลถูกส่งกลับมา ผู้เรียนจะต้องทำตามหัวข้อ 5.2.1 เพื่อทำการสร้างข้อมูลวิเคราะห์และประมวลผลขึ้น

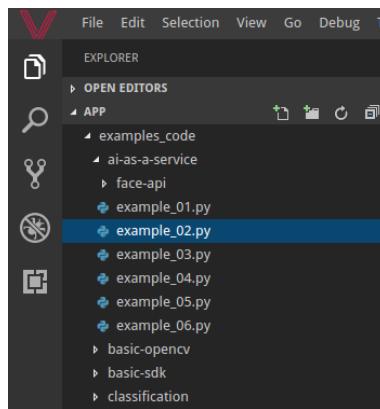
ค่าที่เกี่ยวข้อง

- machine_ip: หมายเลข IP Address ของเครื่องประมวลผลที่ติดตั้ง VAM Platform
- limit_data: ค่ากำหนดจำนวนกลุ่มข้อมูล Json ที่เซอร์วิสทำการดึงมาแสดงผล

ตัวอย่าง กำหนดค่า limit_data คือ 1 หมายความว่า ข้อมูลที่ถูกดึงมาแสดงผลจะมีเพียง 1 ชุดข้อมูลเท่านั้น

ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับบุคคลบน VAM Studio

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > ai-as-a-service > example_02.py ที่แถบเมนูด้านซ้าย



รูปที่ 5.8 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการตรวจจับบุคคล

ตัวอย่างโปรแกรมการตรวจจับบุคคล

```
# import necessary library
import requests
import pprint
import json

# parameters for calling ai service
HOST_IP = "10.1.1.212"
PORT = "20000"
endpoint = "/human-detection"

# parameter for requests human detection ai services
request_form = {
    "machine_ip": "10.1.1.212",
    "limit_data": 1
}

# calling ai service function
response = requests.get(f'http://{{HOST_IP}}:{{PORT}}{{endpoint}}',
params=request_form)

# response and results in json format
print(response.status_code)
pprint.pprint(response.json())
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์ตัวอย่างโปรแกรม

```
cd example_code/ai-as-a-service
```

- ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_02.py
```

The screenshot shows the VAM Studio IDE interface. The left sidebar displays a file tree with several Python files under the 'example_code' directory, including 'example_02.py'. The main editor window shows the code for 'example_02.py', which performs a request to an AI service for human detection. The bottom terminal window shows the execution of the script and its JSON response.

```
File Edit Selection View Go Debug Terminal Help
OPEN EDITORS
APP
example_02.py examples_coderai-as-a-service...
example_01.py
example_02.py
example_03.py
example_04.py
example_05.py
example_06.py
basic_ai_service
basic-sdk
classification
data_visualization
object_detection
examples_image
modules
photos
protos
vam
app.py

example_02.py * example_02.py - app - vam-studio

1 # import necessary library
2 import requests
3 import json
4 import json
5
6 # parameters for calling ai service
7 HOST_IP = "10.1.1.212"
8 PORT = "20000"
9 endpoint = "/human-detection"
10
11 # parameter for requests human detection ai services
12 request_form = {
13     "machine_ip": "10.1.1.212",
14     "limit_data": 5
15 }
16
17 # calling ai service function
18 response = requests.get(f "http://HOST_IP:{PORT}{endpoint}", params=request_form)
19
20 # response and results in json format
21 print(response.status_code)
22 pprint(response.json())

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
data: {"analyticsId": 2,
"analyticsResult": {"ent": 0,
"objInfo": {"color_name": "black",
"color_rgb": "(31, 29, 33)",
"cropping": "/assets/vamstack/images/analytics/results/2-2-250-165381763023500000.jpeg",
"label": "person",
"lat": 37.711891089572146,
"long": -122.17646,
"roiId": 46,
"x1": 176,
"x2": 1444,
"y1": 55,
"y2": 940},
"assignmentId": 2,
"fullImage": "/assets/vamstack/images/analytics/results/2-2-165381763023500000-fullframe.jpg",
"imgData": {"height": 1080, "width": 1920},
"sourceId": "BDH-VAM",
"sourceName": "BDH-VAM"},
"origin_name": "VAMSTACK-ON-PREMISES",
"timestamp": "2023-07-07T09:45:06.999600Z+00:00"
root@felouzeb3b42:/opt/ncc/app/examples_code/ai-as-a-service#
```

รูปที่ 5.9 ผลการรันตัวอย่างโปรแกรมการตรวจจับบุคคลบน VAM Studio

ผลลัพธ์

```
[
  {
    "time": "2022-05-24T04:24:46.636997+00:00",
    "data": {
      "imgDim": {
        "width": 1920,
        "height": 1080
      },
      "fullImg":
      "/assets/vamstack/images/analytics/results/2-2-1653366286366000000-fullframe.jpg",
      "sourceId": 250,
      "sourceName": "BDH-VAM",
      "analyticsId": 2,
      "assignmentId": 2,
      "analyticsResult": {
        "cnt": 0,
        "objsInfo": {
          "x1": 1161,
          "x2": 1415,
          "y1": 537,
          "y2": 1084,
          "label": "person",
          "roiName": null,
          "trackId": 261,
          "color_rgb": "(30, 29, 34)",
          "color_name": "black",
          "croppedImg":
          "/assets/vamstack/images/analytics/results/2-2-250-1653366286366000000.jpeg",
          "probability": 0.6569583415985107
        }
      }
    },
    "origin_name": "VAMSTACK-ONPREMISED",
    "compute_name": "VAM Human Detection and Analytics"
  }
]
```

LAB 5.3 โปรแกรมการตรวจจับสีชุดของบุคคล (People Suit Color Detection)

วัตถุประสงค์

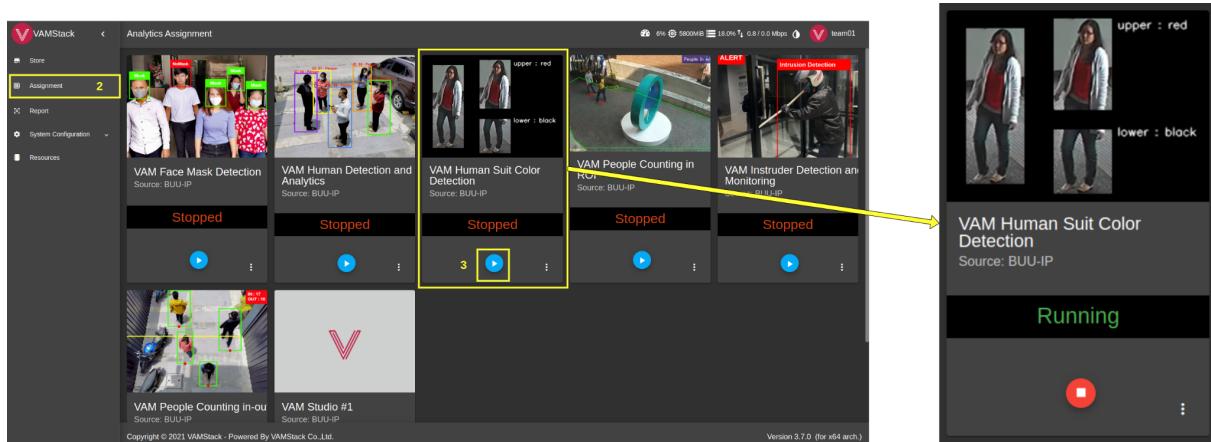
- เพื่อเรียนรู้วิธีการใช้โปรแกรมการตรวจจับสีชุดของบุคคลบน VAM Platform
- เพื่อเรียนรู้วิธีการใช้ตัวอย่างโปรแกรมการตรวจจับสีชุดของบุคคลบน VAM Studio

อุปกรณ์ที่เกี่ยวข้อง

- ชุดการทดลอง VAM Platform

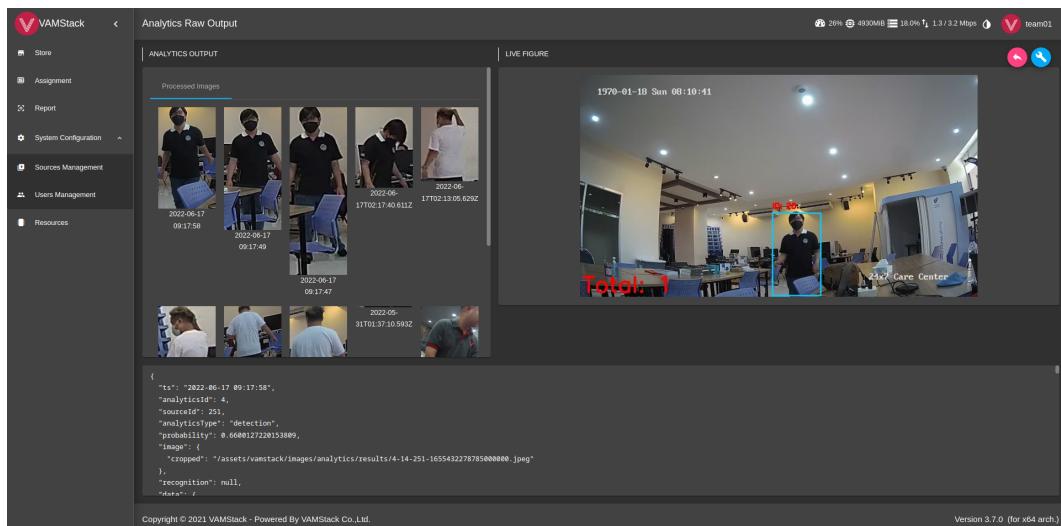
5.3.1 ขั้นตอนการใช้โปรแกรมการตรวจจับสีชุดของบุคคลบน VAM Platform

- เข้าสู่หน้า VAM Platform โดยพิมพ์ “[10.1.1.212](#)” บนเว็บเบราว์เซอร์ และเข้าสู่ระบบของแพลตฟอร์ม (LAB 1.2 หัวข้อ [1.2.1](#))
- คลิก “Assignment” ที่แถบเมนูด้านซ้าย
- คลิกปุ่มเปลี่ยนสถานะของ “VAM Human Suit Color Detection” ให้เป็น “Running”



รูปที่ 5.10 ลำดับการเข้าถึงโปรแกรมการตรวจจับสีชุดของบุคคลบน VAM Platform

- คลิกที่การ์ด “Human Suit Color Detection” เพื่อดูการประมวลผลและวิเคราะห์ข้อมูล



รูปที่ 5.11 ผลการรันโปรแกรมการตรวจจับสีชุดของบุคคลบน VAM Platform

5.3.2 human-suit-color

human-suit-color คือ เซอร์วิสการประมวลผลและวิเคราะห์ข้อมูลสำหรับตรวจจับสีชุดของบุคคล (People Suit Color Detection) ซึ่งได้นำข้อมูลบุคคลที่ตรวจจับบุคคลได้มาต่อ�อด ในส่วนของการแยกแยะสีของชุด ต่อมาจะทำการดึงข้อมูลที่ถูกประมวลผลและวิเคราะห์ข้อมูลมาจาก VAM Platform ที่ถูกจัดเก็บไว้ในฐานข้อมูลมาแสดงผลในรูปแบบของ json บน VAM Studio ซึ่งผู้เรียนสามารถกำหนดจำนวนข้อมูลที่ต้องการรับได้ กรณีที่ไม่มีข้อมูลถูกส่งกลับมา ผู้เรียนจะต้องทำการดึงข้อมูลทั้งหมดที่มีเพียง 1 ชุดข้อมูลเท่านั้น

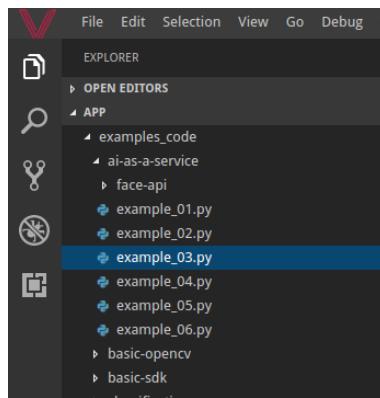
ค่าที่เกี่ยวข้อง

- machine_ip: หมายเลข IP Address ของเครื่องประมวลผลที่ติดตั้ง VAM Platform
- limit_data: ค่ากำหนดจำนวนกลุ่มข้อมูล Json ที่เซอร์วิสทำการดึงมาแสดงผล

ตัวอย่าง กำหนดค่า limit_data คือ 1 หมายความว่า ข้อมูลที่ถูกดึงมาแสดงผลจะมีเพียง 1 ชุดข้อมูลเท่านั้น

ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับสีชุดของบุคคลบน VAM Studio

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > ai-as-a-service > example_03.py ที่แนบเมนูด้านซ้าย



รูปที่ 5.12 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการตรวจจับสีชุดของบุคคล

ตัวอย่างโปรแกรมการตรวจจับสีชุดของบุคคล

```
# import necessary library
import requests
import pprint
import json

# parameters for calling ai service
HOST_IP = "10.1.1.212"
PORT = "20000"
endpoint = "/human-suit-color"

# parameter for requests human suit color detection ai services
request_form = {
    "machine_ip": "10.1.1.212",
    "limit_data": 1
}

# calling ai service function
response = requests.post(f'http://{HOST_IP}:{PORT}{endpoint}',
files=request_form)

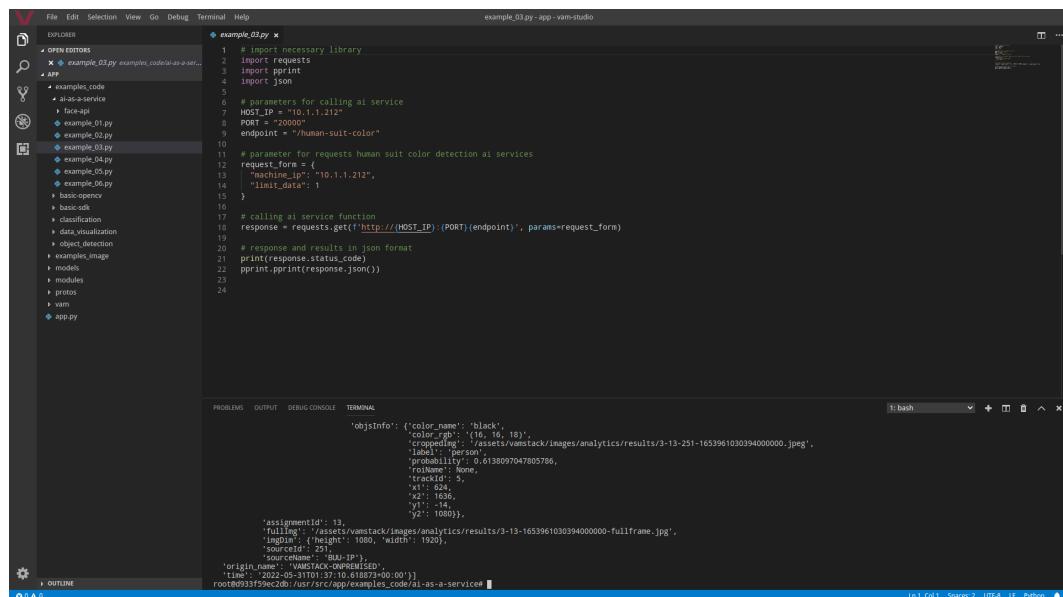
# response and results in json format
print(response.status_code)
pprint.pprint(response.json())
```

3. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์เดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/ai-as-a-service
```

4. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_03.py
```



รูปที่ 5.13 ผลการรันตัวอย่างโปรแกรมการตรวจจับสีชุดของบุคคลบน VAM Studio

ผลลัพธ์

```
[
  {
    "time": "2022-05-24T03:46:20.099987+00:00",
    "data": {
      "imgDim": {
        "width": 1920,
        "height": 1080
      },
      "fullImg":
      "/assets/vamstack/images/analytics/results/3-7-1653363979775000000-fullframe.jpg",
      "sourceId": 250,
      "sourceName": "BDH-VAM",
      "analyticsId": 3,
      "assignmentId": 7,
      "analyticsResult": {
        "cnt": 0,
        "objsInfo": {
          "x1": 1147,
          "x2": 1286,
          "y1": 360,
          "y2": 739,
          "label": "person",
          "roiName": null,
          "trackId": 3,
          "color_rgb": "(45, 47, 52)",
          "color_name": "DarkGray",
          "croppedImg":
          "/assets/vamstack/images/analytics/results/3-7-250-1653363979775000000.jpeg",
          "probability": 0.6465272307395935
        }
      }
    },
    "origin_name": "VAMSTACK-ONPREMISED",
    "compute_name": "VAM Human Suit Color Detection"
  }
]
```

LAB 5.4 โปรแกรมการตรวจจับหน้ากากอนามัยบนใบหน้า (Face Mask Detection)

วัตถุประสงค์

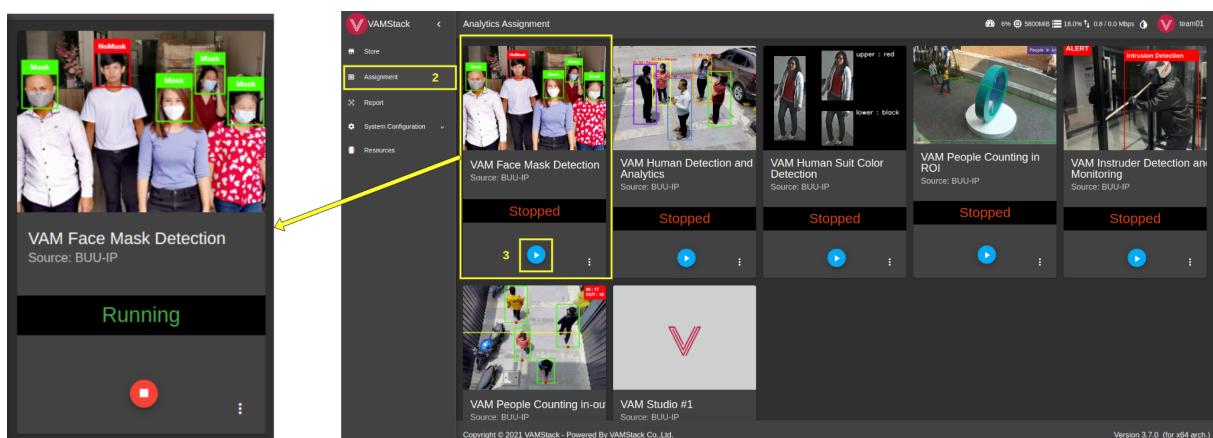
- เพื่อเรียนรู้วิธีการใช้โปรแกรมการตรวจจับหน้ากากอนามัยบนใบหน้าบน VAM Platform
- เพื่อเรียนรู้วิธีการใช้ตัวอย่างโปรแกรมการตรวจจับหน้ากากอนามัยบนใบหน้าบน VAM Studio

อุปกรณ์ที่เกี่ยวข้อง

- ชุดการทดลอง VAM Platform

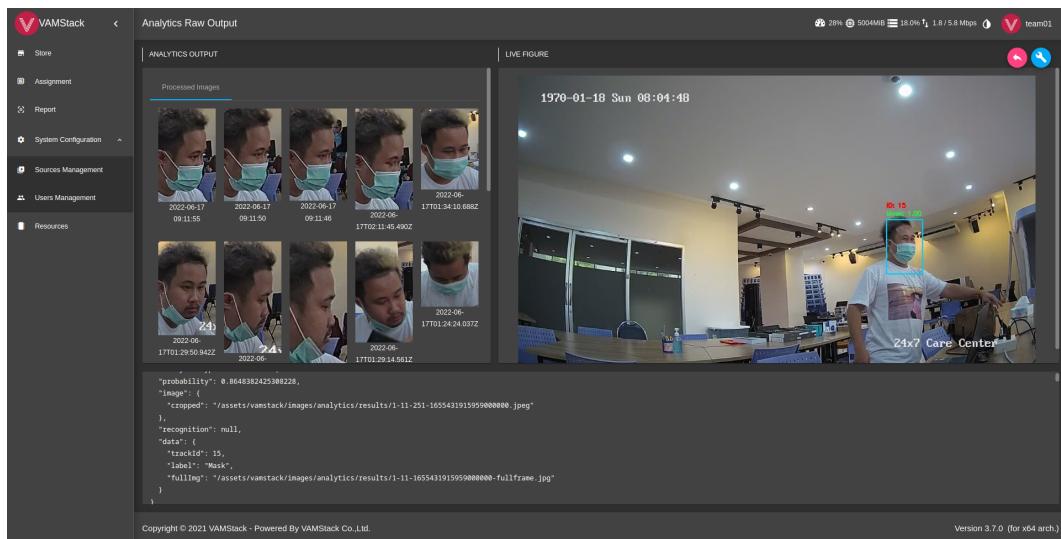
5.4.1 ขั้นตอนการใช้โปรแกรมการตรวจจับหน้ากากอนามัยบนใบหน้าบน VAM Platform

- เข้าสู่หน้า VAM Platform โดยพิมพ์ “[10.1.1.212](#)” บนเว็บเบราว์เซอร์ และเข้าสู่ระบบของแพลตฟอร์ม (LAB 1.2 หัวข้อ [1.2.1](#))
- คลิก “Assignment” ที่ແຄบเมนูด้านซ้าย
- คลิกปุ่มเปลี่ยนสถานะของ “VAM Face Mask Detection” ให้เป็น “Running”



รูปที่ 5.14 ลำดับการเข้าสู่โปรแกรมการตรวจจับหน้ากากอนามัยบนใบหน้าบน VAM Platform

- คลิกที่การ์ด “VAM Face Mask Detection” เพื่อดูการประมวลผลและวิเคราะห์ข้อมูล



รูปที่ 5.15 ผลการรันโปรแกรมการตรวจจับหน้ากากอนามัยบนใบหน้าบน VAM Studio

5.4.2 face-mask-detection

face mask detection คือ เซอร์วิสการประมวลผลและวิเคราะห์ข้อมูลสำหรับตรวจจับการสวมหน้ากากอนามัยบนใบหน้า โดยจะทำการดึงข้อมูลที่ถูกประมวลผลและวิเคราะห์ข้อมูลมาจาก VAM Platform ที่ถูกจัดเก็บไว้ในฐานข้อมูลมาแสดงผลในรูปแบบของ json บน VAM Studio ซึ่งผู้เรียนสามารถกำหนดจำนวนข้อมูลที่ต้องการรับได้ กรณีที่ไม่มีข้อมูลถูกส่งกลับมา ผู้เรียนจะต้องทำตามหัวข้อ 5.4.1 เพื่อทำการสร้างข้อมูลวิเคราะห์และประมวลผลขึ้น

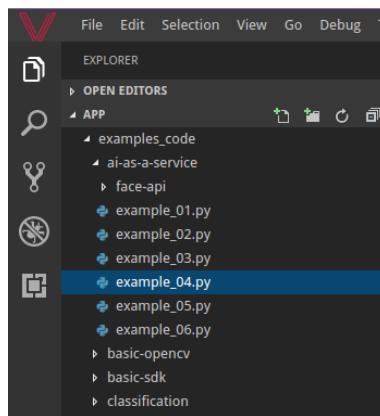
ค่าที่เกี่ยวข้อง

- machine_ip: หมายเลข IP Address ของเครื่องประมวลผลที่ติดตั้ง VAM Platform
- limit_data: ค่ากำหนดจำนวนครุ่นข้อมูล Json ที่เซอร์วิสทำการดึงมาแสดงผล

ตัวอย่าง กำหนดค่า limit_data คือ 1 หมายความว่า ข้อมูลที่ถูกดึงมาแสดงผลจะมีเพียง 1 ชุดข้อมูลเท่านั้น

ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับหน้ากากอนามัยบนใบหน้าบน VAM Studio

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > ai-as-a-service > example_04.py ที่แนบเมนูด้านซ้าย



รูปที่ 5.16 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการตรวจจับหน้ากากอนามัยบนใบหน้า

ตัวอย่างโปรแกรมการตรวจจับหน้ากากอนามัยบนใบหน้า

```
# import necessary library
import requests
import pprint
import json

# parameters for calling ai service
HOST_IP = "10.1.1.212"
PORT = "20000"
endpoint = "/face-mask-detection"

# parameter for requests face mask detection ai services
request_form = {
    "machine_ip": "10.1.1.212",
    "limit_data": 1
}

# calling ai service function
response = requests.post(f'http://{HOST_IP}:{PORT}{endpoint}', 
files=request_form)

# response and results in json format
print(response.status_code)
pprint.pprint(response.json())
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงโอลเดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/ai-as-a-service
```

4. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_04.py
```

The screenshot shows the VAM Studio IDE interface. The left sidebar (EXPLORER) lists files and folders, including 'example_04.py' which is currently selected. The main area (EDITOR) displays the Python code for 'example_04.py'. The code imports requests, pprint, and json, sets parameters for calling an AI service (HOST_IP: '10.1.1.212', PORT: 20000, endpoint: 'face-mask-detection'), creates a request form, makes a GET request to the service, and prints the response in JSON format. The bottom section (TERMINAL) shows the command 'python3 example_04.py' being run and the resulting JSON output, which includes details about a detected face mask.

```

File Edit Selection View Go Debug Terminal Help
example_04.py x
OPEN EDITORS
example_04.py examples_code/ai-as-a-service...
APP
examples_code
ai-as-a-service
  app.py
  example_01.py
  example_02.py
  example_03.py
  example_04.py
  example_05.py
  example_06.py
basic-privacy
  basic-privacy
classification
data-visualization
object-detection
examples-image
models
modules
  python
  vam
app.py

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Ln 1 Col 1 Spaces: 2 UTF-8 LF Python
[{"compute_name": "VAM Face Mask Detection", "data": {"analyticsId": 1}, "analyticsResult": {"objInfo": {"cropping": '/assets/vamstack/images/analytics/results/1-11-251-165484100038500000.jpeg', "label": "NoMask", "score": 0.0005447807312012, "roiName": None, "trackId": 41, "x1": 100, "x2": 132}, "assignmentId": 1, "fullImage": '/vamstack/images/analytics/results/1-11-165484100038500000-fullframe.jpg', "imgDim": {"height": 1080, "width": 1920}, "sourceId": 251, "sourceName": "VAM-IP"}, "origin_name": "VAMSTACK-UNREMOVED", "time": "2022-06-10T09:05:20.81364+00:00"}]
root@023f59ec0b:/user/src/app/examples_code/ai-as-a-service#
```

รูปที่ 5.17 ผลการรันตัวอย่างโปรแกรมการตรวจจับสีชุดของบุคคลบน VAM Studio

ผลลัพธ์

```
[  
  {  
    "time": "2022-05-24T03:28:53.321003+00:00",  
    "data": {  
      "imgDim": {  
        "width": 1920,  
        "height": 1080  
      },  
      "fullImg":  
"/assets/vamstack/images/analytics/results/1-3-1653362933154000000-fullframe.jpg",  
      "sourceId": 250,  
      "sourceName": "BDH-VAM",  
      "analyticsId": 1,  
      "assignmentId": 3,  
      "analyticsResult": {  
        "objsInfo": {  
          "x1": 1215,  
          "x2": 1269,  
          "y1": 610,  
          "y2": 715,  
          "label": "NoMask",  
          "roiName": null,  
          "trackId": 183,  
          "croppedImg":  
"/assets/vamstack/images/analytics/results/1-3-250-1653362933154000000.jpeg",  
          "probability": 0.996286928653717  
        }  
      }  
    },  
    "origin_name": "VAMSTACK-ONPREMISED",  
    "compute_name": "VAM Face Mask Detection"  
  }  
]
```

LAB 5.5 โปรแกรมการนับบุคคลในพื้นที่ที่กำหนด (People Counting in ROI)

วัตถุประสงค์

- เพื่อเรียนรู้วิธีการใช้โปรแกรมการนับบุคคลในพื้นที่ที่กำหนดบน VAM Platform
- เพื่อเรียนรู้วิธีการใช้ตัวอย่างโปรแกรมการนับบุคคลในพื้นที่ที่กำหนดบน VAM Studio

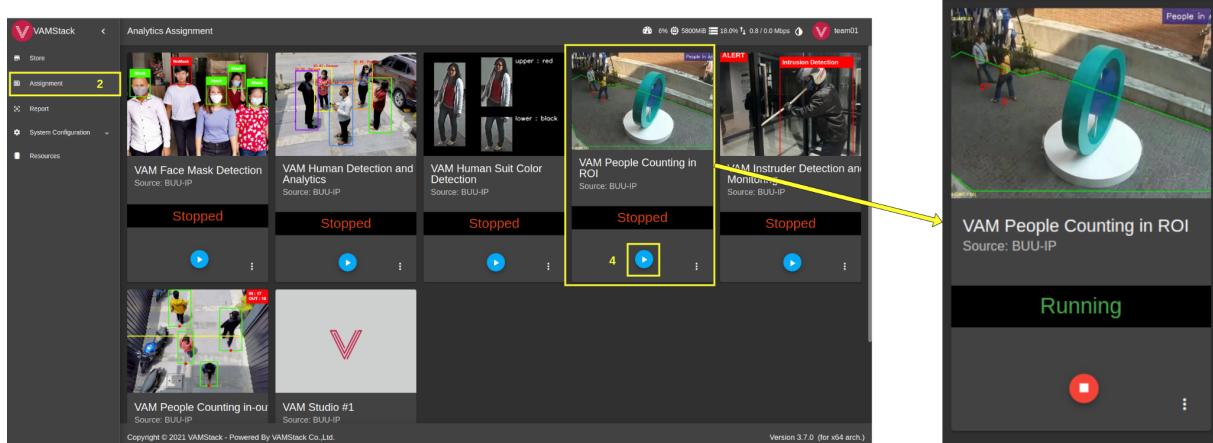
อุปกรณ์ที่เกี่ยวข้อง

- ชุดการทดลอง VAM Platform

5.5.1 ขั้นตอนการใช้โปรแกรมการนับบุคคลในพื้นที่ที่กำหนดบน VAM Platform

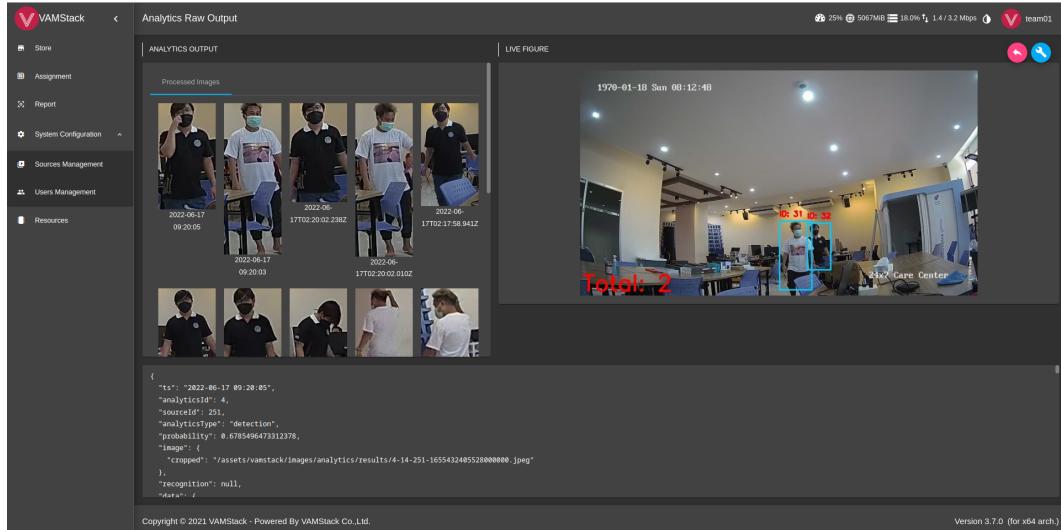
- เข้าสู่หน้า VAM Platform โดยพิมพ์ “[10.1.1.212](#)” บนเว็บเบราว์เซอร์ และเข้าสู่ระบบของแพลตฟอร์ม (LAB 1.2 หัวข้อ [1.2.1](#))
- คลิก “Assignment” ที่แถบเมนูด้านซ้าย
- ทำการตั้งค่าพื้นที่ที่สนใจ (LAB 1.4 หัวข้อ [1.4.4](#)) หากไม่ทำการตั้งค่า โปรแกรมจะทำการประมาณผลและวิเคราะห์ข้อมูลภาพพื้นที่ทั้งหมด
- คลิกปุ่มเปลี่ยนสถานะของ “VAM People Counting in ROI” ให้เป็น “Running”

[OBJ OBJ OBJ]



รูปที่ 5.18 ลำดับการเข้าถึงโปรแกรมการนับบุคคลในพื้นที่ที่กำหนดบน VAM Platform

- คลิกที่การตั้งค่า “VAM People Counting in ROI” เพื่อดูการประมาณผลและวิเคราะห์ข้อมูล



รูปที่ 5.19 ผลการรันตัวอย่างโปรแกรมการนับบุคคลในพื้นที่ที่กำหนดบน VAM Platform

5.5.2 people-counting-in-roi

people counting in roi คือ เซอร์วิสการประมวลผลและวิเคราะห์ข้อมูลสำหรับนับจำนวนบุคคล (People Counting) โดยนำมานับบุคคลที่เข้ามาในสถานที่ที่กำหนด ต่อมาจะทำการดึงข้อมูลที่ถูกประมวลผลและวิเคราะห์ข้อมูลมาจาก VAM Platform ที่ถูกจัดเก็บไว้ในฐานข้อมูลมาแสดงผลในรูปแบบของ json บน VAM Studio ซึ่งผู้เรียนสามารถกำหนดจำนวนข้อมูลที่ต้องการรับได้ กรณีที่ไม่มีข้อมูลถูกส่งกลับมา ผู้เรียนจะต้องทำการตั้งค่า limit_data ตามที่ระบุไว้ในหัวข้อ 5.5.1 เพื่อทำการสร้างข้อมูลวิเคราะห์และประมวลผลขึ้น

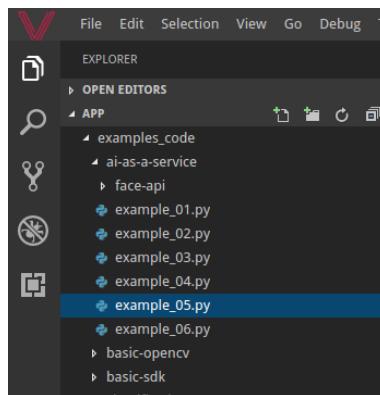
ค่าที่เกี่ยวข้อง

- machine_ip: หมายเลข IP Address ของเครื่องที่รองรับการประมวลผลที่ติดตั้ง VAM Platform
- limit_data: ค่ากำหนดจำนวนกลุ่มข้อมูล Json ที่เซอร์วิสทำการดึงมาแสดงผล

ตัวอย่าง กำหนดค่า limit_data คือ 1 หมายความว่า ข้อมูลที่ถูกดึงมาแสดงผลจะมีเพียง 1 ชุดข้อมูลเท่านั้น

ขั้นตอนการใช้ตัวอย่างโปรแกรมการนับบุคคลในพื้นที่ที่กำหนดบน VAM Studio

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > ai-as-a-service > example_05.py ที่ແນບเมนูด้านซ้าย



รูปที่ 5.20 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการนับบุคคลในพื้นที่ที่กำหนด

ตัวอย่างโปรแกรมการนับบุคคลในพื้นที่ที่กำหนด

```
# import necessary library
import requests
import pprint
import json

# parameters for calling ai service
HOST_IP = "10.1.1.212"
PORT = "20000"
endpoint = "/people-counting-in-roi"

# parameter for requests face mask detection ai services
request_form = {
    "machine_ip": "10.1.1.212",
    "limit_data": 1
}

# calling ai service function
response = requests.post(f'http://{HOST_IP}:{PORT}{endpoint}', 
files=request_form)

# response and results in json format
print(response.status_code)
pprint.pprint(response.json())
pprint.pprint(response.json())
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์ตัวอย่างโปรแกรม

```
cd example_code/ai-as-a-service
```

- ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_05.py
```

The screenshot shows the VAM Studio interface. The left sidebar displays a file tree with several Python files under the 'example_code/ai-as-a-service' directory. The main area shows the content of 'example_05.py'. The code imports requests, json, and os, and defines parameters for calling an AI service at '10.1.1.212:20000' with endpoint '/people-counting-in-roi'. It then makes a request to this endpoint and prints the JSON response. The terminal window at the bottom shows the command 'root@933f59ec2db:/u07/src/app/examples_code/ai-as-a-service# python3 example_05.py' being run.

```

File Edit Selection View Go Debug Terminal Help
OPEN EDITORS
example_05.py examples_code/ai-as-a-service...
APP
example_code
  ai-as-a-service
    face-api
      example_01.py
      example_02.py
      example_03.py
      example_04.py
      example_05.py
      example_06.py
    basic-sdk
    classification
    data_visualization
    object_detection
    example_image
    modules
    protos
    vam
    app.py

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
't: bash' + - x
data: { 'analyticsId': 4,
  'analyticsResult': { 'cnt': 1,
    'objInfo': { 'cropping': '/assets/vamstack/images/analytics/results/4-14-251-1653961030394000000.jpeg',
      'probability': 0.613009747805786,
      'roiName': None,
      'x1': 5,
      'y1': 624,
      'x2': 1636,
      'y2': 14,
      'x2': 1080},
    'assignmentId': 14,
    'fulling': '/assets/vamstack/images/analytics/results/4-14-1653961030394000000-fullframe.jpg',
    'imgDim': { 'height': 1080, 'width': 1920 },
    'sourceName': 'BUU-IP',
    'origin_ip': 'VAMSTACK-09-09-09-09-09-09',
    'time': '2023-09-09T09:09:09.632Z+00:00' }
root@933f59ec2db:/u07/src/app/examples_code/ai-as-a-service#
```

รูปที่ 5.21 ผลการรันตัวอย่างโปรแกรมการนับบุคคลในพื้นที่บน VAM Studio

ผลลัพธ์

```
[
  {
    "time": "2022-05-29T09:56:04.383922+00:00",
    "data": {
      "imgDim": {
        "width": 1920,
        "height": 1080
      },
      "fullImg":
      "/assets/vamstack/images/analytics/results/4-5-1653818164243000000-fullframe.jpg",
      "sourceId": 250,
      "sourceName": "BDH-VAM",
      "analyticsId": 4,
      "assignmentId": 5,
      "analyticsResult": {
        "cnt": 1,
        "objsInfo": {
          "x1": 1118,
          "x2": 1199,
          "y1": 379,
          "y2": 571,
          "label": "person",
          "roiName": null,
          "trackId": 73,
          "croppedImg":
          "/assets/vamstack/images/analytics/results/4-5-250-1653818164243000000.jpg",
          "probability": 0.6139332056045532
        }
      }
    },
    "origin_name": "VAMSTACK-ONPREMISED",
    "compute_name": "VAM People Counting in ROI"
  }
]
```

LAB 5.6 โปรแกรมการนับบุคคลเข้า-ออก (People Counting in-out)

วัตถุประสงค์

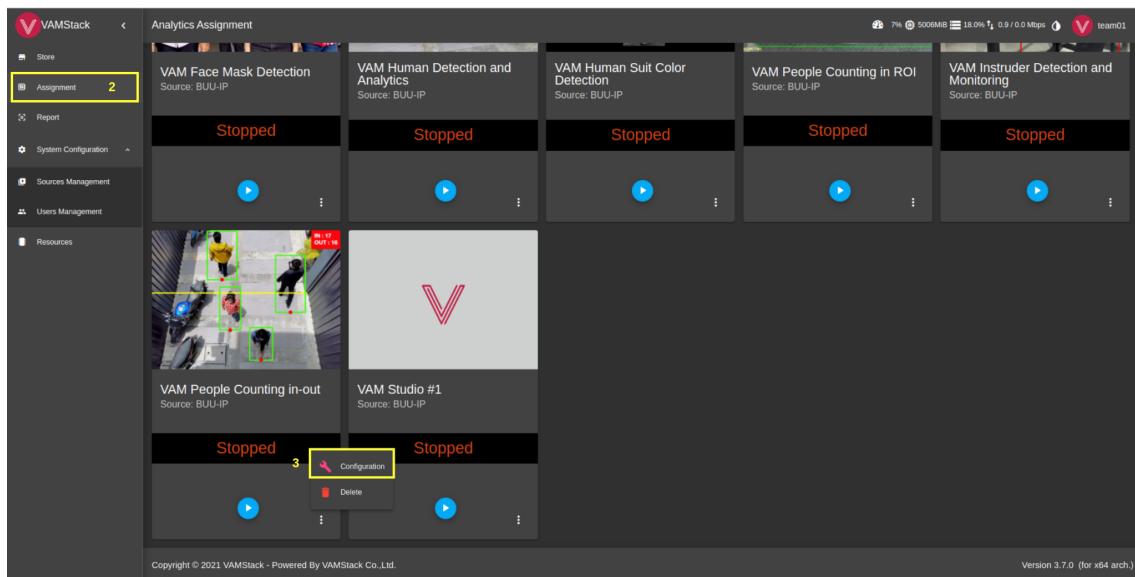
- เพื่อเรียนรู้วิธีการใช้โปรแกรมการนับบุคคลเข้า-ออกบน VAM Platform
- เพื่อเรียนรู้วิธีการใช้ตัวอย่างโปรแกรมการนับบุคคลในพื้นที่ที่กำหนดบน VAM Studio

อุปกรณ์ที่เกี่ยวข้อง

- ชุดการทดลอง VAM Platform

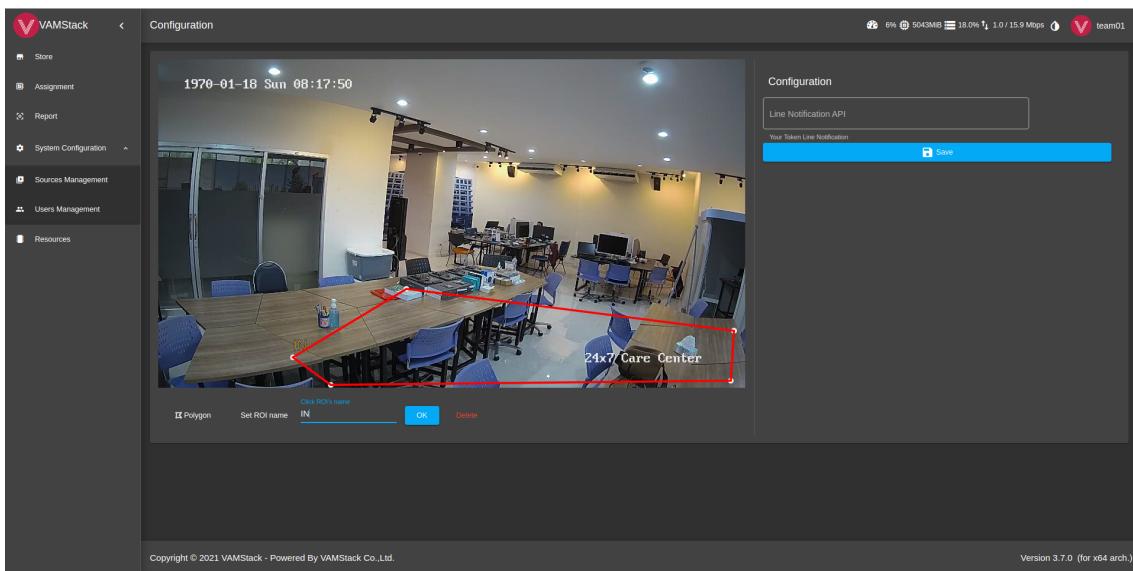
5.6.1 ขั้นตอนการใช้โปรแกรมการนับบุคคลเข้า-ออกบน VAM Platform

- เข้าสู่หน้า VAM Platform โดยพิมพ์ “[10.1.1.212](#)” บนเว็บเบราว์เซอร์ และเข้าสู่ระบบของแพลตฟอร์ม (LAB 1.2 หัวข้อ [1.2.1](#))
- คลิก “Assignment” ที่แถบเมนูด้านซ้าย
- คลิกที่ปุ่ม 3 จุดในการดูของ “VAM People Counting in-out” และคลิกที่เมนู “Configuration”



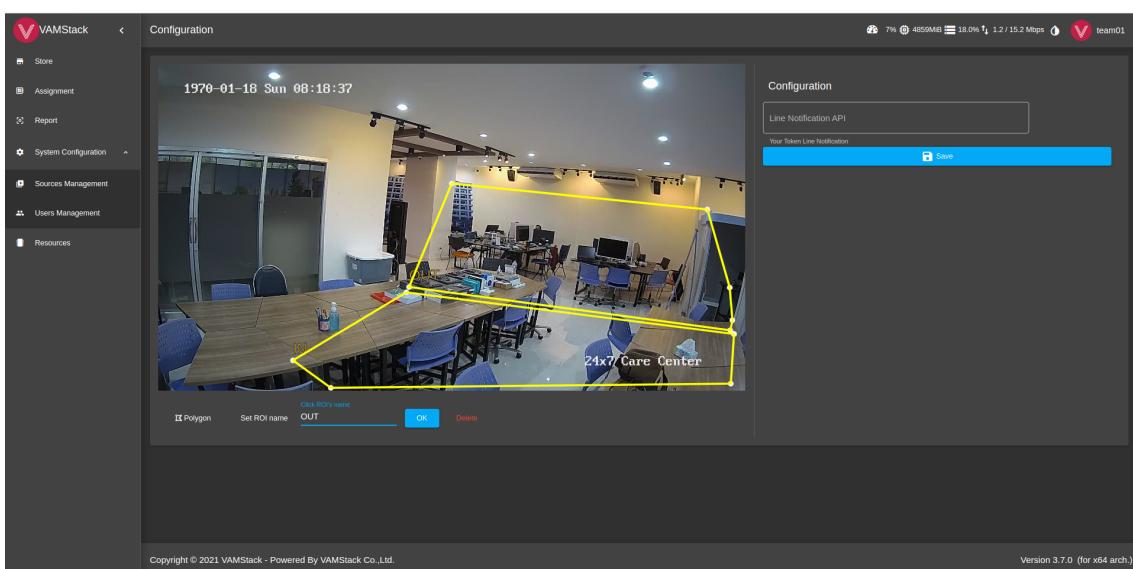
รูปที่ 5.22 ลำดับการเข้าถึงโปรแกรมการนับบุคคลเข้า-ออกบน VAM Platform

- สร้างพื้นที่ ROI และทำการตั้งชื่อว่า “IN” เป็นพื้นที่ที่ใช้นับจำนวนคนเข้าพื้นที่



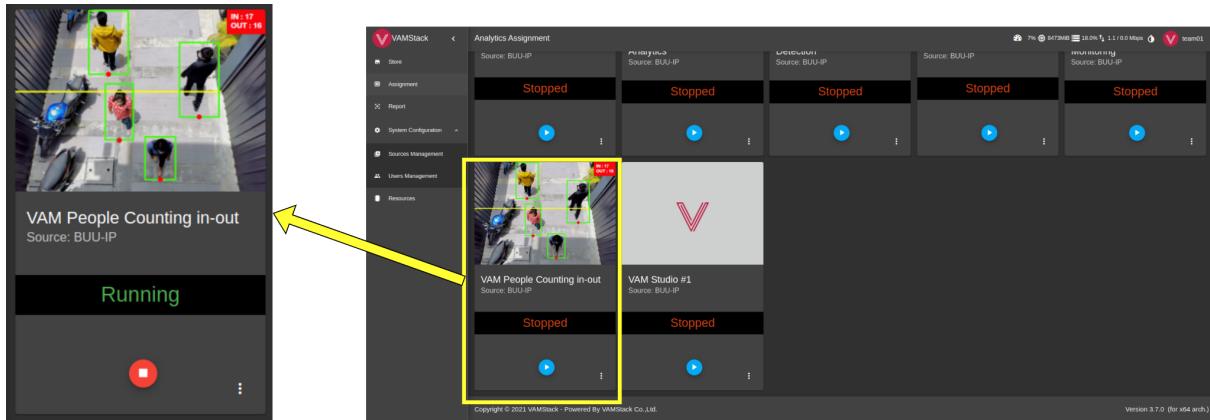
รูปที่ 5.23 กำหนดพื้นที่ที่ใช้นับจำนวนคนเข้าพื้นที่

5. สร้างพื้นที่ ROI และทำการตั้งชื่อว่า “OUT” เป็นพื้นที่ที่ใช้นับจำนวนคนออกจากพื้นที่ จากนั้นคลิกปุ่ม “save” เพื่อยืนยัน การสร้าง ROI



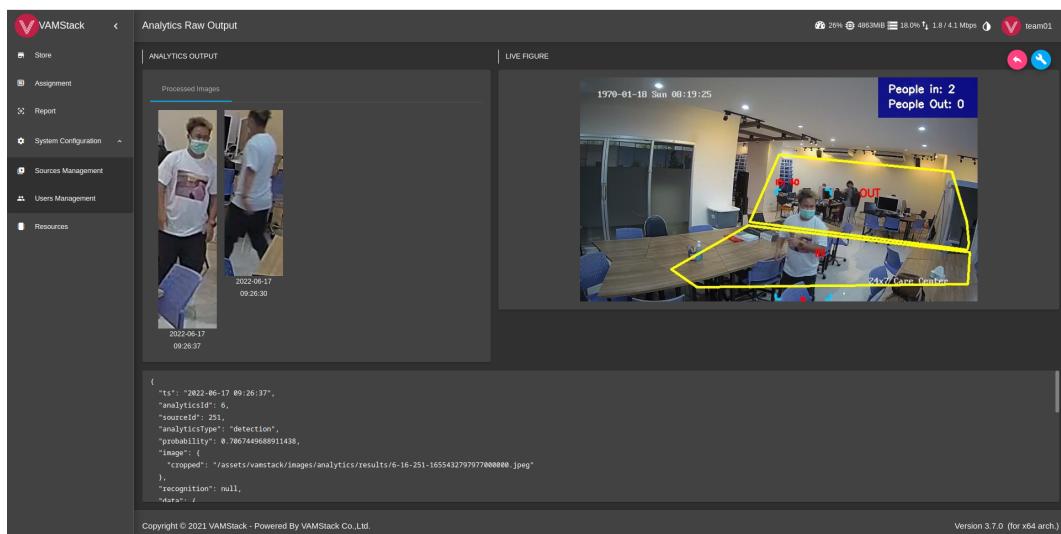
รูปที่ 5.24 กำหนดพื้นที่ที่ใช้นับจำนวนคนออกจากพื้นที่

6. คลิกปุ่มเปลี่ยนสถานะของ “VAM People Counting in-out” ให้เป็น “Running”



รูปที่ 5.25 เปลี่ยนสถานะการทำงานของโปรแกรม

7. คลิกที่การ์ด “People Counting in-out” เพื่อดูการประมวลผลและวิเคราะห์ข้อมูล



รูปที่ 5.26 ผลการรันตัวอย่างโปรแกรมการนับบุคคลเข้า-ออกบน VAM Platform

5.6.2 people-counting-in-out

people counting in-out คือ เซอร์วิสการประมวลผลและวิเคราะห์ข้อมูลสำหรับนับจำนวนการเข้าหรือออกของบุคคล (People Counting in-out) เพื่อนับจำนวนบุคคลเข้าและออกในสถานที่นั้นๆ โดยจะทำการดึงข้อมูลที่ถูกประมวลผลและวิเคราะห์ข้อมูลจาก VAM Platform ที่ถูกจัดเก็บไว้ในฐานข้อมูลมาแสดงผลในรูปแบบของ json บน VAM Studio ซึ่งผู้เรียนสามารถกำหนดจำนวนข้อมูลที่ต้องการรับได้ กรณีที่ไม่มีข้อมูลถูกส่งกลับมา ผู้เรียนจะต้องทำการตั้งค่าตามหัวข้อ 5.6.1 เพื่อทำการสร้างข้อมูลวิเคราะห์และประมวลผลขึ้น

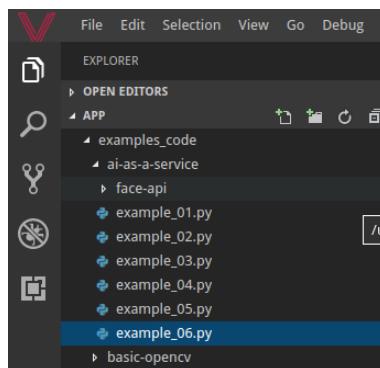
ค่าที่เกี่ยวข้อง

- machine_ip: หมายเลข IP Address ของเครื่องประมวลผลที่ติดตั้ง VAM Platform
- limit_data: ค่ากำหนดจำนวนกลุ่มข้อมูล Json ที่เซอร์วิสทำการดึงมาแสดงผล

ตัวอย่าง กำหนดค่า limit_data คือ 1 หมายความว่า ข้อมูลที่ถูกดึงมาแสดงผลจะมีเพียง 1 ชุดข้อมูลเท่านั้น

ขั้นตอนการใช้ตัวอย่างโปรแกรมการนับบุคคลเข้า-ออกบน VAM Studio

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > ai-as-a-service > example_06.py ที่แถบเมนูด้านซ้าย



รูปที่ 5.27 การถึงไฟล์ตัวอย่างโปรแกรมการนับบุคคลเข้า-ออก

ตัวอย่างโปรแกรมการนับบุคคลเข้า-ออก

```
# import necessary library
import requests
import pprint
import json

# parameters for calling ai service
HOST_IP = "10.1.1.212"
PORT = "20000"
endpoint = "/people-counting-in-out"

# parameter for requests face mask detection ai services
request_form = {
    "machine_ip": "10.1.1.212",
    "limit_data": 1
}

# calling ai service function
response = requests.post(f'http://{HOST_IP}:{PORT}{endpoint}', 
files=request_form)

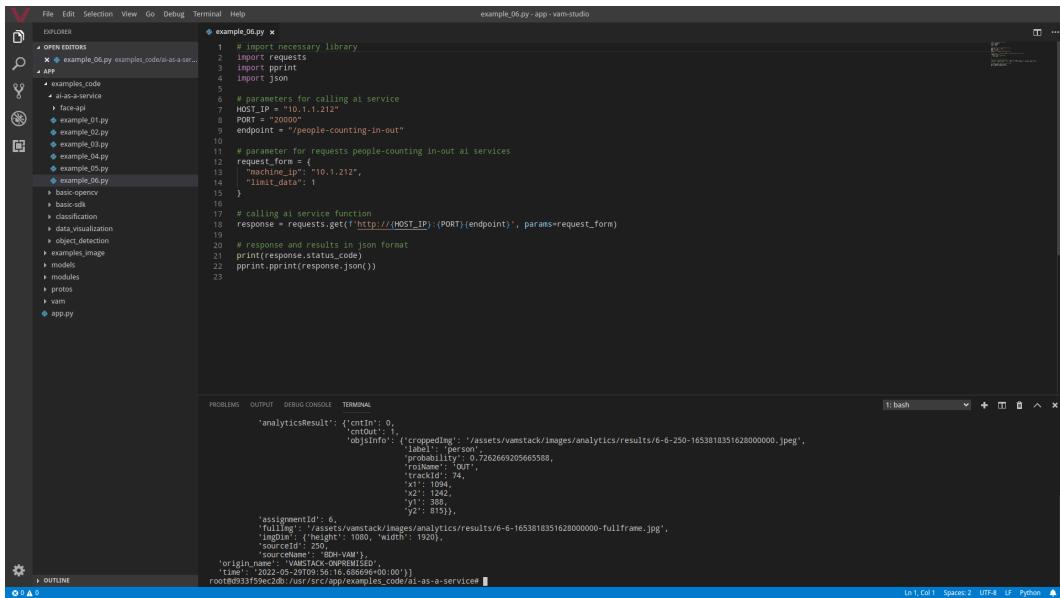
# response and results in json format
print(response.status_code)
pprint.pprint(response.json())
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์เดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/ai-as-a-service
```

4. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_06.py
```



รูปที่ 5.28 ผลการรันตัวอย่างโปรแกรมการนับบุคคลเข้า-ออกบน VAM Studio

លេខព័ត៌មាន

```
[  
 {  
   "time": "2022-05-29T09:56:16.686696+00:00",  
   "data": {  
     "imgDim": {  
       "width": 1920,  
       "height": 1080  
     },  
     "fullImg":  
     "/assets/vamstack/images/analytics/results/6-6-1653818351628000000-fullfr  
ame.jpg",  
     "sourceId": 250,  
     "sourceName": "BDH-VAM",  
     "analyticsId": 6,  
     "assignmentId": 6,  
     "analyticsResult": {  
       "cntIn": 0,  
       "cntOut": 1,  
       "objsInfo": {  
         "x1": 1094,  
         "x2": 1242,  
         "y1": 388,  
         "y2": 538  
       }  
     }  
   }  
 }]
```

```
"y2": 815,  
    "label": "person",  
    "roiName": "OUT",  
    "trackId": 74,  
    "croppedImg":  
        "/assets/vamstack/images/analytics/results/6-6-250-1653818351628000000.jpeg",  
        "probability": 0.7262669205665588  
    }  
}  
},  
"origin_name": "VAMSTACK-ONPREMISED",  
"compute_name": "VAM People Counting in-out"  
}  
]
```

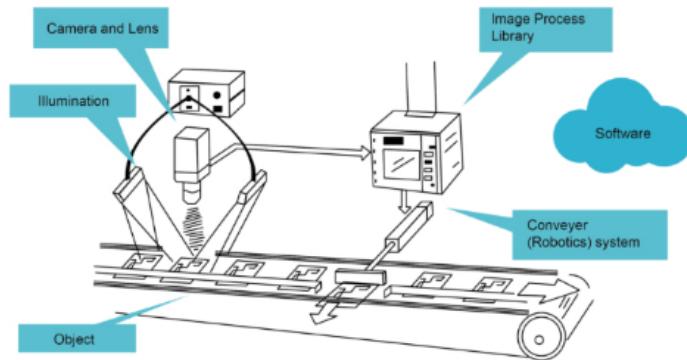
LAB 6 | VAM Industry 4.0 - Machine Vision

Machine Vision

ระบบกล้องจักษ์วิทัศน์ (Machine Vision) คือ ระบบที่เพิ่มความสามารถในการมองเห็นให้เครื่องจักรหรือหุ่นยนต์ ทำหน้าที่ตรวจสอบและควบคุมคุณภาพของชิ้นงาน โดยอ้างอิงจากข้อมูลการประมวลผลและวิเคราะห์ภาพถ่ายจากสภาพแวดล้อมที่ติดตั้งระบบ เนื่องจากมีความแม่นยำสูง ทั้งยังสามารถทำงานได้อย่างสม่ำเสมอและไม่มีความเหลื่อยล้าเหมือนมนุษย์ จึงเป็นระบบที่นิยมใช้ในโรงงานอุตสาหกรรม

องค์ประกอบของระบบกล้องจักษ์วิทัศน์

- ส่วนการรับภาพ ทำหน้าที่ส่งภาพวัตถุไปยังส่วนการประมวลผลข้อมูลภาพต่อไป ซึ่งส่วนของการรับภาพนี้สามารถทำได้หลายทาง เช่น โหลดไฟล์รูปภาพจากแหล่งข้อมูล, จับภาพผ่านกล้องถ่ายภาพหรือภาพผ่านกล้องวิดีโอที่ประกอบไปด้วยกล้อง (Camera) และเลนส์ (Lens) เป็นต้น
- ส่วนการให้แสงสว่าง (Illumination) ทำหน้าที่ให้แสงสว่างแก่วัตถุ นอกเหนือนี้ ระยะห่างของแหล่งกำเนิดแสงจากกล้องและวัตถุ มุม ความเข้ม ความสว่างและสีของแสงจะต้องได้รับการปรับให้เหมาะสมเพื่อให้กล้องมองเห็นวัตถุและคุณลักษณะของวัตถุได้อย่างชัดเจน
- ส่วนการประมวลผลข้อมูลภาพ (Image Process) ทำหน้าที่ประมวลผลและวิเคราะห์ภาพที่ถูกส่งมาจากการรับภาพ และยังทำหน้าที่ตัดสินใจตอบโต้กับข้อมูลที่วิเคราะห์มาได้ เช่น ส่วนการประมวลผลข้อมูลภาพได้รับภาพวัตถุบนสายสานการผลิตแล้วตัดสินใจได้ว่าต้องทำการตัดแยกวัตถุออกจากหรือไม่ ถ้าตัดสินใจว่าตัดวัตถุออก จึงส่งคำสั่งไปที่แขนกล (Robotics system) เพื่อหยิบหรือตันวัตถุออกจากสายพานการผลิต โดยส่วนมากจะใช้คอมพิวเตอร์หรือไมโครโปรเซสเซอร์ที่ทำงานเฉพาะทางมาเป็นส่วนการประมวลผล
- วัตถุที่ต้องการประมวลผล (Object)



รูปที่ 6.1 องค์ประกอบของระบบกล้องจักษ์วิทัศน์

(ที่มา: https://www.mostori.com/blog_detail.php?b_id=79)

ประเภทการทำงานของกล้องจักษ์วิทัศน์

1. Guidance คือ การนำทางหรือการระบุตำแหน่งและทิศทางให้หุ่นยนต์ทำการหยิบจับวัตถุ เพื่อเคลื่อนย้ายผลิตภัณฑ์ การประกอบหรือการจัดเตรียมบรรจุผลิตภัณฑ์
2. Identification คือ การระบุตัวตนหรือการยืนยันป้ายผลิตภัณฑ์
3. Gauging คือ การวัดระยะห่างระหว่างจุดสองจุดขึ้นไปหรือตำแหน่งทางเรขาคณิตบนวัตถุ และกำหนดค่าการวัดเหล่านี้ตรงตามข้อกำหนดหรือไม่
4. Inspection คือ การตรวจสอบหรือตรวจจับข้อบกพร่อง สิ่งแปรเปลี่ยนและความผิดปกติอื่นๆ ในผลิตภัณฑ์ที่ผลิตขึ้น

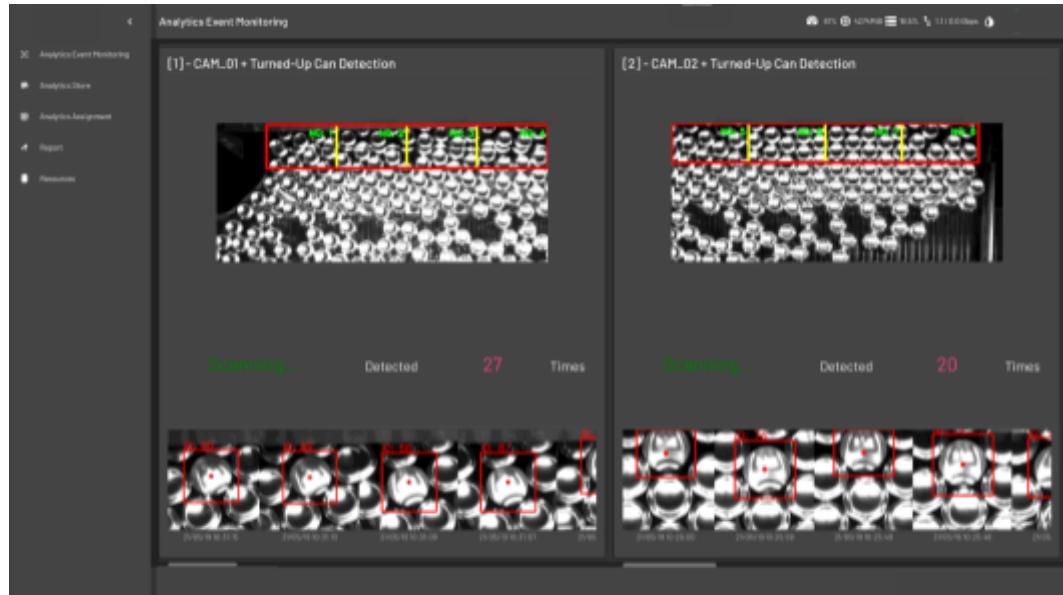
VAM Industry 4.0 - Machine Vision

VAM Industry 4.0 - Machine Vision (VAM MV) เป็น VAM SDK สำหรับใช้งานในระบบกล้องจักษ์วิทัศน์ ทำหน้าที่ประมวลผลและวิเคราะห์ภาพที่เกี่ยวข้องกับงานด้านโรงงานอุตสาหกรรมโดยเฉพาะ ซึ่งในบทเรียนนี้ ผู้เรียนจะได้ทดสอบใช้ตัวอย่างฟังก์ชันด้านกล้องจักษ์วิทัศน์ประเภท Identification และ Inspection

ตัวอย่าง VAM Industry 4.0

1. การตรวจจับกระป๋อง hairy

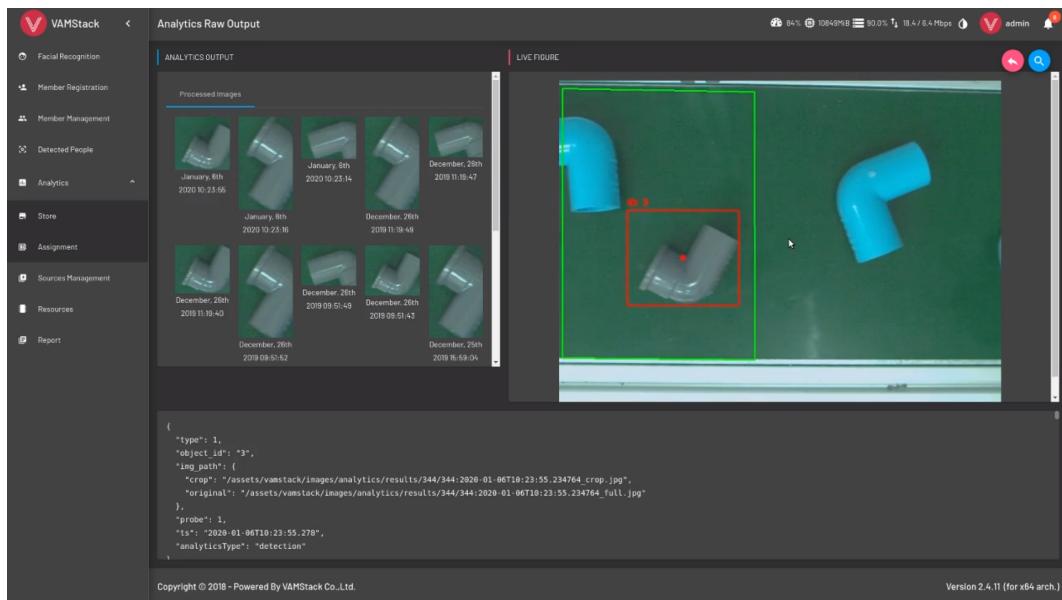
การตรวจจับกระป๋อง hairy จัดเป็นระบบกลจักซ์วิทศน์ประเภท Inspection ทำหน้าที่ตรวจจับกระป๋อง hairy บนสายพานลำเลียง (Conveyor) ที่ผ่านกรอบพื้นที่ที่สนใจ (ROI) เนื่องจากการลำเลียงกระป๋องบนสายพานโดยปกติจะอยู่ในลักษณะค่าว่า



รูปที่ 6.2 โปรแกรมตรวจจับกระป๋อง hairy

2. การตรวจจับสีท่อพีวีซี (PVC)

การตรวจจับสีท่อพีวีซี (PVC) จัดเป็นระบบกลจักซ์วิทศน์ประเภท Inspection ทำหน้าที่ตรวจจับสีท่อพีวีซีที่ไม่ใช้สีฟ้าบนสายพานลำเลียง (Conveyor) ที่ผ่านกรอบพื้นที่ที่สนใจ (ROI) เนื่องจากโปรแกรมจะถูกนำไปใช้ในการตรวจสอบแผนกการผลิตท่อพีวีซีสีฟ้าว่ามีท่อพีวีซีสีอื่นเข้ามาปะปนอยู่หรือไม่



รูปที่ 6.3 โปรแกรมตรวจจับตรวจจับสีห่อพวช (PVC)

LAB 6.1 Identification

วัตถุประสงค์

1. เพื่อเรียนรู้วิธีการใช้ฟังก์ชันการอ่าน OCR
2. เพื่อเรียนรู้วิธีการใช้ฟังก์ชันการอ่าน QRcode และ Barcode

อุปกรณ์ที่เกี่ยวข้อง

1. ชุดการทดลอง VAM Platform

6.1.1 ฟังก์ชันการอ่าน OCR

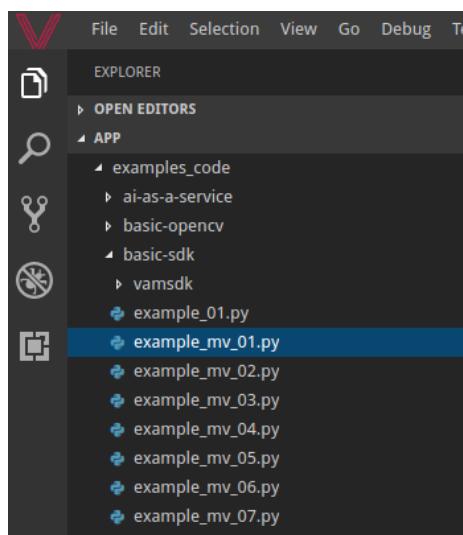
ฟังก์ชันการอ่าน OCR ทำหน้าที่ในการช่วยอ่านและแปลงข้อความตัวอักษรภาษาอังกฤษรวมถึงตัวเลขในรูปภาพให้อยู่ในรูปแบบของตัวอักษรหรือข้อความที่สามารถนำไปประมวลผลและใช้สำหรับค้นหา

ฟังก์ชันที่เกี่ยวข้อง

1. OCR(image)
 - 1.1. image: รับพารามิเตอร์เป็นรูปภาพที่มีตัวอักษรหรือตัวเลข
 - 1.2. ส่งคืนรูปภาพที่ผ่านการประมวลผลเป็นข้อมูลประเภท numpy.ndarray
 - 1.3. ส่งคืนข้อมูลประเภท Json ประกอบด้วย
 - 1.3.1. text คือ ข้อความหรือตัวอักษรที่อ่านได้จากรูปภาพ
 - 1.3.2. bbox คือ จุดพิกัดคู่อันดับ (x,y) สำหรับวัดกรอบตรวจจับข้อความหรือตัวอักษรเป็นข้อมูลประเภท List

ขั้นตอนการใช้ตัวอย่างโปรแกรมการอ่าน OCR

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > basic-sdk > example_mv_01.py ที่แถบเมนูด้านซ้าย



รูปที่ 6.4 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการอ่าน OCR

ตัวอย่างโปรแกรมการอ่าน OCR

```
# import library
import cv2
from vamsdk import machine_vision
from pprint import pprint

# select image and calling opencv imread function
image = cv2.imread("../examples_image/text/text1.png")
```

```
# calling function for read OCR
it = machine_vision.Identification()
ocr_image,results = it.OCR(image)

# returning result from read OCR
# {'text' : text, 'bbox' : [x1, y1, x2, y2]}

# where (x1, y1) is topleft of boundingbox
# where (x2, y2) is bottomright of boundingbox
pprint(results)

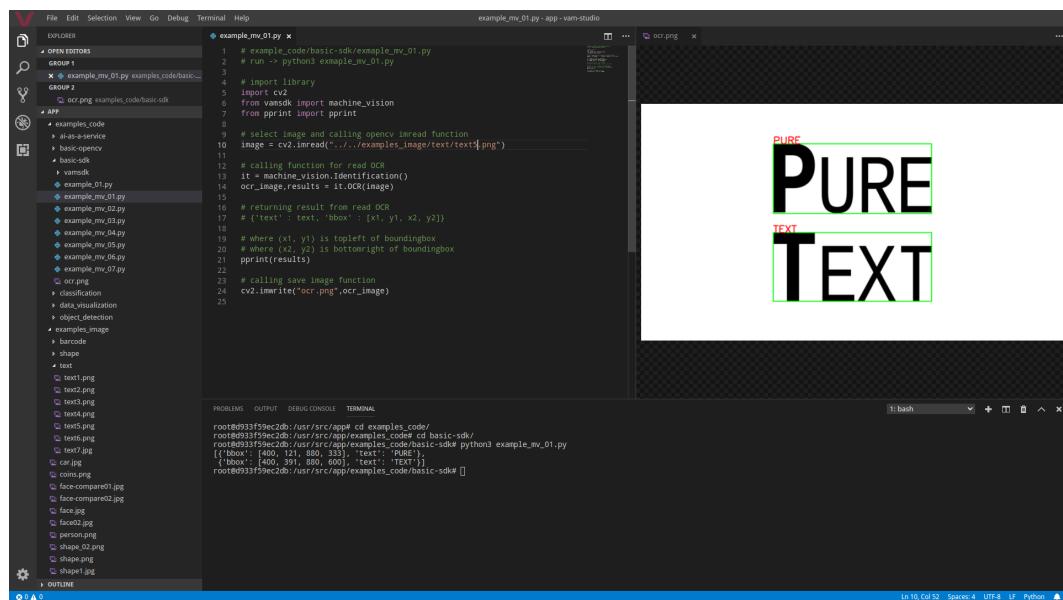
# calling save image function
cv2.imwrite("ocr.png",ocr_image)
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงโฟลเดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

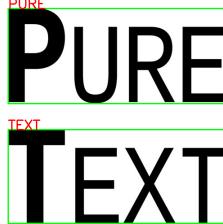
```
cd example_code/basic-sdk
```

4. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_mv_01.py
```



รูปที่ 6.5 ผลการรันตัวอย่างโปรแกรมการอ่าน OCR บน VAM Studio

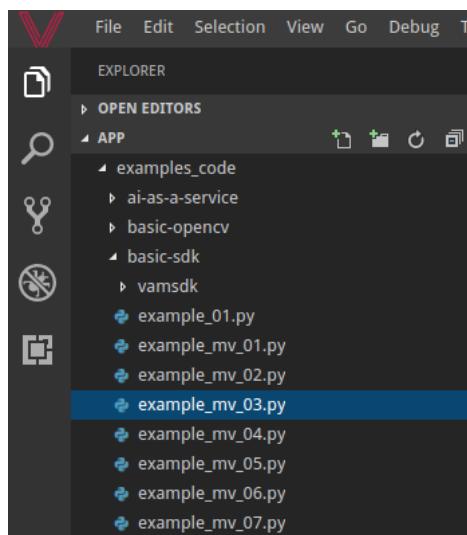
ผลลัพธ์	
ocr.png	results
	[{ 'bbox': [400, 121, 880, 333], 'text': 'PURE'}, { 'bbox': [400, 391, 880, 600], 'text': 'TEXT' }]]

6.1.2 พังก์ชันการอ่าน QRCode และ Barcode

พังก์ชันอ่าน QRCode และ Barcode ทำหน้าที่ในการช่วยอ่านและแปลงข้อมูลจากรูปภาพที่มี QRcode และ Barcode ให้อยู่ในรูปแบบของตัวอักษรหรือข้อความที่สามารถนำไปประมวลผลและใช้สำหรับค้นหา

พังก์ชันที่เกี่ยวข้อง

1. barcode_qrcode(image)
 - 1.1. image: รับพารามิเตอร์เป็นรูปภาพที่ QRcode และ Barcode
 - 1.2. ส่งคืนรูปภาพที่ผ่านการประมวลผลเป็นข้อมูลประเภท numpy.ndarray
 - 1.3. ส่งคืนข้อมูลประเภท Json ประกอบด้วย
 - 1.3.1. text คือ ข้อความหรือตัวอักษรที่อ่านได้จากรูปภาพ
 - 1.3.2. type คือ ประเภทของ barcode ที่ตรวจจับ
 - 1.3.3. bbox คือ จุดพิกัดคู่อันดับ (x,y) สำหรับวดกรอบตรวจจับ QRcode และ Barcode เป็นข้อมูลประเภท List



รูปที่ 6.6 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการอ่าน QRCode และ Barcode

ขั้นตอนการใช้ตัวอย่างโปรแกรมการอ่าน QRcode

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > basic-sdk > example_mv_03.py ที่แถบเมนูด้านซ้าย

ตัวอย่างโปรแกรมการอ่าน QRcode

```
# import library
```

```

import cv2
from vamsdk import machine_vision
from pprint import pprint

# select image and calling opencv imread function
image = cv2.imread("../examples_image/barcode/qrcode.jpg")

# calling function for read barcode and QRCode
it = machine_vision.Identification()
qrcode_image,result = it.barcode_qrcode(image)

# returning result from read barcode and QRCode
# {'text' : text, 'type': type, 'bbox' : [x1, y1, x2, y2]}

# where type is type of barcode
# where (x1, y1) is topleft of boundingbox
# where (x2, y2) is bottomright of boundingbox
pprint(result)

# calling save image function
cv2.imwrite("qrcode.png",qrcode_image)

```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงโฟลเดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/basic-sdk
```

4. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_mv_03.py
```

```

File Edit Selection View Go Debug Terminal Help
OPEN EDITORS
GROUP 1
example_mv_03.py examples_code/basic-sdk/
GROUP 2
qrcode.png examples_code/basic-sdk/
APP
examples_code
av-as-service
basic-app
basic-sdk
vamsdk
example_mv_01.py
example_mv_02.py
example_mv_03.py
example_mv_04.py
example_mv_05.py
example_mv_06.py
example_mv_07.py
qr.png
qrcode.png
qrcode_image
barcode
examples_image
barcode
examples_image
text
text.png
text2.png
text3.png
text4.png
text5.png
text6.png
text7.png
car.jpg
coins.png
face-compare01.jpg
face-compare02.jpg
face.jpg
face01.jpg
example_mv_03.py
# example_mv_03.py
# run -> python3 example_mv_02.py
# importing library
import cv2
from vamsdk import machine_vision
from pprint import pprint
# select image and calling opencv imread function
image = cv2.imread('../examples_image/barcode/qrcode.jpg')
# calling function for read barcode and QRcode
it = machine_vision.Identification()
qr_code_image,result = it.barcode_qr_code(image)
# returning result from read barcode and QRCode
if result != None:
    # where (x1, y1) is topleft of boundingbox
    # where (x2, y2) is bottomright of boundingbox
    pprint(result)
# calling save image function
cv2.imwrite("qrcode.png",qr_code_image)

```

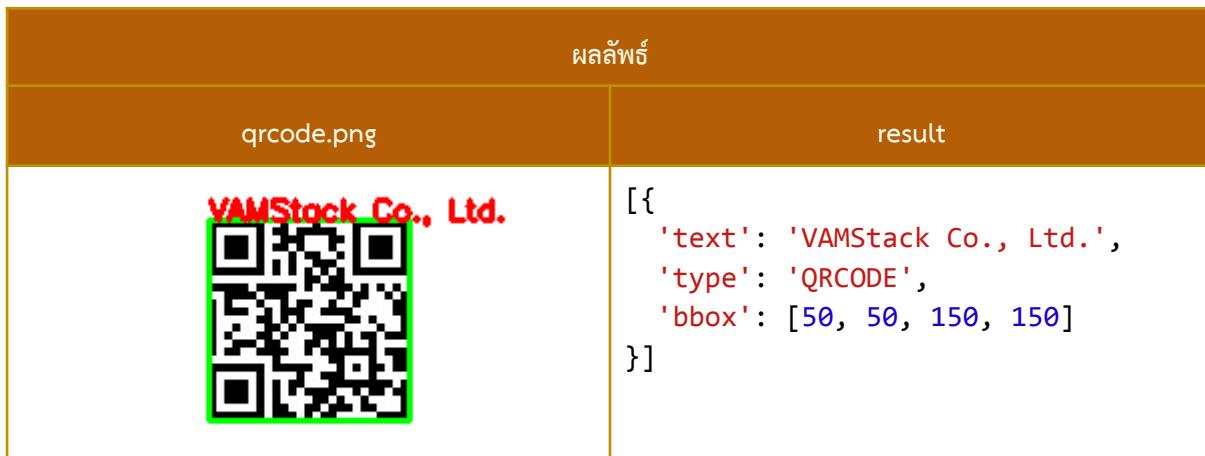
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```

root@933f59ec2db:/usr/src/app# cd examples_code/
root@933f59ec2db:/usr/src/app/examples_code# cd basic-sdk/
root@933f59ec2db:/usr/src/app/examples_code/basic-sdk# python3 example_mv_01.py
[{"bbox": [400, 121, 880, 335], "text": "PURE"}, {"bbox": [400, 391, 880, 600], "text": "TEXT"}, {"bbox": [50, 50, 150, 160], "text": "VAMStack Co., Ltd.", "type": "QRCODE"}]
root@933f59ec2db:/usr/src/app/examples_code/basic-sdk# []

```

รูปที่ 6.7 ผลการรันตัวอย่างโปรแกรมการอ่าน QRcode และ Barcode บน VAM Studio



LAB 6.2 Inspection

วัตถุประสงค์

1. เพื่อเรียนรู้วิธีการใช้ฟังก์ชันการตรวจสอบรูปร่างวัตถุ
2. เพื่อเรียนรู้วิธีการใช้ฟังก์ชันการตรวจสอบเส้นขอบวัตถุ
3. เพื่อเรียนรู้วิธีการใช้ฟังก์ชันการตรวจสอบน้ำหนัก
4. เพื่อเรียนรู้วิธีการใช้ฟังก์ชันการหาค่าสีของวัตถุ

อุปกรณ์ที่เกี่ยวข้อง

1. ชุดการทดลอง VAM Platform

6.2.1 ฟังก์ชันการตรวจสอบรูปร่างวัตถุ (Shape Detection)

ฟังก์ชันการตรวจสอบรูปร่างวัตถุ ทำหน้าที่ประมวลผลและวิเคราะห์ข้อมูลภาพ เพื่อหาวัตถุที่มีโครงสร้างเชิงเรขาคณิตที่อยู่ในภาพ เพื่อนำไปใช้ประโยชน์ในการตรวจสอบรูปร่างขึ้นงานที่มีโครงสร้างเชิงเรขาคณิตในโรงงานอุตสาหกรรม

ฟังก์ชันที่เกี่ยวข้อง

1. setThreshold(thresh, maxThresh, type)

ฟังก์ชันนี้ใช้สำหรับการคัดแยกวัตถุจากภาพพื้นหลังสีดำโดยใช้เทคนิคของ Threshold หากต้องการเรียกใช้ต้องถูกเรียกใช้ก่อนฟังก์ชัน detect

- 1.1. รับพารามิเตอร์ประเภทตัวเลขตั้งแต่ 0-255

1.1.1. thresh: ค่าจุดเปลี่ยนสี (ถ้าค่าสีที่เพ็บในรูปภาพต่ำกว่าที่กำหนดจะถูกเปลี่ยนเป็นสีดำ ถ้าค่าสีที่เพ็บในรูปภาพมากกว่าจะถูกปรับเป็นสีขาว)

1.1.2. maxThresh: ค่าสูงสุดถูกกำหนดเป็นสีขาว (255) ซึ่งผู้ใช้สามารถให้ค่าของ maxThresh ต่ำกว่า 255 ได้

- 1.2. type: ประเภทรูปแบบการเปลี่ยนค่าพิกเซล หากผู้ใช้ไม่ได้กำหนด พารามิเตอร์นี้จะถูกกำหนดเป็น

cv2.THRESH_BINARY โดยอัตโนมัติ ซึ่งผู้ใช้สามารถค่าเป็น cv2.THRESH_BINARY_INV ได้โดยที่เนื่องจากการเปลี่ยนสีของภาพจะเปลี่ยนไปตามทฤษฎี ([Simple Threshold](#))

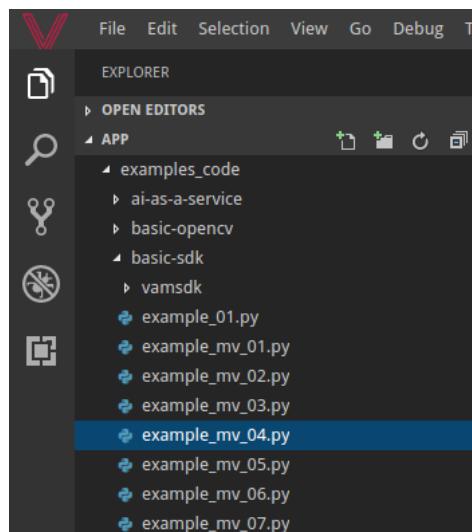
- 1.3. ฟังก์ชันนี้ไม่มีการส่งคืนค่าใด ๆ

2. detect(image)

- 2.1. รับพารามิเตอร์เป็นรูปภาพสี RGB ที่มีวัตถุโครงสร้างเชิงเรขาคณิต ได้แก่ สามเหลี่ยม สี่เหลี่ยม ห้าเหลี่ยมและวงกลม
- 2.2. 送คืนรูปภาพที่ผ่านการประมวลผลเป็นข้อมูลประเภท numpy.ndarray
- 2.3. 送คืนข้อมูลประเภท Json ประกอบด้วย
 - 2.3.1. shape คือ ข้อความระบุปร่างของวัตถุที่ตรวจจับ
 - 2.3.2. bbox คือ จุดพิกัดคู่อันดับ (x,y) สำหรับวัดกรอบตรวจจับวัตถุ เป็นข้อมูลประเภท List

ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับรูปร่างวัตถุ

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > basic-sdk > example_mv_04.py ที่แถบเมนูด้านซ้าย



รูปที่ 6.8 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการตรวจจับรูปร่างวัตถุ

ตัวอย่างโปรแกรมการตรวจจับรูปร่างวัตถุ

```
# import library
import cv2
from vamsdk import machine_vision
from pprint import pprint

# select image and calling opencv imread function
image = cv2.imread("../examples_image/shape/shape1.jpg")

# calling function for detecting shape
sd = machine_vision.ShapeDetector()
sd.setThreshold(thresh=50, maxThresh=100, type=cv2.THRESH_BINARY)
shape_image,result = sd.detect(image)

# returning result from shape detection
# {'shape' : shape, 'bbox' : [x1, y1, x2, y2]}

# where shape is triangle, rectangle, pentagon and circle
# where (x1, y1) is topleft of boundingbox
# where (x2, y2) is bottomright of boundingbox
pprint(result)

# calling save image function
cv2.imwrite("shape_detection.png",shape_image)

# detected only rectangle
for r in result:
    if r["shape"] == "rectangle":
        (x1,y1) = r["bbox"][0], r["bbox"][1]
        (x2,y2) = r["bbox"][2], r["bbox"][3]
        cv2.rectangle(image, (x1,y1), (x2,y2), (36,255,12), 2)

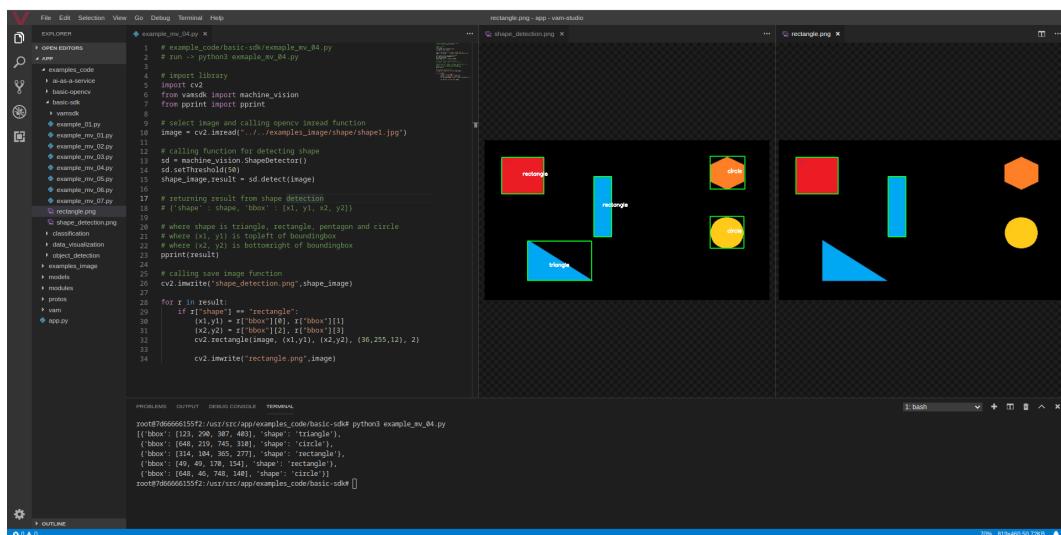
    # calling save image function
    cv2.imwrite("rectangle.png",image)
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์เดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

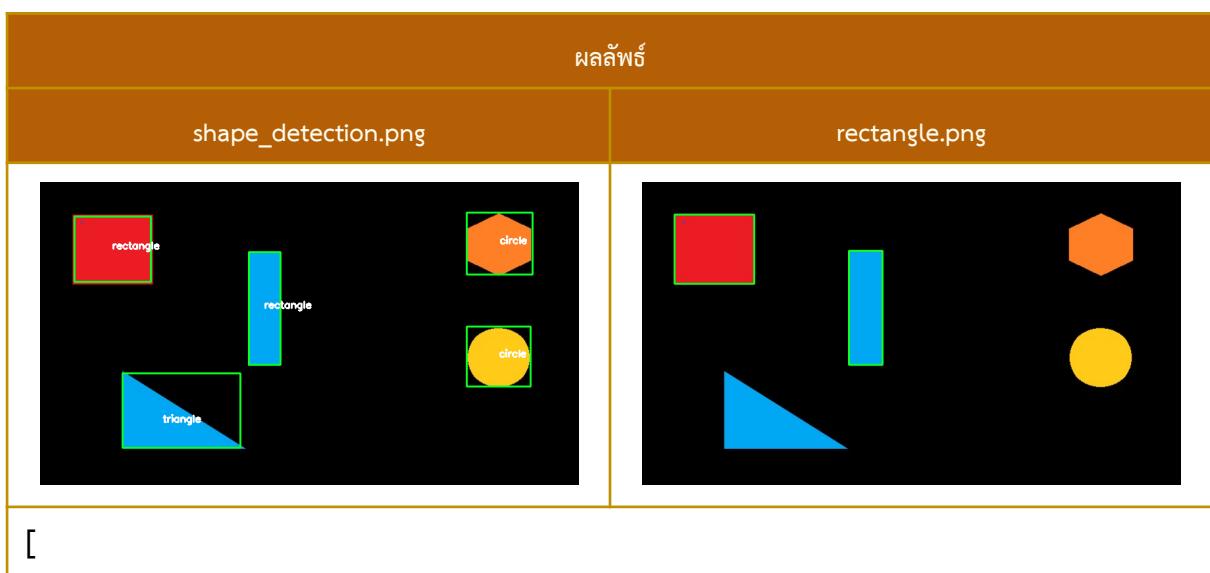
```
cd example_code/basic-sdk
```

4. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_mv_04.py
```



รูปที่ 6.9 ผลการรันตัวอย่างโปรแกรมการตรวจจับรูปร่างวัตถุบน VAM Studio



```

{'bbox': [134, 123, 723, 215], 'shape': 'triangle'},
{'bbox': [648, 215, 723, 239], 'shape': 'circle'},
{'bbox': [317, 239, 723, 263], 'shape': 'rectangle'},
{'bbox': [52, 263, 723, 287], 'shape': 'rectangle'},
{'bbox': [648, 287, 723, 311], 'shape': 'circle'}
]

```

6.2.2 ฟังก์ชันการตรวจจับเส้นขอบวัตถุ (Edge Detection)

ฟังก์ชันการตรวจจับเส้นขอบวัตถุ ทำหน้าที่ประมวลผลและวิเคราะห์ข้อมูลภาพเพื่อหาเส้นขอบวัตถุภายในภาพ เพื่อใช้ในการตรวจสอบความสมบูรณ์และสามารถใช้ในการตรวจสอบหารอยร้าวหรือรอยแตกของวัตถุหรือผลิตภัณฑ์

ฟังก์ชันที่เกี่ยวข้อง

1. setCanny(lower_threshold, upper_threshold)

ฟังก์ชันนี้ใช้สำหรับค้นหาเส้นขอบของวัตถุโดยใช้เทคนิคของ Canny Edge Detection หากต้องการเรียกใช้ต้องถูกเรียกใช้ก่อนฟังก์ชัน detect

1.1. รับพารามิเตอร์ประเภทตัวเลขตั้งแต่ 0-255

1.1.1. lower_threshold: ค่าคงที่ต่ำสุดที่นำไปใช้ในกระบวนการ Hysteresis Thresholding ของ Canny Edge Detection

1.1.2. upper_threshold: ค่าคงที่สูงสุดที่นำไปใช้ในกระบวนการ Hysteresis Thresholding ของ Canny Edge Detection

2. detect(image)

2.1. image: รับพารามิเตอร์รูปภาพสี RGB

2.2. ส่งคืนรูปภาพที่ผ่านการประมวลผลเป็นข้อมูลประเภท numpy.ndarray

2.3. ส่งคืนข้อมูลประเภท Json ประกอบด้วย

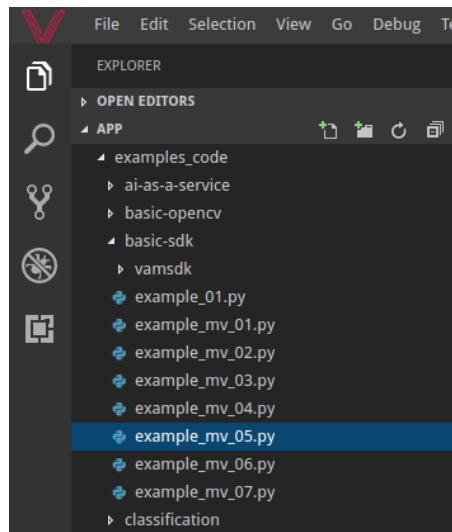
2.3.1. area-pixel คือ ตัวเลขพื้นที่ทั้งหมดของวัตถุที่ตรวจพบเส้นขอบ มีหน่วยเป็น pixel

2.3.2. bbox คือ จุดพิกัดคู่อันดับ (x,y) สำหรับวัดกรอบตรวจจับวัตถุ เป็นข้อมูลประเภท List

ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับเส้นขอบวัตถุ

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))

2. คลิก example_code > basic-sdk > example_mv_05.py ที่แถบเมนูด้านซ้าย



รูปที่ 6.10 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการตรวจจับเส้นขอบวัตถุ

ตัวอย่างโปรแกรมการตรวจจับเส้นขอบวัตถุ

```
# import library
import cv2
from vamsdk import machine_vision
from pprint import pprint

# select image and calling opencv imread function
image = cv2.imread("../examples_image/shape/shape2.png")

# calling function for detect edge
ed = machine_vision.EdgeDetection()
ed.setCanny(10,100)
edge_image,result = ed.detect(image)

# returning result from detect edge
# {'area-pixe' : area, 'edge' : [c]}

# where the area is an area of detected object
# where the edge is the location of the edge in the image
pprint(result)

for r in result:
```

```
cv2.drawContours(image, r["edge"], -1, (0, 255, 0), 2)

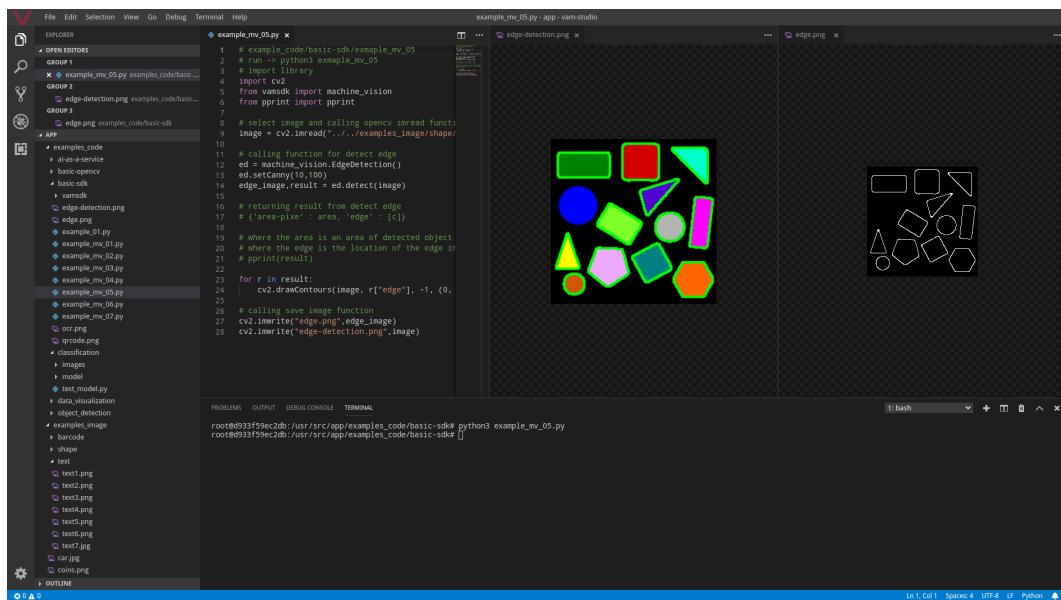
# calling save image function
cv2.imwrite("edge.png",edge_image)
cv2.imwrite("edge-detection.png",image)
```

3. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์เดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/basic-sdk
```

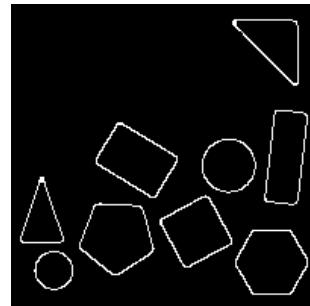
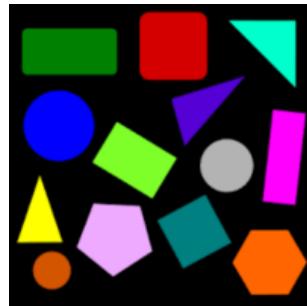
4. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_mv_05.py
```



รูปที่ 6.11 ผลการรันตัวอย่างโปรแกรมการตรวจจับเส้นขอบวัตถุบน VAM Studio





```
[{'area-pixel': 1315,
 'edge': [array([[174, 70],
                [[174, 72],
                 [[173, 73],
                  [[173, 80],
                   [[172, 81],
                    [[172, 89],
                     ...
                     [[185, 70]]], dtype=int32)]]]]
```

6.2.3 พังก์ชันการตรวจนับจำนวนวัตถุ (Object Counting)

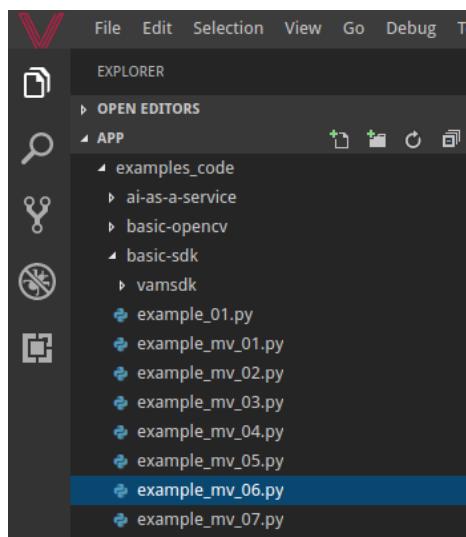
พังก์ชันการตรวจจำนวนนับวัตถุ ทำหน้าที่ในการนับข้อมูลที่ได้จากการประมวลผลและวิเคราะห์ภาพของพังก์ชันที่เกี่ยวกับการตรวจจับวัตถุ

พังก์ชันที่เกี่ยวข้อง

1. counting(data)
 - 1.1. data: รับข้อมูลพารามิเตอร์ประเภท Json หรือ List
 - 1.2. ส่งคืนข้อมูลตัวเลขจำนวนที่นับวัตถุได้จากการมิเตอร์ที่รับมา

ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจนับจำนวนวัตถุ

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > basic-sdk > example_mv_06.py ที่แถบเมนูด้านซ้าย



รูปที่ 6.12 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการตรวจจับจำนวนวัตถุ

ตัวอย่างโปรแกรมการตรวจจับจำนวนวัตถุ

```
# import library
import cv2
from vamsdk import machine_vision
from pprint import pprint

# select image and calling opencv imread function
image = cv2.imread("../examples_image/shape/shape1.jpg")

# calling function for detecting shape
sd = machine_vision.ShapeDetector()
sd.setThreshold(50)
shape_image,result = sd.detect(image)

# returning result from shape detection
# {'shape' : shape, 'bbox' : [x1, y1, x2, y2]}

# where shape is triangle, rectangle, pentagon and circle
# where (x1, y1) is topleft of boundingbox
# where (x2, y2) is bottomright of boundingbox
pprint(result)

# calling function for counting objects
```

```
oc = machine_vision.ObjectCounting()
num = oc.counting(result)
print("count : ", num)
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์เดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/basic-sdk
```

4. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_mv_06.py
```

The screenshot shows the VAM Studio interface. The code editor displays the Python script `example_mv_06.py`. The terminal window at the bottom shows the command `python3 example_mv_06.py` being run, and it outputs the detected shapes and their bounding boxes:

```
root@933f59ec2db:/usr/src/app/examples_code/basic-sdk# python3 example_mv_06.py
[{"bbox": [125, 290, 403, 363], "shape": "triangle"}, {"bbox": [648, 219, 745, 310], "shape": "circle"}, {"bbox": [317, 106, 394, 189], "shape": "rectangle"}]
root@933f59ec2db:/usr/src/app/examples_code/basic-sdk#
```

รูปที่ 6.13 ผลการรันตัวอย่างโปรแกรมการตรวจจับวัตถุบน VAM Studio

result	count
[{'bbox': [125, 290, 403, 363], 'shape': 'triangle'}, {'bbox': [648, 219, 745, 310], 'shape': 'circle'}, {'bbox': [317, 106, 394, 189], 'shape': 'rectangle'}]	5

```
{'bbox': [52, 52, 168, 151], 'shape': 'rectangle'},
{'bbox': [648, 46, 748, 140], 'shape': 'circle'}
]
```

6.3.4 พังก์ชันการหาคุณวัดถูกภายในภาพ (Color Pattern Matching)

พังก์ชันการหาคุณวัดถูกภายในภาพ ทำหน้าที่ประมวลผลและวิเคราะห์ข้อมูลภาพ เพื่อค้นหาคุณวัดถูกภายในภาพตามจำนวนที่ผู้ใช้กำหนดและสามารถนำค่าคุณวัดถูกที่ได้ไปใช้ในการตรวจสอบถูกภายในภาพเฉพาะของสีที่ต้องการ เช่น ตรวจสอบว่าสีของห่อที่เป็นสีฟ้าในสายพานลำเลียงวัสดุ เป็นต้น

พังก์ชันที่เกี่ยวข้อง

1. visualize_colors(image,num_color)

พังก์ชันนี้ใช้สำหรับแสดงผลการค้นหาคุณวัดถูกตามจำนวนที่ผู้ใช้กำหนด

1.1. image: รับพารามิเตอร์รูปภาพสี RGB

1.2. num_color: รับพารามิเตอร์เลขจำนวนกลุ่มสีที่ต้องการแสดงผล

1.3. ส่งคืนรูปภาพสีค้นหาเรียงตามค่าสีที่พบมากที่สุดเป็นลำดับที่ 0 ไปน้อยที่สุดเป็นลำดับที่ num_class เป็นข้อมูลประเภท numpy.ndarray

1.4. ส่งคืนข้อมูลประเภท list ของชุดค่าสี RGB ตามลำดับสีที่พบมากที่สุดไปน้อยที่สุด

2. detect(image,rgb)

2.1. image: รับพารามิเตอร์รูปภาพสี RGB

2.2. rgb: รับพารามิเตอร์ค่าสี [R,G,B]

2.3. ส่งคืนรูปภาพที่ผ่านการประมวลผลเป็นข้อมูลประเภท numpy.ndarray

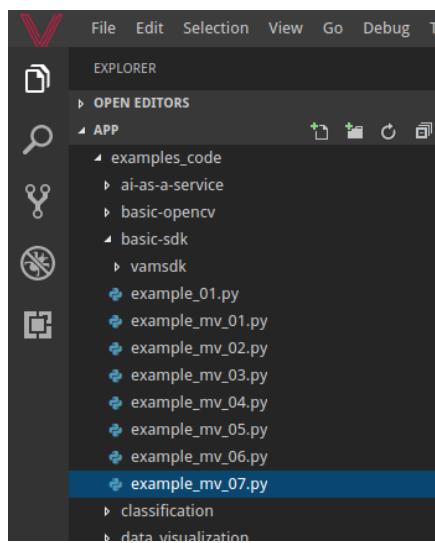
2.4. ส่งคืนข้อมูลประเภท Json ประกอบด้วย

2.4.1. bbox คือ จุดพิกัดคู่อันดับ (x,y) สำหรับวดาระบบตรวจวัดถูก เป็นข้อมูลประเภท List

ขั้นตอนการใช้ตัวอย่างโปรแกรมเพื่อหาคุณวัดถูกภายในภาพ

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))

2. คลิก example_code > basic-sdk > example_mv_07.py ที่ແນບເມນຸດ້ານໜ້າຍ



รูปที่ 6.14 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการหาคุณวัตถุภายในภาพ

ตัวอย่างโปรแกรมการหาคุณวัตถุภายในภาพ

```
# example_code/basic-sdk/exmaple_mv_07.py
# run -> python3 exmaple_mv_07.py

# import library
import cv2
from vamsdk import machine_vision
from pprint import pprint

# select image and calling opencv imread function
image = cv2.imread("../examples_image/shape/shape1.jpg")

# calling function for visualize color
cp = machine_vision.ColorPatternMatching()
color_image,result = cp.visualize_colors(image,num_color=10)

# returning result from visualize color
# [[R,G,B],[R,G,B]]
print(result)

# calling save image function
cv2.imwrite("colors.png",color_image)
```

```
# calling function for detect color
color_detection, bbox = cp.detect(image,result[4])

# returning result from shape detection
# {'bbox' : [x1, y1, x2, y2]}

# where (x1, y1) is topleft of boundingbox
# where (x2, y2) is bottomright of boundingbox
print(bbox)

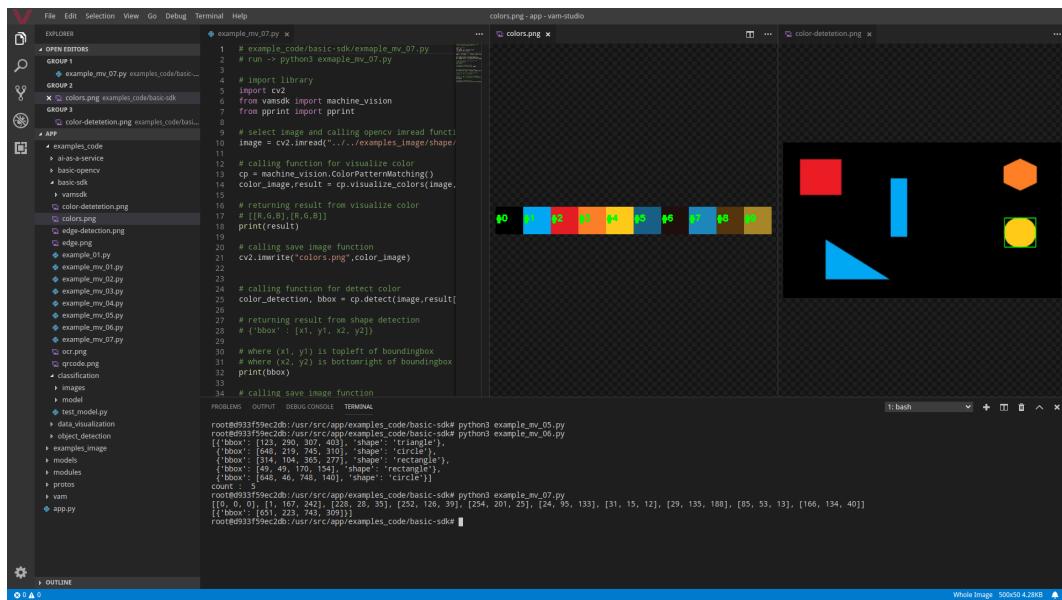
# calling save image function
cv2.imwrite("color-detetetion.png",color_detection)
```

3. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อเข้าถึงโฟลเดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

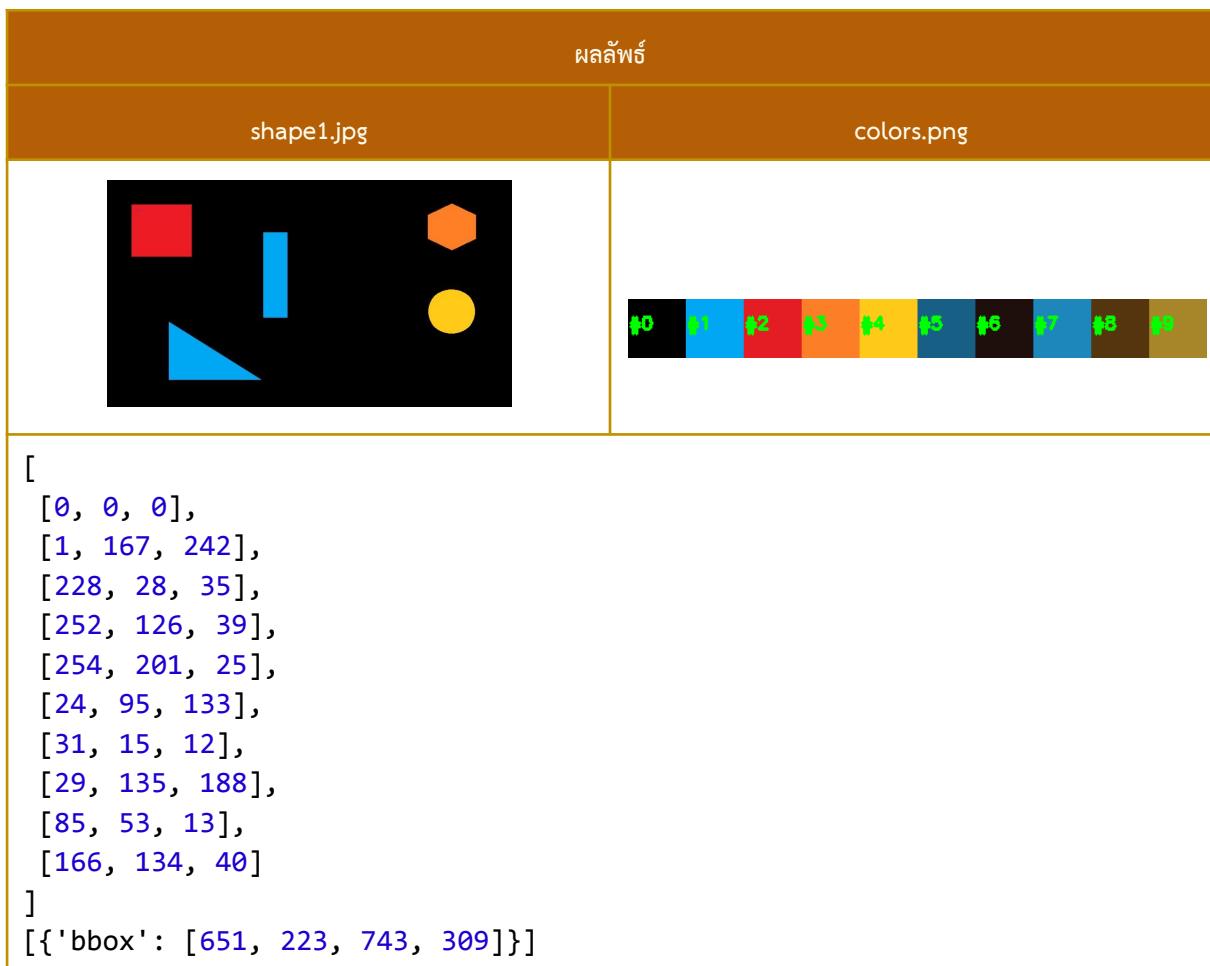
```
cd example_code/basic-sdk
```

4. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_mv_07.py
```



รูปที่ 6.15 ผลการรันตัวอย่างโปรแกรมการหาคุณวิสัตถุภายในภาพบน VAM Studio



ใจทราย

1. จากตัวอย่างการตรวจจับเส้นขอบของวัตถุ ให้ผู้เรียนทดลองปรับหาค่า lower_threshold, upper_threshold ให้โปรแกรมสามารถตรวจจับเส้นขอบของวัตถุได้มากสุด
 2. จากตัวอย่างการตรวจจับวัตถุ ให้ผู้เรียนทำการนับวัตถุที่เป็นผลลัพธ์จากการใช้ฟังก์ชันการตรวจจับเส้นขอบของวัตถุ

สรุปและบันทึกผล

LAB 7 | VAM Industrial Camera

Industrial Camera

กล้องอุตสาหกรรม (Industrial Camera) เป็นกล้องชนิดพิเศษที่ปรับให้ใช้ทำงานในสภาพแวดล้อมที่ไม่อ่อนไหว เช่น อุณหภูมิสูง ความดัน และการสั่นสะเทือน ใช้ติดตามและเพื่อควบคุมจรวจผลิตบนสายพานลำเลียง ตรวจจับชิ้นส่วนขนาดเล็กพิเศษ ฯลฯ

คุณสมบัติหลักของกล้องอุตสาหกรรม คือ

1. ความน่าเชื่อถือ
2. ความแม่นยำในการจับรูปภาพ
3. ความเร็วในการตรวจจับรูปภาพ
4. หลักการทำงานที่เข้าถึงง่าย



รูปที่ 7.1 กล้องอุตสาหกรรม (Industrial Camera)

LAB 7.1 การติดตั้งและประกอบชุดทดลองของกล้องอุตสาหกรรม

วัตถุประสงค์

- เพื่อเรียนรู้วิธีการติดตั้งและประกอบชุดทดลอง VAM Platform สำหรับ Industrial Camera
- เพื่อเรียนรู้วิธีการตีงภาพจาก Industrial Camera

7.1.1 วิธีที่ 1 การติดตั้งและประกอบชุดทดลองเข้ากับอุปกรณ์เสริม

วิธีนี้ผู้เรียนสามารถใช้งานแพลตฟอร์มผ่านอุปกรณ์เสริมที่ต่อเข้ากับเครื่องประมวลผลได้เพียงเครื่องเดียว ซึ่งวิธีการติดตั้งจะคล้ายกับ LAB 1.1.1 แต่จะเปลี่ยนจากการใช้ IP Camera เป็น Industrial Camera

อุปกรณ์ที่เกี่ยวข้อง

1. อุปกรณ์หลัก

1.1. Intel NUC 11 enthusiast mini pc kit	1	เครื่อง
1.2. LITEON AC Adapter	1	ตัว
1.2.1. Input AC 100-240V ~ 50-60Hz 3.5A		
1.2.2. Output 19.5V 11.8A 230.0W		

1.3. POE Switch: D-Link DGS-1210-10P 8-Port	1	เครื่อง
1.4. LEADER ELETRONICS Power supply	1	ตัว

1.4.1. Input 100-240V ~ 50/60Hz 1.2A		
1.4.2. Output 54.0V 1.67A 90.0W		

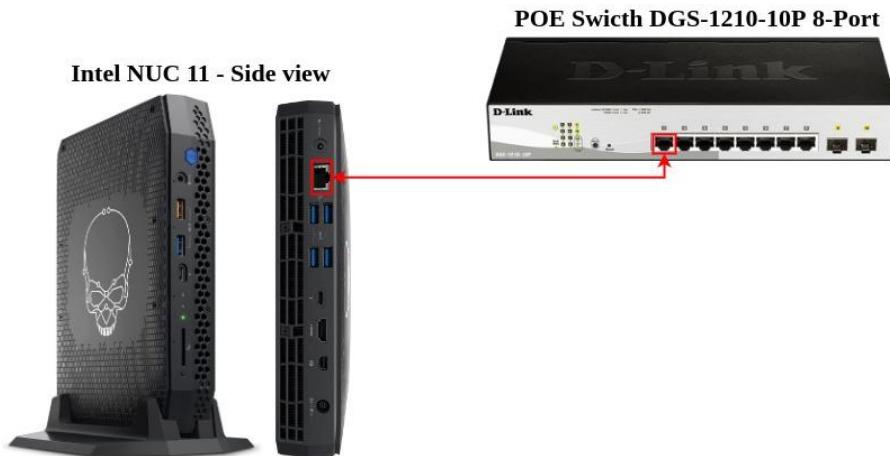
1.5. Industrial Camera: Imaging source DFK 33GX174E	1	ตัว
1.6. สาย LAN	2	เส้น

2. อุปกรณ์เสริม

2.1. Azus ZenScreen	1	จอ
2.2. สาย USB Display type-C to Type-C	1	เส้น
2.3. คีย์บอร์ด	1	ตัว
2.4. เม้าส์	1	ตัว

ขั้นตอนการติดตั้งและประกอบชุดการทดลอง

1. เชื่อมต่อเครื่อง NUC และ POE Switch ด้วยสาย LAN



รูปที่ 7.2 การเชื่อมต่อเครื่อง NUC และ POE Switch

2. เชื่อมต่อกล้องและ POE Switch ด้วยสาย LAN



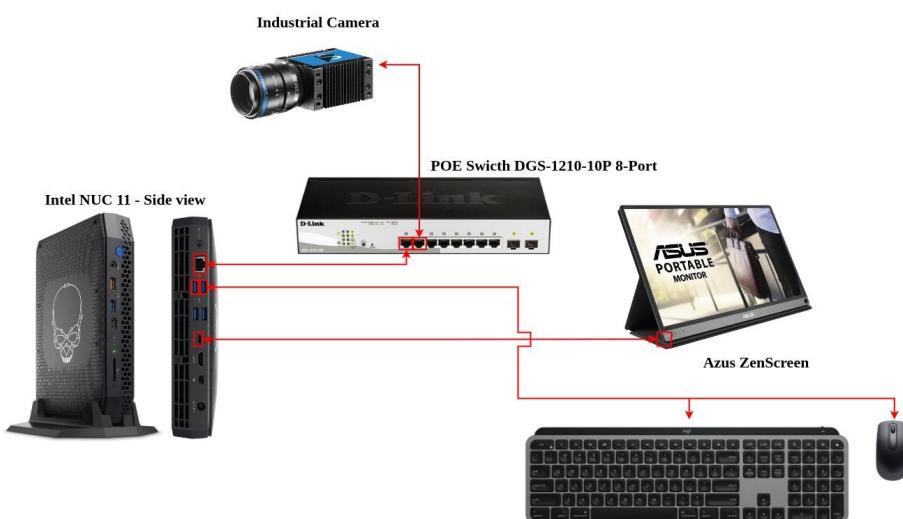
รูปที่ 7.3 การเชื่อมต่อกล้องและ POE Switch

3. เชื่อมต่อเครื่อง NUC และจอ ZenScreen ด้วยสาย USB Display type-C to Type-C



รูปที่ 7.4 การเชื่อมต่อเครื่อง NUC และจอ ZenScreen

4. เชื่อมต่อเครื่อง NUC คีย์บอร์ดและเมาส์



รูปที่ 7.5 การเชื่อมต่อเครื่อง NUC คีย์บอร์ดและเมาส์

5. เชื่อมต่อแหล่งจ่ายไฟจาก LEADER ELETRONICS Power supply เข้า POE Switch



รูปที่ 7.6 การเชื่อมต่อแหล่งจ่ายไฟจาก Power supply เข้า POE Switch

6. เชื่อมต่อแหล่งจ่ายไฟจาก LITEON AC Adapter เข้าเครื่อง NUC



รูปที่ 7.7 การเชื่อมต่อแหล่งจ่ายไฟจาก AC Adapter เข้าเครื่อง NUC

7. ตรวจสอบการเชื่อมต่ออุปกรณ์ให้เรียบร้อย และจึงกดปุ่มเปิดเครื่อง NCU และปุ่มเปิดจอ ZenScreen



รูปที่ 7.8 ปุ่มเปิดเครื่อง NUC และปุ่มเปิดจอ ZenScreen

8. เมื่อเปิดเครื่อง NUC ให้ผู้เรียนทำการกรอกรหัส “eslab” เพื่อเข้าสู่ระบบของเครื่อง NUC

7.1.2 วิธีที่ 2 การติดตั้งและประกอบชุดทดลองเข้ากับเครื่องของผู้เรียน

วิธีนี้จะมีขั้นตอนการติดตั้งคล้ายกับวิธีที่ 1 แตกต่างกันที่เครื่องประมวลผลไม่ต้องต่อเข้ากับอุปกรณ์เสริม แต่จะเชื่อมต่อเข้ากับคอมพิวเตอร์ของผู้เรียน ทำให้ผู้เรียนสามารถใช้งานแพลตฟอร์มบนเครื่องของผู้เรียนได้โดยเครื่องพร้อมกัน ซึ่งวิธีการติดตั้งจะคล้ายกับ LAB 1.1.2 แต่จะเปลี่ยนจากการใช้ IP Camera เป็น Industrial Camera

อุปกรณ์ที่เกี่ยวข้อง

1. อุปกรณ์หลัก

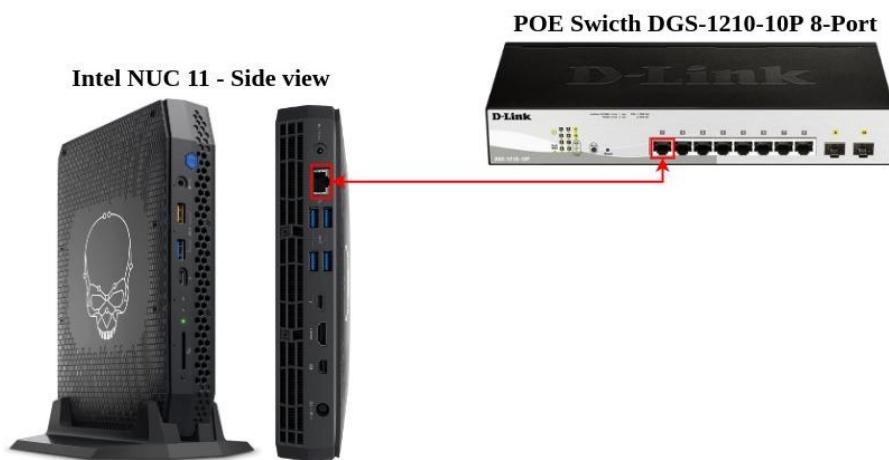
1.1. Intel NUC 11 enthusiast mini pc kit	1	เครื่อง
1.2. LITEON AC Adapter	1	ตัว
1.2.1. Input AC 100-240V ~ 50-60Hz 3.5A		
1.2.2. Output 19.5V 11.8A 230.0W		
1.3. POE Switch: D-Link DGS-1210-10P 8-Port	1	เครื่อง
1.4. LEADER ELETROONICS Power supply	1	ตัว
1.4.1. Input 100-240V ~ 50/60Hz 1.2A		
1.4.2. Output 54.0V 1.67A 90.0W		
1.5. Industrial Camera: Imaging source DFK 33GX174E	1	ตัว

1.6. สาย LAN

- | | | |
|---|---|--------------|
| 1.6.1. สำหรับต่อเข้าอุปกรณ์หลัก | 2 | เส้น |
| 1.6.2. สำหรับต่อเข้าคอมพิวเตอร์ของผู้เรียน | 1 | เส้น/เครื่อง |
| 2. คอมพิวเตอร์ของผู้เรียน | | |
| 2.1. ระบบปฏิบัติการ os: debian 10 ขึ้นไป หรือ ubuntu 18.04 ขึ้นไป | | |
| 2.2. ติดตั้ง Python Version 3.6 ขึ้นไป | | |

ขั้นตอนการติดตั้งและประกอบชุดการทดลอง

- เชื่อมต่อเครื่อง NUC และ POE Switch ด้วยสาย LAN



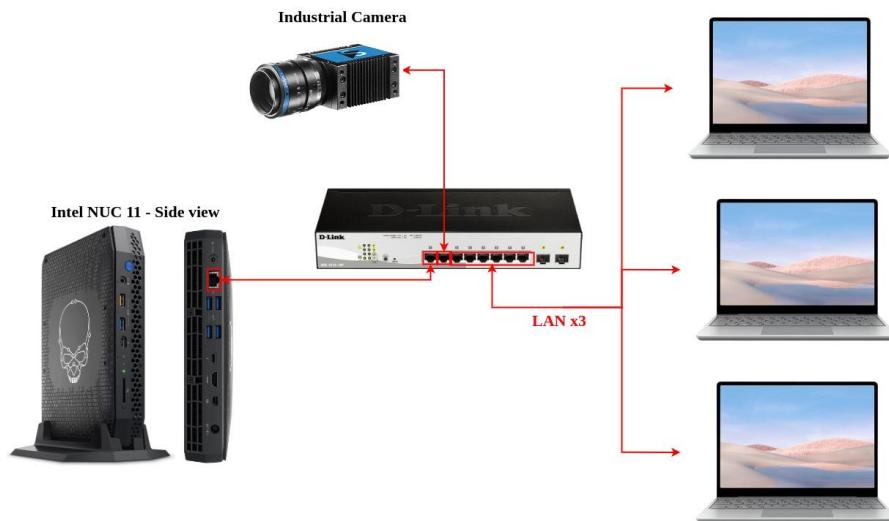
รูปที่ 7.9 การเชื่อมต่อเครื่อง NUC และ POE Switch

2. เชื่อมต่อกล้องและ POE Switch ด้วยสาย LAN



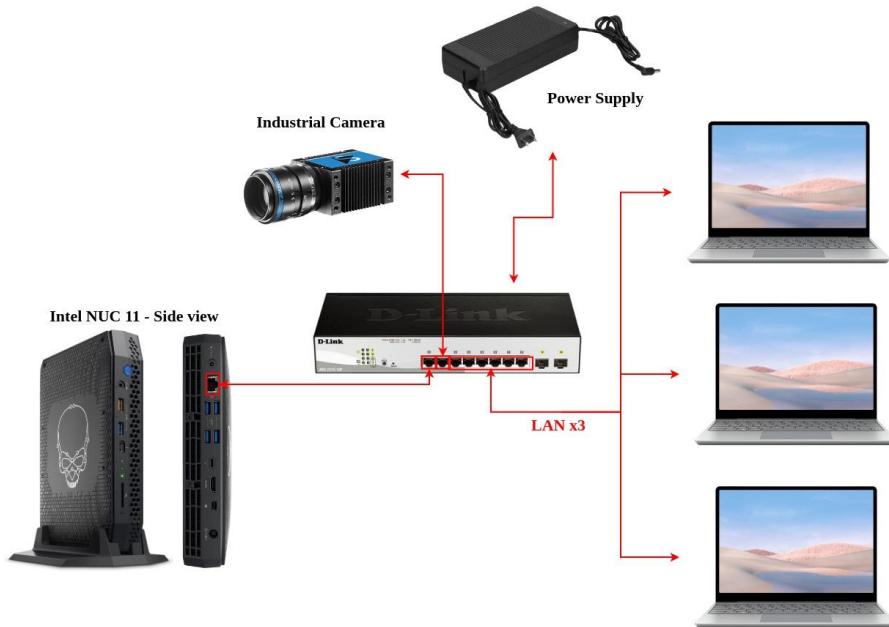
รูปที่ 7.10 การเชื่อมต่อกล้องและ POE Switch

3. เชื่อมต่อคอมพิวเตอร์ของผู้เรียนและ POE Switch ด้วยสาย LAN



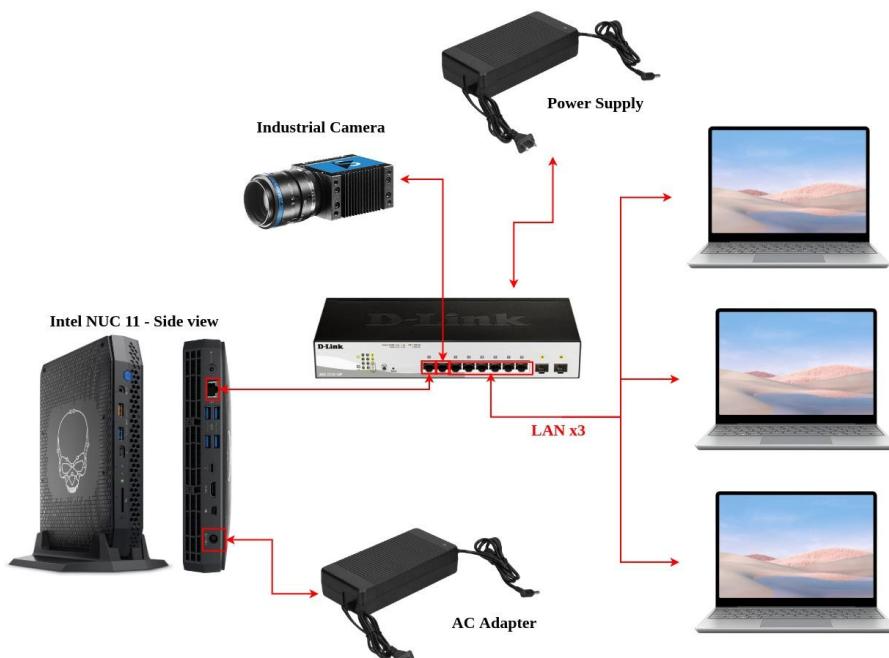
รูปที่ 7.11 การเชื่อมต่อคอมพิวเตอร์ของผู้เรียนและ POE Switch

4. เชื่อมต่อแหล่งจ่ายไฟจาก LEADER ELETRONICS Power supply เข้า POE Switch



รูปที่ 7.12 การเชื่อมต่อแหล่งจ่ายไฟจาก Power supply เข้า POE Switch

5. เชื่อมต่อแหล่งจ่ายไฟจาก LITEON AC Adapter เข้าเครื่อง NUC



รูปที่ 7.13 การเชื่อมต่อแหล่งจ่ายไฟจาก AC Adapter เข้าเครื่อง NUC

6. ตรวจสอบการเชื่อมต่ออุปกรณ์ให้เรียบร้อย และวิ่งกดปุ่มเปิดเครื่อง NUC



รูปที่ 7.14 ปุ่มเปิดเครื่อง NUC

7. ตั้งค่าเครือข่ายอินเทอร์เน็ตภายในเครื่องคอมพิวเตอร์ของผู้เรียน ([หัวข้อ 1.1.2](#))

หมายเหตุ ไม่สามารถนำ Industrial Camera เข้าไปใช้ใน VAM Platform แต่สามารถใช้ในเครื่อง NUC ได้

LAB 7.2 การดึงข้อมูลภาพจากกล้องอุตสาหกรรม

วัตถุประสงค์

1. เพื่อเรียนรู้วิธีการดึงข้อมูลภาพจาก Industrial Camera มาใช้ในการประมวลผลและวิเคราะห์ข้อมูล
2. เพื่อให้ผู้เรียนสามารถดึงภาพจาก Industrial Camera มาแสดงผลบนคอมพิวเตอร์ของผู้เรียนหรือเครื่อง NUC ได้

อุปกรณ์ที่เกี่ยวข้อง

1. ชุดการทดลอง Industrial Camera

7.2.1 ขั้นตอนการใช้ตัวอย่างโปรแกรมการดึงภาพจากกล้องอุตสาหกรรม

1. เปิด terminal ภายในเครื่องของผู้เรียน เพื่อติดตั้ง package ที่จำเป็น ดังนี้

```
sudo apt install python3-opencv python3-numpy
```

2. ดาวน์โหลดตัวอย่างโปรแกรมจาก <https://gist.github.com/thekitp/a80842d8e80417982a1eeadae58dbda8>

- 2.1. วิธีที่ 1 ดาวน์โหลดไฟล์ ZIP และทำการแตกไฟล์ในเครื่องของผู้เรียน
- 2.2. วิธีที่ 2 ดาวน์โหลดไฟล์ด้วยการใช้คำสั่ง git clone

```
git --version  
# git version 2.36.0  
git clone https://gist.github.com/a80842d8e80417982a1eeadae58dbda8.git
```

หมายเหตุ หากไม่พบเวอร์ชันของ git ให้ทำการติดตั้ง git ด้วยคำสั่งต่อไปนี้

```
sudo apt install git
```

ตัวอย่างโปรแกรมการดึงข้อมูลภาพจากกล้องอุตสาหกรรม

```
import numpy as np  
import cv2 as cv
```

```

cap = cv.VideoCapture("aravissrc ! videoconvert !
video/x-raw,format=BGR ! appsink", cv.CAP_GSTREAMER)

if not cap.isOpened():
    print("Cannot open camera")
    exit()
while True:
    # Capture frame-by-frame
    ret, frame = cap.read() # frame is image from industrial camera

    # ----can insert other code here----

    # -----      -----
# if frame is read correctly ret is True
if not ret:
    print("Can't receive frame (stream end?). Exiting ...")
    break

# Display the resulting frame
cv.imshow('frame', frame)
if cv.waitKey(1) == ord('q'):
    break

# When everything done, release the capture
cap.release()
cv.destroyAllWindows()

```

3. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์เดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```

cd <path>/BUU-Industrial-Camera
# example cd /home/user/BUU-Industrial-Camera

```

4. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 industrial-video.py
```

```
gst-launch-1.0 -v /GST_PLUGIN_PATH=/vavstack@vamstack-desktop:/hub/vamstack.com/harbor
GST_PLUGIN_PATH=/usr/local/lib/x86_64-linux-gnu/gst-launch-1.0 -v aravissrc ! videoconvert ! autovideosink

hostPad: caps = video/x-raw, width=(int)480, height=(int)480, framerate=(fraction)5/1, format=(string)YUV
sink: caps = video/x-raw, width=(int)480, height=(int)480, framerate=(fraction)5/1, format=(string)VUV
!autovideosink#0/GstVideoSink:autovideosink#0: Pipeline construction is invalid, please add queues.

peline:pipeline0/GstAutoVideoSink:autovideosink#0: actual-sink-xvimage:
of 0:00:00.015000000, add enough queues to buffer 0:00:00.015000000 additional data. Shortening processing latency to 0:00:00.000000000.

LUGIN_PATH=/usr/local/lib/x86_64-linux-gnu gst-launch-1.0 -v aravissrc ! videoconvert ! autovideosink

video/x-raw, width=(int)480, height=(int)480, framerate=(fraction)5/1, format=(string)YUV
sink: caps = video/x-raw, width=(int)480, height=(int)480, framerate=(fraction)5/1, format=(string)VUV
!autovideosink#0/GstVideoSink:autovideosink#0: actual-sink-xvimage.GstPad: caps = video/x-raw, width=(int)480, height=(int)480, framerate=(fraction)5/1, format=(string)VUV
!autovideosink#0/GstVideoSink:autovideosink#0: Pipeline construction is invalid, please add queues.

databasesink.c(1209): gst_base_sink_query_latency (): /GstPipeline:pipeline0/GstAutoVideoSink:autovideosink#0/GstVideoSink:autovideosink#0: actual-sink-xvimage:
st enough buffering available for the processing deadline of 0:00:00.015000000, add enough queues to buffer 0:00:00.015000000 additional data. Shortening processing latency to 0:00:00.000000000.
Chandler interrupt.

interrupt: Stopping pipeline ...
execution ended at timestamp 50.644232179
pipeline state changed to PAUSED
etting pipeline to READY ...
etting pipeline to NULL ...
freeing pipeline ...

LD_LIBRARY_PATH=/usr/local/lib/x86_64-linux-gnu /GST_PLUGIN_PATH=/usr/local/lib/x86_64-linux-gnu gst-launch-1.0 -v aravissrc ! videoconvert ! autovideosink
etting pipeline to PAUSED ...
etting pipeline to PLAYING ...
etting pipeline to PAUSED ...
etting pipeline to READY ...
etting pipeline to NULL ...
freeing pipeline ...

/GstPipeline:pipeline0/GstAravis:aravis.GstPadsrc: caps = video/x-raw, width=(int)480, height=(int)480, framerate=(fraction)5/1, format=(string)VUV
/GstPipeline:pipeline0/GstVideoconvert:deconv0.GstPadsrc: caps = video/x-raw, width=(int)480, height=(int)480, framerate=(fraction)5/1, format=(string)VUV
/GstPipeline:pipeline0/GstAutoVideoSink:autovideosink#0.GstChestProxyPadproxypad0: caps = video/x-raw, width=(int)480, height=(int)480, framerate=(fraction)5/1, format=(string)VUV
/GstPipeline:pipeline0/GstAutoVideoSink:autovideosink#0/GstVideoSink:autovideosink#0:actual-sink-xvimage.GstPad: caps = video/x-raw, width=(int)480, height=(int)480, framerate=(fraction)5/1, format=(string)VUV
/GstPipeline:pipeline0/GstAutoVideoSink:autovideosink#0.GstChestPad: caps = video/x-raw, width=(int)480, height=(int)480, framerate=(fraction)5/1, format=(string)VUV
/GstPipeline:pipeline0/GstVideoconvert:deconv0.GstPadsrc: caps = video/x-raw, width=(int)480, height=(int)480, framerate=(fraction)5/1, format=(string)VUV
/GstPipeline:pipeline0/GstAutoVideoSink:autovideosink#0/GstVideoSink:autovideosink#0:actual-sink-xvimage.GstPad: caps = video/x-raw, width=(int)480, height=(int)480, framerate=(fraction)5/1, format=(string)VUV
/GstPipeline:pipeline0/GstAutoVideoSink:autovideosink#0/GstVideoSink:autovideosink#0: Pipeline construction is invalid, please add queues.

databasesink.c(1209): gst_base_sink_query_latency (): /GstPipeline:pipeline0/GstAutoVideoSink:autovideosink#0/GstVideoSink:autovideosink#0: actual-sink-xvimage:
of 0:00:00.015000000, add enough queues to buffer 0:00:00.015000000 additional data. Shortening processing latency to 0:00:00.000000000.
```

รูปที่ 7.15 ผลลัพธ์การดึงภาพจาก Industry Camera

5. เมื่อต้องการหยุดการทำงานของโปรแกรมให้พิริยนกดปุ่ม “q”

ໂຄຫຍໍ

- ให้ผู้เรียนเพิ่มชุดโค้ดลงไปในตัวอย่างโปรแกรมจากกล้อง industrial camera เพื่อทำการปรับภาพจากกล้องให้เป็นภาพใบหน้า โดยอ้างอิงจากตัวอย่างโปรแกรมการปรับภาพใบหน้า

สรุปและบันทึกผล

ชุดทดลองสำหรับเรียนรู้ทางด้านการประมวลผลและวิเคราะห์สัญญาณภาพและวิดีโอด้วยเทคโนโลยีปัญญาประดิษฐ์

LAB 8 | Object Classification

วัตถุประสงค์

- เพื่อเรียนรู้วิธีการใช้งานโมเดลสำหรับการตรวจจับวัตถุ (Object Detection) บน VAM Studio

อุปกรณ์ที่เกี่ยวข้อง

- ชุดการทดลอง VAM Platform

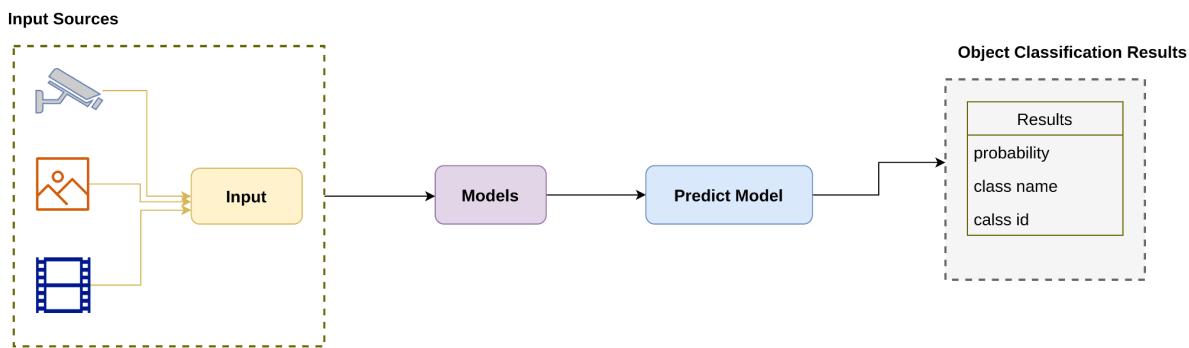
Object Classification

การจำแนกประเภทของวัตถุที่อยู่ในภาพด้วยการวิเคราะห์ข้อมูลเชิงปัญญาประดิษฐ์ (AI) เพื่อแยกข้อมูลจุดภาพทั้งหมดที่ประกอบเป็นพื้นที่ศึกษาออกเป็นกลุ่มย่อย ตามคุณลักษณะที่มีความคล้ายคลึงกันและแตกต่างกันของแต่ละชุดข้อมูลภาพ



รูปที่ 8.1 ภาพตัวอย่าง Object Classification

Diagram Object Classification



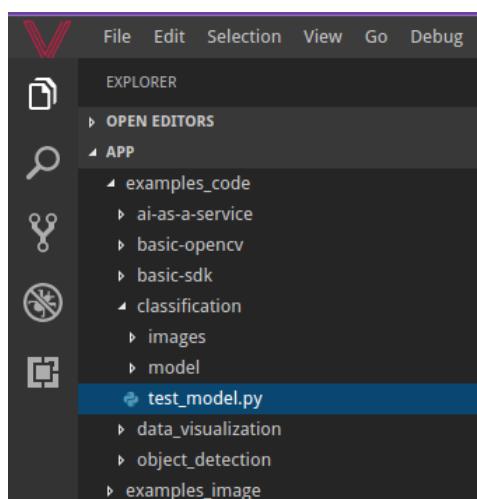
รูปที่ 8.2 Diagram Object Classification

จาก Diagram Object Classification จะวิธีการตามขั้นตอนดังนี้

1. รับข้อมูลนำเข้า เช่น ภาพถ่าย วิดีโอ กล้องวงจรปิด
2. นำข้อมูลนำเข้ามาประมวลผลด้วยโมเดลปัญญาประดิษฐ์ (AI) โดยจะแบ่งเป็น 2 ขั้นตอนย่อยดังนี้
 - i. ดาวน์โหลดโมเดลปัญญาประดิษฐ์
 - ii. นำโมเดลที่ถูกดาวน์โหลดมาประมวลผลข้อมูลร่วมกับข้อมูลนำเข้า (ภาพ วิดีโอ กล้องวงจรปิด)
3. ผลลัพธ์การประมวลผลข้อมูลจาก Object Classification

8.1 ขั้นตอนการใช้ตัวอย่างโปรแกรมการจำแนกประเภทวัตถุ

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > classification > test_model.py ที่แสดงเมนูด้านซ้าย



รูปที่ 8.3 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการจำแนกประเภทวัตถุ

ตัวอย่างโปรแกรมการจำแนกประเภทวัตถุ

```
# import the necessary packages
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
import numpy as np
import imutils
import cv2

IMAGE_PATH = "images/classification/cat_01.jpg"
MODEL_PARTH = "model/cat&dog.model"
LB_PATH = "model/labels.txt"

# load the image
image = cv2.imread(IMAGE_PATH)
orig = image.copy()

# pre-process the image for classification
image = cv2.resize(image, (96, 96))
image = image.astype("float") / 255.0
image = img_to_array(image)
image = np.expand_dims(image, axis=0)

# load the label
model = load_model(MODEL_PARTH)
labels = open(LB_PATH, "r")

# classify the input image
proba = model.predict(image)[0]
idx = np.argmax(proba)
label = labels.read().split(",")[idx]

# build the label and draw the label on the image
label = "{}: {:.2f}%".format(label, proba[idx] * 100)
output = imutils.resize(orig, width=400)
cv2.putText(output, label, (10, 25), cv2.FONT_HERSHEY_SIMPLEX,
            0.7, (0, 255, 0), 2)
print("Output is {}".format(label))

# save the output image
```

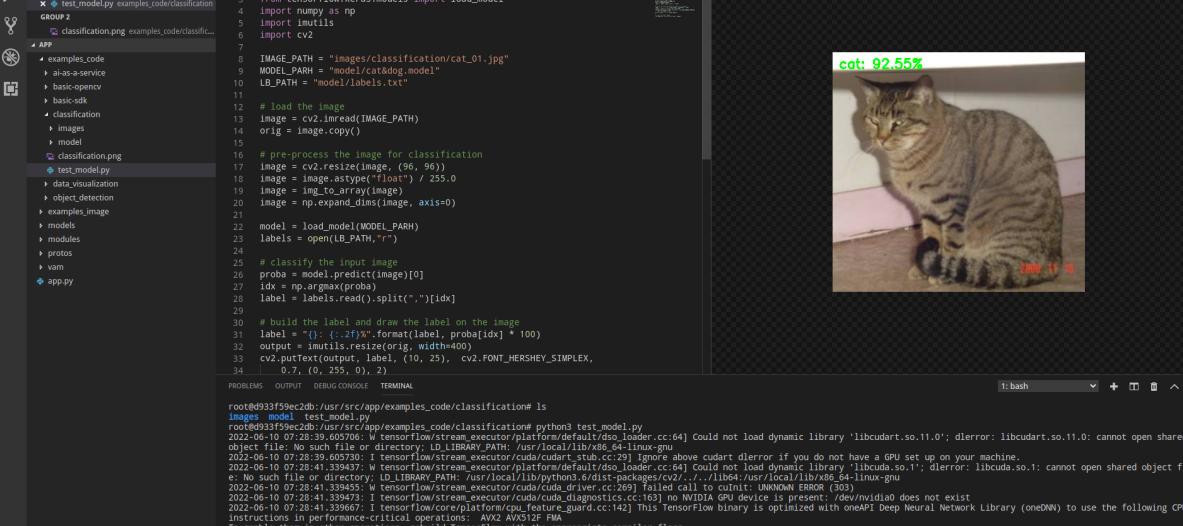
```
cv2.imwrite("classification.png", output)
```

3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงโฟลเดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/classification
```

4. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 test_model.py
```



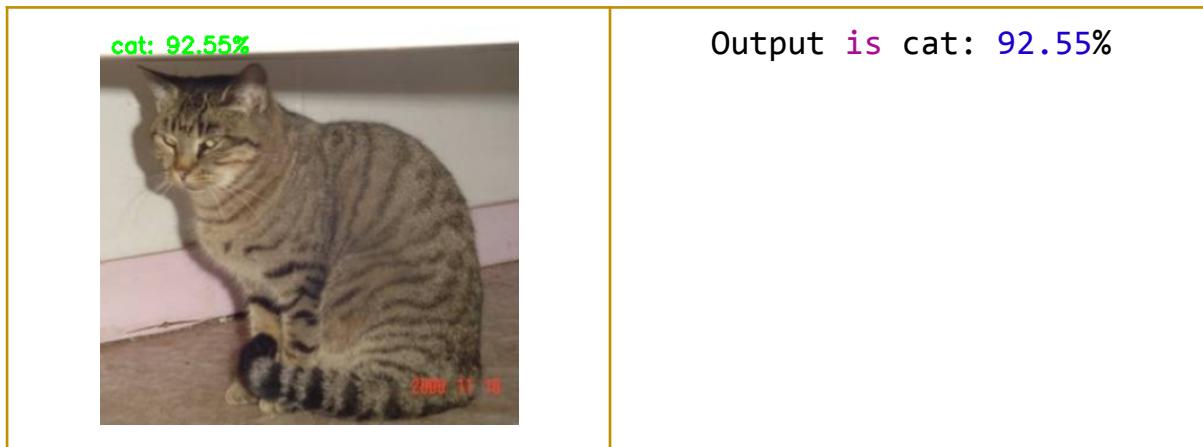
The screenshot shows a terminal window with several tabs open. The active tab is titled "classification.png" and displays a photograph of a brown tabby cat sitting on a surface. Overlaid on the image is the text "cat: 92.55%". Below the image, the terminal output shows the command "python3 test_model.py" and its execution results, including various error messages related to TensorFlow library loading and optimization passes.

```
test_model.py - app - vam - studio
EXPLORER GROUP 1 GROUP 2 APP
OPEN EDITORS
GROUP 1
x test_model.py examples_code/classification
classification.png examples_code/classification
GROUP 2
classification.png examples_code/classification
APP
examples_code
ai-as-a-service
basic-opencv
basic-sdk
classification
images
model
classification.png
test_model.py
data_visualization
object_detection
examples_image
models
modules
protos
vam
app.py

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
root@93:5f9ec2b:/usr/src/app/examples_code/classification# ls
images model test_model.py
root@93:5f9ec2b:/usr/src/app/examples_code/classification# python3 test_model.py
2022-06-10 07:28:41: I tensorflow/core/platform/cpu_feature_guard.cc:14] Could not load dynamic library "libcudart.so.11.0"; dlerror: libcudart.so.11.0: cannot open shared object file: no such file or directory
2022-06-10 07:28:41: I tensorflow/stream_executor/cuda/cuda_stub.cc:20] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
2022-06-10 07:28:41:339453: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library "/lib/x86_64-linux-gnu/libcudabu.so.1"; dlerror: libcudabu.so.1: cannot open shared object file
2022-06-10 07:28:41:339455: W tensorflow/stream_executor/cuda/cuda_driver.cc:269] Failed call to cuInit: UNKNOWN ERROR (303)
2022-06-10 07:28:41:339473: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:163] no NVIDIA GPU device is present: /dev/nvidia0 does not exist
2022-06-10 07:28:41:339602: W tensorflow/stream_executor/core/plugins/mkl/mkl.h:142] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX2 AVX512F FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2022-06-10 07:28:41:706982: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185] None of the MLIR Optimization Passes are enabled (registered 2)
Output in cat: 92.55%
root@93:5f9ec2b:/usr/src/app/examples_code/classification# [
```

รูปที่ 8.4 ผลการรันตัวอย่างโปรแกรมการจำแนกประเภทวัตถุน ของ VAM Studio





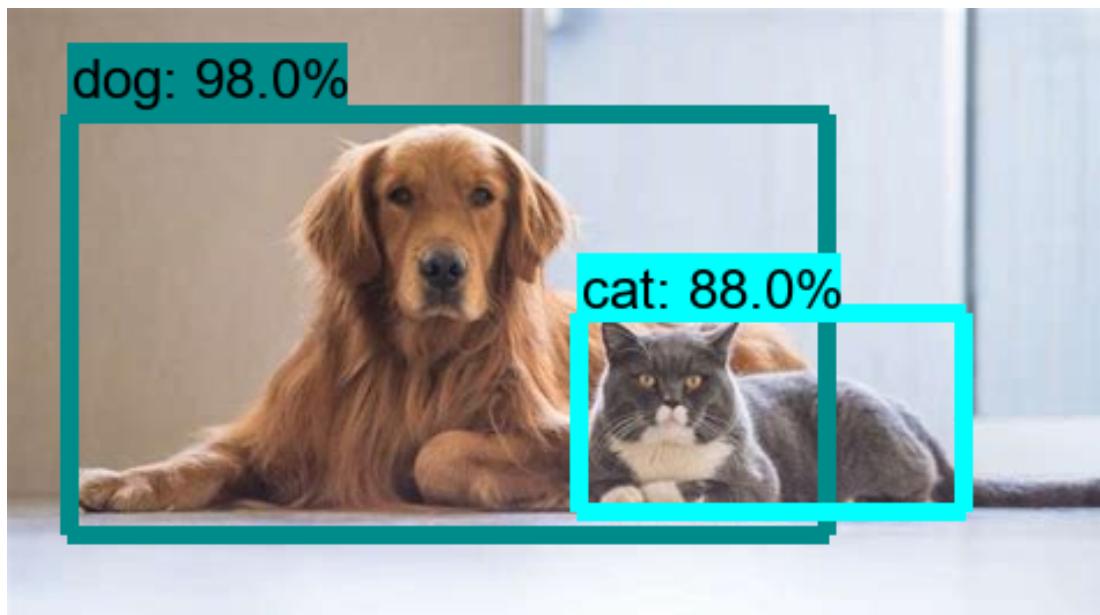
LAB 9 | Object Detection

วัตถุประสงค์

- เพื่อทำความรู้จักการเรียกใช้งานโมเดลสำหรับการตรวจจับวัตถุบน VAM Studio

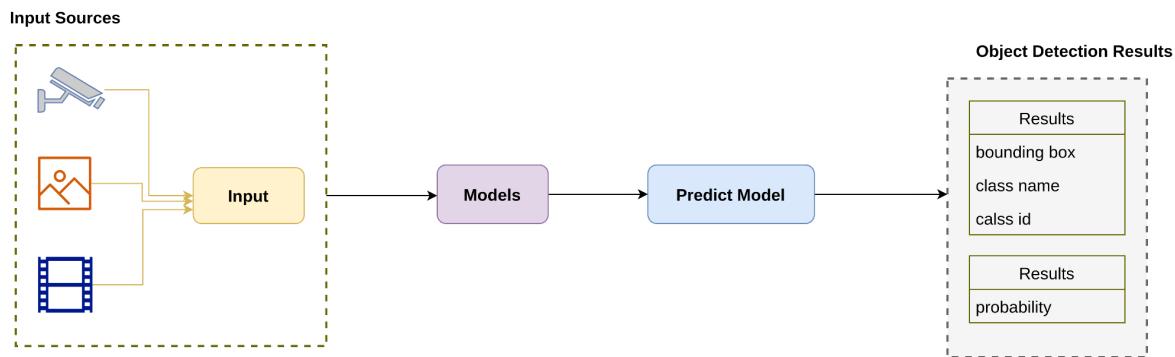
Object Detection

"เทคโนโลยีตรวจจับวัตถุ" (Object detection) คือ หนึ่งในฟีเจอร์หลักของปัญญาประดิษฐ์ (AI: Artificial Intelligence) ที่ใช้กับรูปภาพจากวิดีโอรวมถึงรูปภาพจากกล้องวงจรปิด สามารถนำรูปภาพหล่านั้นมาค้นหาและระบุตำแหน่งของวัตถุต่าง ๆ โดยใช้ AI มาวิเคราะห์ข้อมูลจากการมองเห็นของคอมพิวเตอร์ (Computer Vision) และการประมวลผลภาพ (Image Processing) เพื่อตรวจจับวัตถุที่อยู่ในรูปภาพหรือวิดีโอ เช่น มนุษย์ สัตว์ สิ่งของ รถยนต์ อาคาร และวัตถุอื่น ๆ



รูปที่ 9.1 ภาพตัวอย่าง Object Detection

Diagram Object Detection



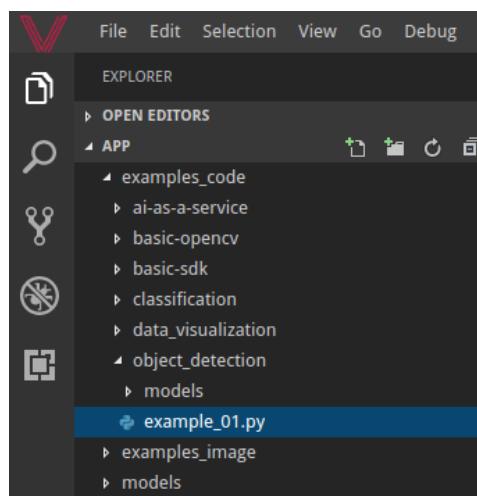
รูปที่ 9.2 Diagram Object Detection

จาก Diagram Object Detection จะวิธีการตามขั้นตอนดังนี้

1. รับข้อมูลนำเข้า เช่น ภาพถ่าย วิดีโอ กล้องวงจรปิด
2. นำข้อมูลนำเข้ามาประมวลผลด้วยโมเดลปัญญาประดิษฐ์ (AI) โดยจะแบ่งเป็น 2 ขั้นตอนย่อยดังนี้
 - i. ดาวน์โหลดโมเดลปัญญาประดิษฐ์
 - ii. นำโมเดลที่ถูกดาวน์โหลดมาประมวลผลข้อมูลร่วมกับข้อมูลนำเข้า (ภาพ วิดีโอ กล้องวงจรปิด)
3. ผลลัพธ์การประมวลผลข้อมูลจาก object detection

9.1 ขั้นตอนการใช้ตัวอย่างโปรแกรมการตรวจจับวัตถุ

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > object_detection > example_01.py ที่ແຄบເມັນດ້ານຊ້າຍ



รูปที่ 9.3 การเข้าถึงไฟล์ตัวอย่างโปรแกรมการตรวจจับวัตถุ

ตัวอย่างโปรแกรมการตรวจจับวัตถุ

```
# import library
import cv2
import numpy as np

# parameter for config models
thres = 0.45
nms_threshold = 0.2

#image path
IMAGE_PATH = "../examples_image/car.jpg"
#insert classname
classNames= []
classFile = "models/coco.names"
with open(classFile, "rt") as f:
    classNames = f.read().rstrip("\n").split("\n")

#model and config path
configPath = "models/ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt"
weightsPath = "models/frozen_inference_graph.pb"

#load and configuration model
net = cv2.dnn_DetectionModel(weightsPath, configPath)
net.setInputSize(320,320)
net.setInputScale(1.0/ 127.5)
net.setInputMean((127.5, 127.5, 127.5))
net.setInputSwapRB(True)

img = cv2.imread(IMAGE_PATH)

#inference models
classIds, confs, bbox = net.detect(img, confThreshold=thres)
bbox = list(bbox)
confs = list(np.array(confs).reshape(1,-1)[0])
confs = list(map(float, confs))

# call non maximum suppression
```

```

indices = cv2.dnn.NMSBoxes(bbox, confs, thres, nms_threshold)

#overlay image
for i in indices:
#    i = i[0]
    # print(classIds[i])
    box = bbox[i]
    x,y,w,h = box[0],box[1],box[2],box[3]
    # print(x, y, w, h)
    cv2.rectangle(img, (x,y),(x+w,h+y), color=(0, 255, 0),
thickness=2)

cv2.putText(img,classNames[classIds[i]-1].upper(),(box[0]-5,box[1]-5),
cv2.FONT_HERSHEY_COMPLEX, 0.5, (0,255,255), 2)

# save result
cv2.imwrite("test.jpg", img)

```

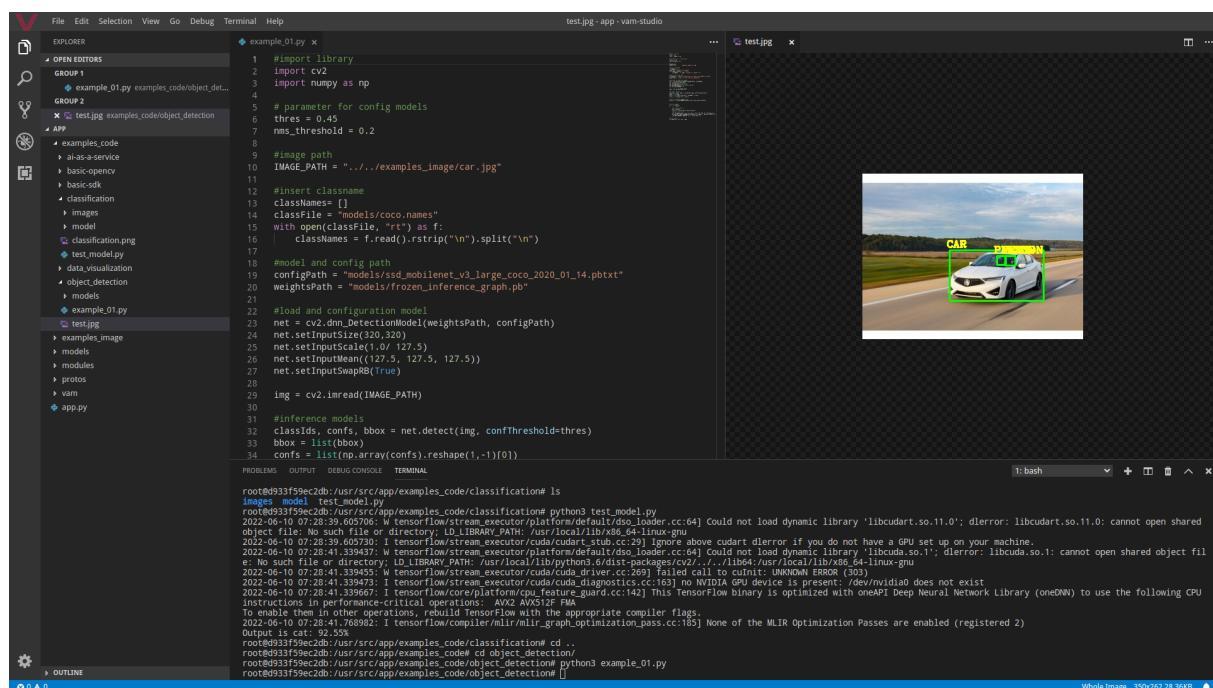
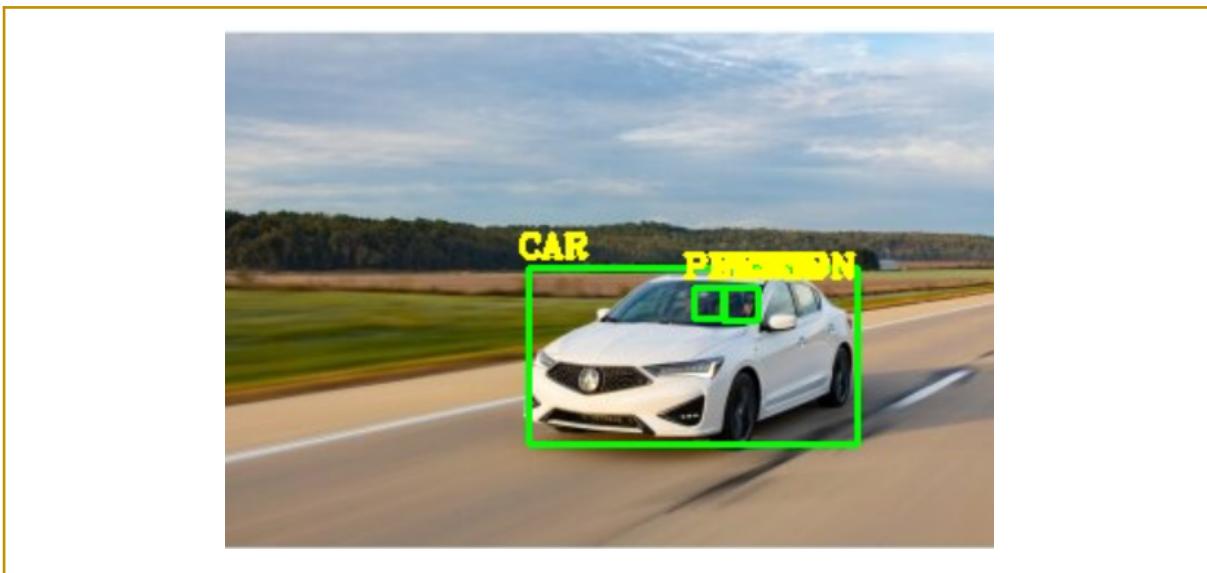
3. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์เดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

```
cd example_code/object_detection
```

4. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```
python3 example_01.py
```

ผลลัพธ์



รูปที่ 9.4 ผลการรันตัวอย่างโปรแกรมการตรวจจับบน VAM Studio

LAB 10 | Performance Assessment

Performance Assessment

Performance Assessment คือ การวัดและการประเมินประสิทธิภาพการทำงาน ซึ่งในบทเรียนนี้จะกล่าวถึงการวัดและการประเมินประสิทธิภาพการทำงานของโปรแกรมการประมวลผลข้อมูลหรือโมเดลที่มีความสามารถในการคำนวณและตัดสินใจ

การประเมินผลจะมีอยู่หลายรูปแบบ เช่น การประเมินผลจากความเร็วในการประมวลผลข้อมูล การประเมินผลจากความแม่นยำในการประมวลผลข้อมูล และ การประเมินผลจากการใช้ทรัพยากรของเครื่องประมวลผลของโปรแกรมประมวลผลข้อมูล

Confusion Matrix

Confusion matrix คือ เครื่องมือสำหรับใช้ในการวัดประสิทธิภาพการทำงานของโมเดลในทางปัญญาประดิษฐ์ (Artificial Intelligence: AI) หรือการเรียนรู้ของเครื่องจักร (Machine Learning) ที่มีการจำแนกหรือทำนายของ 2 วัตถุหรือมากกว่า 2 วัตถุขึ้นไป ว่ามีประสิทธิภาพการทำงานมากน้อยเพียงใดเมื่อเทียบระหว่างค่าความเป็นจริงกับค่าที่โมเดลทำนายผลออกมาได้

โดย Confusion matrix จะมีอยู่ 4 ค่าหลัก ๆ ที่สำคัญดังนี้

- True Positive (TP) คือ การทำนายว่า “จริง” และสิ่งที่เกิดขึ้น คือ “จริง”

ตัวอย่าง ทำนายว่าผู้หญิงคนหนึ่งกำลังตั้งครรภ์และเธอกำลังตั้งครรภ์จริง

- True Negative (TN) คือ การทำนายว่า “ไม่จริง” และสิ่งที่เกิดขึ้น คือ “ไม่จริง”

ตัวอย่าง ทำนายว่าผู้ชายคนหนึ่งไม่ได้ตั้งครรภ์และเขาก็ไม่ได้ตั้งครรภ์จริง

- False Positive (FP) คือ การทำนายว่า “จริง” แต่สิ่งที่เกิดขึ้น คือ “ไม่จริง”

ตัวอย่าง ทำนายว่าผู้หญิงคนหนึ่งตั้งครรภ์ แต่ในความเป็นจริงเขามิได้ตั้งครรภ์

- False Negative (FN) คือ การทำนายว่า “ไม่จริง” แต่สิ่งที่เกิดขึ้น คือ “จริง”

ตัวอย่าง ทำนายว่าผู้หญิงคนหนึ่งไม่ตั้งครรภ์ แต่ในความเป็นจริงเธอกำลังตั้งครรภ์

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

รูปที่ 10.1 ตาราง confusion matrix

(ที่มา: towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62)

ผู้เรียนสามารถนำองค์ความรู้มาต่อยอดในส่วนของการประเมินประสิทธิภาพความแม่นยำในการประมวลผลข้อมูลตามรูปแบบได้ดังนี้

Accuracy

Accuracy คือ ความถูกต้องที่โปรแกรมทำงานได้ตรงกับสิ่งที่เกิดขึ้นจริง

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN) \quad (10.1)$$

Precision

Precision คือ ค่าความแม่นยำการเปรียบเทียบการทำงานที่ถูกต้องว่า “จริง” และเกิดขึ้นจริง (TP) กับ การทำงานว่า “จริง” แต่สิ่งที่เกิดขึ้น คือ ไม่จริง (FP)

$$\text{Precision} = TP / (TP + FP) \quad (10.2)$$

Recall

Recall คือ ความถูกต้องของการทำนายว่าจะเป็น “จริง” เทียบกับ จำนวนครั้งของเหตุการณ์ที่ทำนายและเกิดขึ้นว่า “เป็น จริง”

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (10.3)$$

F1-Score

F1-Score เป็นค่าเฉลี่ยแบบ harmonic mean ระหว่าง precision และ recall

$$\text{F1} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \quad (10.4)$$

LAB 10.1 การคำนวณ Confusion Matrix

วัตถุประสงค์

1. เพื่อศึกษา Confusion Matrix
2. เพื่อเรียนรู้วิธีการประเมินประสิทธิภาพด้านความแม่นยำของโปรแกรมหรือโมเดลทำนายผลด้วย Confusion Matrix

10.1.1 การหาค่าความถูกต้องในการทำนายผลไม้

โจทย์

จงหาค่าความถูกต้องในการทำนายผลไม้ผลไม้ทั้งหมด 2 ชนิด โดยจะแบ่งเป็นแอปเปิล 9 ลูกและสตรอว์เบอร์รี่ 10 ลูก แต่ผลลัพธ์ที่ประมวลผลผ่านโมเดลมีความผิดพลาดไปจากค่าต้นฉบับ โดยจะแบ่งออกเป็น ค่า TP, TN, FP, FN ดังนี้

- **True Positive:** โปรแกรมทายว่าเป็นแอปเปิลถูกต้องจำนวน 6 ลูก
- **True Negative:** โปรแกรมทายว่าเป็นสตรอว์เบอร์รี่ถูกต้องจำนวน 8 ลูก
- **False Positive:** โปรแกรมทายว่าเป็นสตรอว์เบอร์รี่ แต่ความจริงคือแอปเปิลจำนวน 2 ลูก
- **False Negative:** โปรแกรมทายว่าเป็นแอปเปิล แต่ความจริงคือสตรอว์เบอร์รี่จำนวน 3 ลูก

		PREDICTED VALUES			
		Apple		Strawberry	
ACTUAL VALUES	Apple	Apple	Apple	Apple	Strawberry
	Strawberry		Strawberry	Strawberry	Strawberry

รูปที่ 10.2 ตัวอย่างตารางสำหรับใช้หาค่าความถูกต้องในการคำนวณผลลัพธ์ 2 ชนิด

(ที่มา: towardsdatascience.com/baffling-concept-of-true-positive-and-true-negative-bffbc340f107)

วิธีคำนวณ

จากตัวอย่างผลลัพธ์ผู้เรียนสามารถคำนวณ TP, TN, FP, FN มาคำนวณตามสูตรเพื่อประเมินประสิทธิภาพการทำงานของโปรแกรมได้ ดังนี้

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN})$$

$$= (6 + 8) / (6 + 8 + 2 + 3)$$

$$= 0.74 \text{ หรือ } 74\%$$

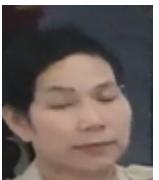
10.1.2 การประเมินประสิทธิภาพความแม่นยำในการทำนายใบหน้า

โดยทั่วไป

จะประเมินประสิทธิภาพความแม่นยำในการทำนายใบหน้าจากตารางผลการประมวลผลและวิเคราะห์ข้อมูลสำหรับการตรวจจับและรู้จำใบหน้า โดยมีเงื่อนไขความแม่นยำที่ยอมรับได้ว่าต้องเป็นใบหน้าเดียวกันและมีค่า Probability มากกว่าหรือเท่ากับ 75.0000%

ตารางที่ 10.1 ผลการประมวลผลและวิเคราะห์ข้อมูลสำหรับการตรวจจับและรู้จำใบหน้า

No.	Ground-truth	Detected face	Actually	Probability
1			True	76.9269
2			True	98.9958
3			False	98.5704
4			True	99.8225
5			True	92.1119

6			True	99.1964
7			False	99.6613
8			True	90.7335
9			True	95.4595
10			True	72.1353
11			False	70.2568

12			True	90.3368
13			True	98.2586
14			True	94.8709
15			True	74.9158

จากตารางผลการประมวลผลและวิเคราะห์ข้อมูลสำหรับการตรวจสอบและรู้จักใบหน้าสามารถนำวิเคราะห์ประสมท้องภาพความแม่นยำได้ดังนี้

- **True Positive:** โปรแกรมทายว่าเป็นใบหน้าเดียวกันและมีค่า probability มากกว่าหรือเท่ากับ 75.0000% มีจำนวน 10 ภาพ คือ หมายเลข คือ 1, 2, 4, 5, 6, 8, 9, 12, 13 และ 14
- **True Negative:** โปรแกรมทายว่าไม่ใช่ใบหน้าเดียวกันและมีค่า probability น้อยกว่า 75.0000% มีจำนวน 1 ภาพ คือ หมายเลข 11
- **False Positive:** โปรแกรมทายว่าไม่ใช่ใบหน้าเดียวกันและมีค่า probability มากกว่าหรือเท่ากับ 75.0000% จำนวน 2 ภาพ คือ หมายเลข 3 และ 7
- **False Negative:** โปรแกรมทายว่าเป็นใบหน้าเดียวกันและมีค่า probability น้อยกว่า 75.0000% มีจำนวน 2 ภาพ คือ หมายเลข 10 และ 15

วิธีคำนวณ

ผู้เรียนสามารถนำค่า TP, TN, FP, FN มาคำนวณตามสูตรเพื่อประเมินประสิทธิภาพการทำงานของโปรแกรมได้ ดังนี้

$$\begin{aligned} \text{Accuracy} &= (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN}) \\ &= (10 + 1) / (10 + 1 + 2 + 2) \\ &= 11/15 \\ &= 0.73 \text{ หรือ } 73\% \end{aligned}$$

$$\begin{aligned} \text{Precision} &= \text{TP} / (\text{TP} + \text{FP}) \\ &= 10 / (10 + 2) \\ &= 10/12 \\ &= 0.83 \text{ หรือ } 83\% \end{aligned}$$

$$\begin{aligned} \text{Recall} &= \text{TP} / (\text{TP} + \text{FN}) \\ &= 10 / (10 + 2) \\ &= 10/12 \\ &= 0.83 \text{ หรือ } 83\% \end{aligned}$$

$$\begin{aligned} \text{F1-Score} &= 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall}) \\ &= 2 \times (0.83 \times 0.83) / (0.83 + 0.83) \\ &= 2 \times (0.6889) / (1.66) \\ &= 1.3778 / 1.66 \\ &= 0.83 \text{ หรือ } 83\% \end{aligned}$$

โจทย์

จงประเมินประสิทธิภาพความแม่นยำในการทนายของผู้หญิง 2 แบบ โดยจะแบ่งเป็นผู้หญิงที่ตั้งครรภ์ 8 คนและผู้หญิงที่ไม่ตั้งครรภ์ 8 คน ให้ผู้เรียนทำการประเมินประสิทธิภาพการทนายของโปรแกรมในด้านต่าง ๆ ดังนี้

- Accuracy
- Precision
- Recall
- F1-Score

		PREDICTED VALUES	
		Pregnant	Not Pregnant
ACTUAL VALUES	Pregnant		
	Not Pregnant		

รูปที่ 10.3 ตารางประเมินประสิทธิภาพความแม่นยำในการทนายของผู้หญิง 2 แบบ
(ที่มา: towardsdatascience.com/baffling-concept-of-true-positive-and-true-negative-bffbc340f107)

สรุปและบันทึกผล

.....

.....

.....

.....

.....

.....

.....

LAB 11 | Data Visualization

วัตถุประสงค์

- เพื่อเรียนรู้วิธีการใช้งานตัวอย่างโปรแกรมการสร้างแพนคูมิแสดงข้อมูลบน VAM Studio
 - เพื่อเรียนรู้วิธีการดึงข้อมูลมาแสดงผลในรูปแบบของแพนคูมิด้วย Vue.js

อุปกรณ์ที่เกี่ยวข้อง

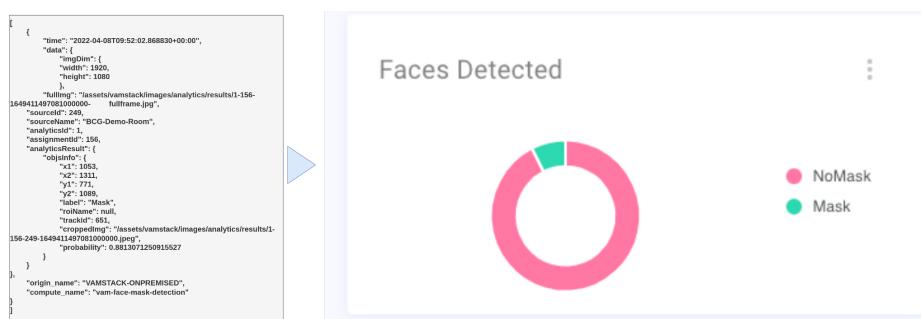
- ## 1. ชุดการทดลอง VAM Platform

Data Visualization

Data Visualization คือ การนำข้อมูลที่ได้มาจากการแหล่งข้อมูลต่าง ๆ มาประมวลผลและวิเคราะห์แล้วนำเสนอในรูปแบบที่มองเห็นและทำความเข้าใจได้ง่าย เช่น แผนภูมิ รูปภาพ แผนที่ กราฟแสดงเทอนด์ ตาราง วิดีโอ อินโฟกราฟิก (Infographic) รวมถึงแดชบอร์ด (dashboard) เป็นต้น โดยในบทเรียนจะเน้นไปที่การนำเสนอข้อมูล

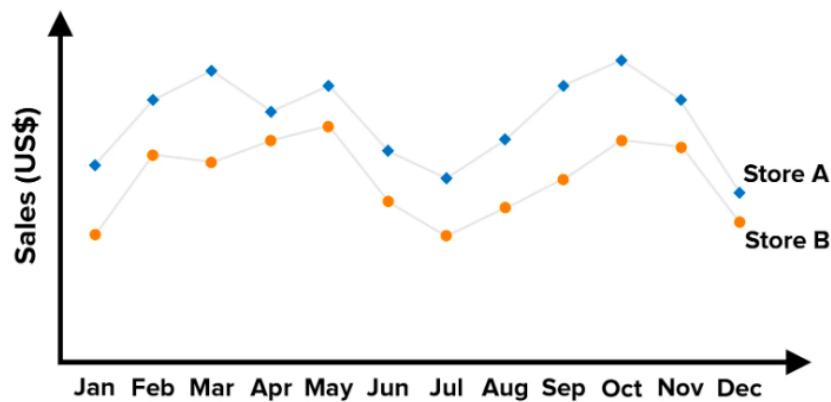
รูปแบบพื้นฐานของการทำ Data Visualization

- แผนภูมิ (Charts) เป็นรูปแบบที่มีหลายหลายชนิดเพื่อมาช่วยกับการนำเสนอข้อมูลที่แตกต่างกันไปตามวัตถุประสงค์ เช่น การใช้แผนภูมิเส้น (Line Chart) แสดงค่าสถิติ การใช้แผนภูมิวงกลม (Pie Chart) แสดงปริมาณความแตกต่างของข้อมูล เป็นต้น



รูปที่ 11.1 การนำเสนอข้อมูลจำนวนการตรวจจับผู้สูมิใช้หน้ากากอนามัยด้วย Pie Chart

2. กราฟ (Graphs) เป็นอีกประเภทหนึ่งของแผนภูมิ กราฟจะทำหน้าที่แสดงความสัมพันธ์ระหว่างข้อมูล 2 ตัวแปร คือ แกนแนวโน้ม (แกน X) และแกนแนวตั้ง (แกน Y) ช่วยให้เห็นเท่านั้นถ้าการณ์สามารถการณ์ประกอบกับบริบทได้เป็นอย่างดี



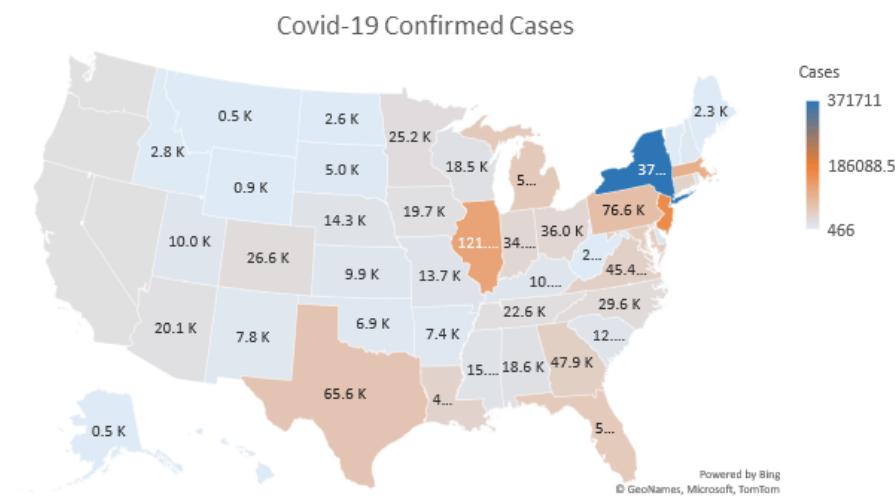
รูปที่ 11.2 การนำเสนอข้อมูลสถิติยอดการขายสินค้าในแต่ละเดือนของร้าน A และ B

(ที่มา: https://www.mindtools.com/pages/article/Charts_and_Diagrams.htm)

3. ตาราง (Tables) เป็นรูปนำเสนอข้อมูลให้ออกมาดูง่าย ตารางประกอบไปด้วย 2 ส่วน ได้แก่ คอลัมน์และแถว ซึ่งช่วยจัดการข้อมูลให้เรียบร้อย

รูปที่ 11.3 การนำเสนอรายละเอียดข้อมูลของการตรวจจับผู้สูมิฬหน้ากากอนามัย

4. แผนที่ (Maps) เป็นรูปแบบการนำเสนอข้อมูลบนแผนที่เพื่อแสดงข้อมูลเกี่ยวกับพื้นที่ต่าง ๆ



รูปที่ 11.4 การนำเสนอข้อมูลยอดผู้ติดเชื้อ Covid-19 ในแต่ละรัฐของประเทศไทย

(ที่มา: <https://www.spreadsheetweb.com/excel-map-chart/>)

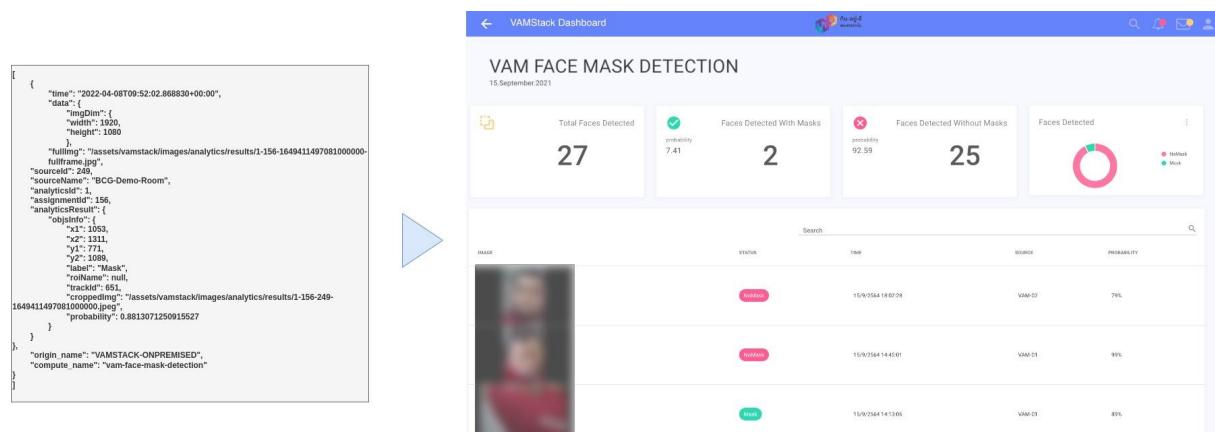
5. อินโฟกราฟิก (Infographics) คือ การนำเสนอสารสนเทศ (Info: information) ด้วยภาพกราฟิก (Graphic) เป็นรูปแบบการนำเสนอข้อมูลที่ใช้ภาพสื่อแทน มีการนำเทคนิคการเล่าเรื่องมาใช้ จึงนิยมใช้สำหรับการเสนอเนื้อหา ความรู้ หรือเป็นสื่อการเรียนการสอน



รูปที่ 11.5 การนำเสนอข้อมูลวิธีการป้องกันตัวเองจาก Covid-19

(ที่มา: <https://mahidol.ac.th/th/info/graphic-clips2/>)

6. แดชบอร์ด (Dashboards) คือ การนำข้อมูลมาเรียบเรียงและสรุปผลเป็นภาพ โดยมีแผนภูมิหรือกราฟมาใช้ในการนำเสนอ ปัจจุบันนิยมใช้สำหรับการนำเสนอข้อมูลแบบ Realtime ผ่านซอฟต์แวร์หรือเครื่องมือการจัดการและวิเคราะห์ข้อมูล



รูปที่ 11.6 การนำเสนอบัญชีผลการตรวจจับผู้สวมใส่หน้ากากอนามัย

ประโยชน์ของการทำ Data Visualization

1. ช่วยให้เข้าใจข้อมูลได้ง่ายขึ้น
2. ช่วยให้อ่านมานความสัมพันธ์ของข้อมูลได้ง่ายขึ้น
3. ช่วยประหยัดเวลาในการตีความข้อมูลและตัดสินใจ ลดภาระการค้นหาและเปรียบเทียบข้อมูล
4. ช่วยให้สามารถมองเห็นจุดที่น่าสนใจของข้อมูล
5. ช่วยให้ข้อมูลมีความน่าสนใจมากขึ้น

Vue.js

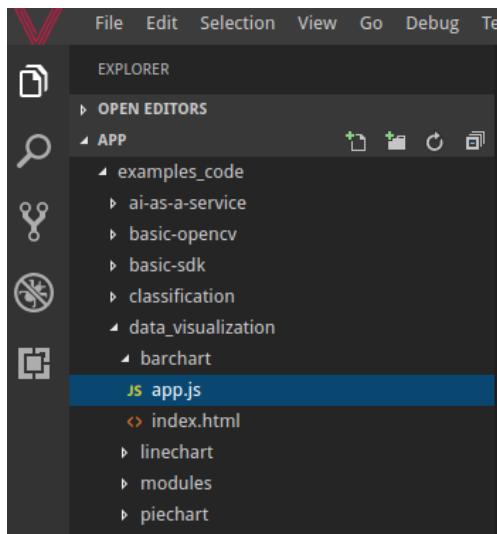
Vue.js คือ JavaScript Framework ที่สามารถสร้าง Web application แบบ SPA (Single Page Application) เป็นตัวที่กำเนิดมาหลัง React และ Angular โดย Concept ของการเขียน มีนักพัฒนาหลาย ๆ คน กล่าวว่ามันคือ Framework ที่รวมข้อดีของ React และ Angular มาไว้ด้วยกัน

11.1 การเข้าถึงตัวอย่างโปรแกรมสำหรับสร้าง Bar Chart

ขั้นตอนการเข้าถึงตัวอย่างโปรแกรมสำหรับสร้าง Bar Chart

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))

2. คลิก example_code > data_visualization > barchart ที่ແຄບເມຸນດ້ານຊ້າຍ



ຮູບທີ 11.7 ການເຂົ້າສີ່ໄຟລ໌ຕ້ອງຢ່າງໂປຣແກຣມສໍາຫຼັບສ້າງ Bar Chart

3. ໄຟລ໌ຕ້ອງຢ່າງໂປຣແກຣມຈະປະກອບໄປດ້ວຍ 2 ສ່ວນດັ່ງນີ້

- 3.1. ໃຟລ໌ app.js ໃໃນການคำນວນ ສ້າງອັກອີຣີມ ສ້າງກາຟ ສ້າງແພນກຸມແລ້ວນຳສິ່ງທີ່ສ້າງມາໄດ້ເຂື່ອມຕ່ອກກັບ HTML
- 3.2. index.html ໃໃນການເຂື່ອມຕ່ອກກັບ Vue.js ແລະ ແສດງຜົດຕ່າງ ๆ ອອກມາໃນຮູບແບບຂອງ web-based

ຕ້ອງຢ່າງໂປຣແກຣມໃນໄຟລ໌ app.js

```
# example_code/data-visualization/barchart/app.js

Vue.component('bar-chart', {
  extends: VueChartJs.Bar,
  data: function () {
    return {
      datacollection: {
        labels: [], // x axis
        datasets: [{
          label: 'people counting',
          backgroundColor: '#f87979', //can change color of bar chart here.
          link: https://www.color-hex.com/
        pointBackgroundColor: 'white',
        borderWidth: 1,
        pointBorderColor: '#249EBF',
        data: [] // y axis
      }],
    },
  },
})
```

```

options: {
  scales: {
    yAxes: [
      ticks: {
        beginAtZero: true
      },
      gridLines: {
        display: true
      }
    ],
    xAxes: [
      ticks: {
        beginAtZero: true
      },
      gridLines: {
        display: true
      }
    ]
  },
  legend: {
    display: false
  },
  tooltips: {
    enabled: true,
    mode: 'single',
    callbacks: {
      label: function (tooltipItems, data) {
        return '#' + tooltipItems.yLabel;
      }
    }
  },
  responsive: true,
  maintainAspectRatio: false,
  height: 200
}
};

},
async mounted() {
  await this.peopleCounting()
  this.renderChart(this.datacollection, this.options)
}

},
methods: {
  async peopleCounting() {
    const limit = 5 //parameterfor modify limit graph

```

```
machine_ip = '10.1.1.212' //parameter for change machine ip

text_01 = 'http://'
text_02 = ':8888/analytics/?limit='
text_03 = '&compute_id=7'
let result = text_01.concat(machine_ip, text_02, limit, text_03);
try {
  let data = await (await fetch(result)).json()

  for (let i = 0; i < data["length"]; i++) {
    this.datacollection.labels.push(data[i]["time"])

this.datacollection.datasets[0].data.push(data[i]["data"]["analyticsResult"]
["cnt"])
  }
  catch (err) {
    console.log(err)
  }

  // });
},
}
});

var app = new Vue({
  el: '#app',
})
```

ตัวอย่างโปรแกรมในไฟล์ index.html

```
# example_code/data-visualization/barchart/index.html

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Data Visualization</title>

    <!-- CSS only -->
    <!-- <link
        href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/css/bootstrap.min.cs
        s" rel="stylesheet"> -->
    <meta http-equiv="Access-Control-Allow-Origin" content="*">

</head>

<body>
    <!-- connected barchart information from app.js -->
    <div id="app" class="contrainer">
        <h1>People Counting</h1>
        <bar-chart></bar-chart>
    </div>
    <!-- connected barchart information from app.js -->

    <!-- development version, includes helpful console warnings -->
    <script src="../modules/vue.js"></script>
    <!-- <script src="{{ url_for('modules', filename='vue.js') }}></script>
-->

    <script src="../modules/Chart.min.js"></script>
    <script src="../modules/vue-chartjs.min.js"></script>

    <script src="../modules/axios.min.js"></script>
    <!-- <script src="https://unpkg.com/axios/dist/axios.min.js"></script>
-->

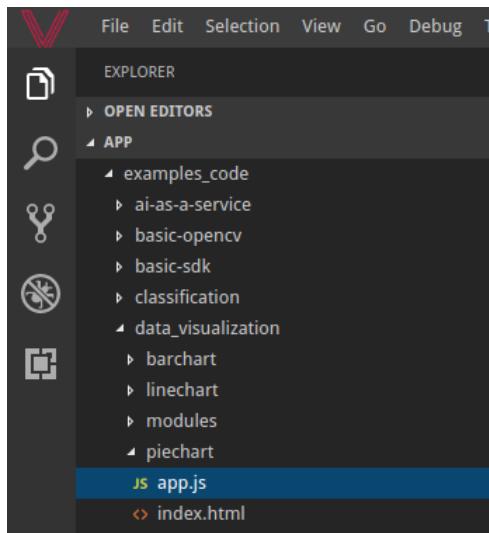
    <!-- JavaScript Bundle with Popper -->
    <script src="../modules/bootstrap.bundle.min.js"></script>
    <script src="app.js" charset="utf-8"></script>

</body>
</html>
```

11.2 การเข้าถึงตัวอย่างโปรแกรมสำหรับสร้าง Pie Chart

ขั้นตอนการเข้าถึงตัวอย่างโปรแกรมสำหรับสร้าง Pie Chart

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > data_visualization > piechart ที่แถบเมนูด้านซ้าย



รูปที่ 11.8 การเข้าถึงไฟล์ตัวอย่างโปรแกรมสำหรับสร้าง Pie Chart

3. ไฟล์ตัวอย่างโปรแกรมจะประกอบไปด้วย 2 ส่วนดังนี้
 - 3.1. ไฟล์ app.js ใช้ในการคำนวณ สร้างอัลกอริธึม สร้างกราฟ สร้างแผนภูมิแล้วนำสิ่งที่สร้างมาได้เชื่อมต่อกับ HTML
 - 3.2. index.html ใช้ในการเชื่อมต่อกับ Vue.js และแสดงผลต่าง ๆ ออกมาในรูปแบบของ web-based

ตัวอย่างโปรแกรมในไฟล์ app.js

```
# example_code/data-visualization/piechart/app.js

Vue.component('pie-chart', {
  extends: VueChartJs.Pie,
  data: function () {
    return {
      datacollection: {
        labels: [], // labelname
        datasets: [
          {
            backgroundColor: ['#1E9600', '#99C802'], //can change
            color of line chart here. link: https://www.color-hex.com/
          }
        ]
      }
    }
  }
})
```

```

        data: [], // data in piechart
    },
],
},
options: {
    responsive: true,
    legend: {
        display: true,
        position: 'right',
    },
},
},
},
async mounted() {

    await this.FaceMaskDetection()

    // this.chartData is created in the mixin
    this.renderChart(this.datacollection, this.options)
},
methods: {
    async FaceMaskDetection() {
        const day = 100 //parameter for modify day for getting data
        machine_ip = '10.1.1.212' //parameter for change machine ip

        text_01 = 'http://'
        text_02 = ':8888/activity_piechart?day='
        text_03 = '&compute_id=1'

        let result = text_01.concat(machine_ip, text_02, day, text_03);

        try {
            let data = await (await fetch(result)).json()
            this.datacollection.labels = Object.keys(data)
            this.datacollection.datasets[0].data = Object.values(data)
        }
        catch (err) {
            console.log(err)
        }
    }
},
}),
var app = new Vue({

```

```
    el: '#app',
})
```

ตัวอย่างโปรแกรมในไฟล์ index.html

```
# example_code/data-visualization/piechart/index.html

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Data Visualization</title>

  <!-- CSS only -->
  <!-- <link
  href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/css/bootstrap.min.css"
  rel="stylesheet"> -->
  <meta http-equiv="Access-Control-Allow-Origin" content="*">

  <style>
    * {
      box-sizing: border-box;
    }

    /* Create three equal columns that floats next to each other */
    .column {
      float: left;
      width: 33.33%;
      padding: 10px;
      height: 300px;
      /* Should be removed. Only for demonstration */
    }

    /* Clear floats after the columns */
    .row:after {
      content: "";
      display: table;
      clear: both;
    }
  </style>
</head>
```

```
<body>
  <!-- connected piechart information from app.js -->
  <div id="app" class="row">

    <div class="row">
      <div class="column">
        <h1>Face Mask Detection</h1>
        <pie-chart></pie-chart>
      </div>
    </div>
  </div>
  <!-- connected piechart information from app.js -->
  <!-- development version, includes helpful console warnings -->
  <script src="../modules/vue.js"></script>
  <!-- <script src="{{ url_for('modules', filename='vue.js') }}"/></script>
-->

  <script src="../modules/Chart.min.js"></script>
  <script src="../modules/vue-chartjs.min.js"></script>

  <script src="../modules/axios.min.js"></script>
  <!-- <script src="https://unpkg.com/axios/dist/axios.min.js"></script>
-->

  <!-- JavaScript Bundle with Popper -->
  <script src="../modules/bootstrap.bundle.min.js"></script>

  <script src="app.js" charset="utf-8"></script>

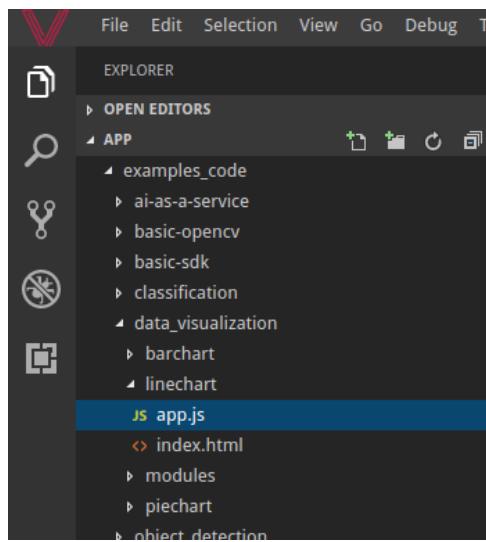
</body>

</html>
```

11.3 การเข้าถึงตัวอย่างโปรแกรมสำหรับสร้าง Line Chart

ขั้นตอนการเข้าถึงตัวอย่างโปรแกรมสำหรับสร้าง Line Chart

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. คลิก example_code > data_visualization > linechart ที่แถบเมนูด้านซ้าย



รูปที่ 11.9 การเข้าถึงไฟล์ตัวอย่างโปรแกรมสำหรับสร้าง Line Chart

3. ไฟล์ตัวอย่างโปรแกรมจะประกอบไปด้วย 2 ส่วนดังนี้
 - 3.1. ไฟล์ app.js ใช้ในการคำนวณ สร้างอัลกอริธึม สร้างกราฟ สร้างแผนภูมิแล้วนำสิ่งที่สร้างมาได้เชื่อมต่อกับ HTML
 - 3.2. index.html ใช้ในการเชื่อมต่อกับ Vue.js และแสดงผลต่าง ๆ ออกมาในรูปแบบของ web-based

ตัวอย่างโปรแกรมในไฟล์ app.js

```
# example_code/data-visualization/linechart/app.js

Vue.component('line-chart', {
  extends: VueChartJs.Line,
  data: function () {
    return {
      datacollection: {
        labels: [], // x axis
        datasets: [
          {
            label: 'amount peoples in zones',
            backgroundColor: '#FFF2CC', //can change color of

```

```

line chart here. #f87979 is hex value color. link:
https://www.color-hex.com/
    data: [] // y axis
  }
],
},
option: {
  responsive: true,
  maintainAspectRatio: false
}
);
},
async mounted() {
  await this.peopleCounting()
  this.renderChart(this.datacollection, this.option)
},
methods: {
  async peopleCounting() {
    const limit = 5 //parameterfor modify limit graph
    machine_ip = '10.1.1.212' //parameter for change machine ip

    text_01 = 'http://'
    text_02 = ':8888/analytics/?limit='
    text_03 = '&compute_id=7'
    let result = text_01.concat(machine_ip, text_02, limit, text_03);
    try {
      let data = await (await fetch(result)).json()

      for (let i = 0; i < data["length"]; i++) {

        this.datacollection.labels.push(data[i]["data"]["analyticsResult"]["objsInfo"]
        ["roiName"])

        this.datacollection.datasets[0].data.push(data[i]["data"]["analyticsResult"]
        ["cnt"])
      }
    }
    catch (err) {
      console.log(err)
    }

    // });
  },
}

```

```
});  
  
var app = new Vue({  
  el: '#app',  
})
```

ตัวอย่างโปรแกรมในไฟล์ index.html

```
# example_code/data-visualization/linechart/index.html  
  
<!DOCTYPE html>  
<html lang="en">  
  
<head>  
  <meta charset="UTF-8">  
  <meta http-equiv="X-UA-Compatible" content="IE=edge">  
  <meta name="viewport" content="width=device-width, initial-scale=1.0">  
  <title>Data Visualization</title>  
  
  <!-- CSS only -->  
  <!-- <link  
  href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.0/dist/css/bootstrap.min.cs  
s" rel="stylesheet"> -->  
  <meta http-equiv="Access-Control-Allow-Origin" content="*"/>  
  
</head>  
  
<body>  
  <!-- connected linechart information from app.js -->  
  <div id="app" class="contrainer">  
    <h1>People Counting in Line Chart</h1>  
    <line-chart></line-chart>  
  </div>  
  <!-- connected line information from app.js -->  
  
  <!-- development version, includes helpful console warnings -->  
  <script src="../modules/vue.js"></script>  
  <!-- <script src="{{ url_for('modules', filename='vue.js') }}></script>  
-->  
  
  <script src="../modules/Chart.min.js"></script>  
  <script src="../modules/vue-chartjs.min.js"></script>
```

```

<script src="../modules/axios.min.js"></script>
<!-- &lt;script src="https://unpkg.com/axios/dist/axios.min.js"&gt;&lt;/script&gt;
--&gt;

&lt;!-- JavaScript Bundle with Popper --&gt;
&lt;script src="../modules/bootstrap.bundle.min.js"&gt;&lt;/script&gt;

&lt;script src="app.js" charset="utf-8"&gt;&lt;/script&gt;

&lt;/body&gt;

&lt;/html&gt;
</pre>

```

11.4 ขั้นตอนการใช้ตัวอย่างโปรแกรมแสดงผลข้อมูลในลักษณะแผนภูมิ

1. เข้าสู่ระบบของ VAM Studio ([LAB 2.1](#))
2. ให้ผู้เรียนพิมพ์คำสั่งดังต่อไปนี้ใน terminal เพื่อเข้าถึงไฟล์เดอร์ที่เก็บไฟล์ตัวอย่างโปรแกรม

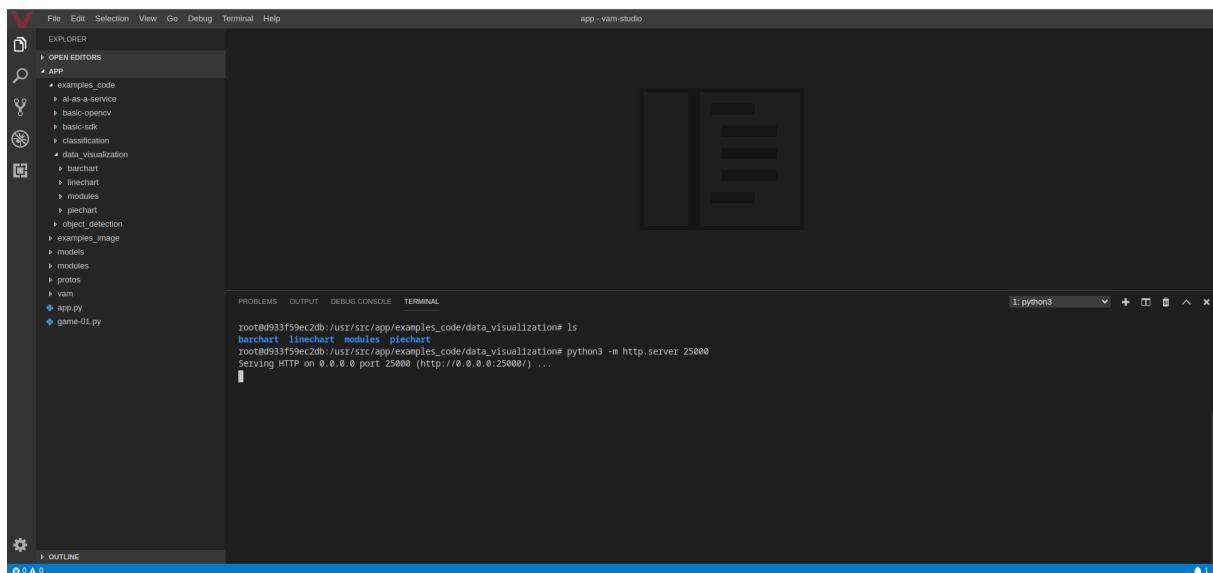
```
cd example_code/data-visualization
```

3. ให้ผู้เรียนพิมพ์คำสั่งต่อไปนี้ใน terminal เพื่อรันโปรแกรม

```

python3 -m http.server <port>
# example python3 -m http.server 25000

```



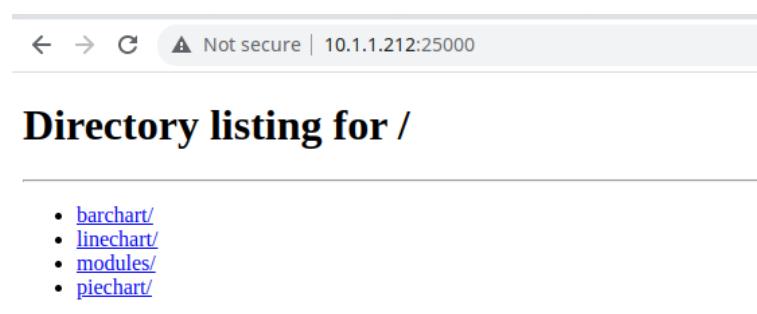
รูปที่ 11.10 ตัวอย่างหน้าต่างการทำงานหลังรันโปรแกรม

หมายเหตุ

แต่ละกลุ่มห้ามใส่ตัวเลข port ที่ซ้ำกัน โดยมีเงื่อนไขกำหนดเลข prot ดังนี้

1. บัญชี Team01 ใช้เลข port ที่ 25000
2. บัญชี Team02 ใช้เลข port ที่ 25001
3. บัญชี Team03 ใช้เลข port ที่ 25002
4. บัญชี Team04 ใช้เลข port ที่ 25003
5. บัญชี Team05 ใช้เลข port ที่ 25004

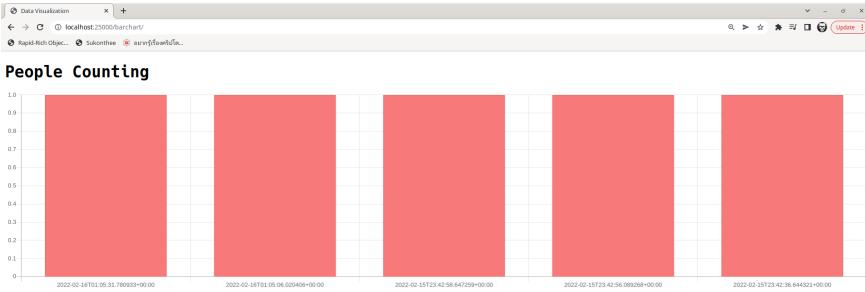
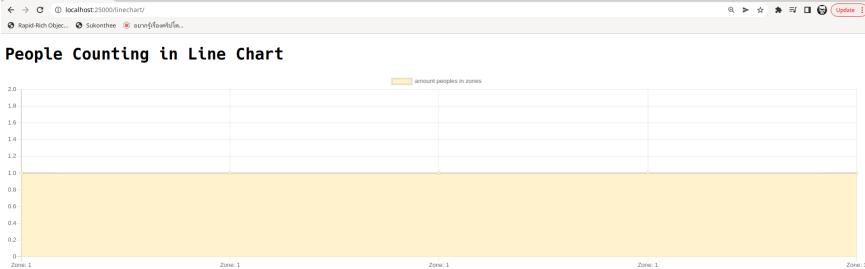
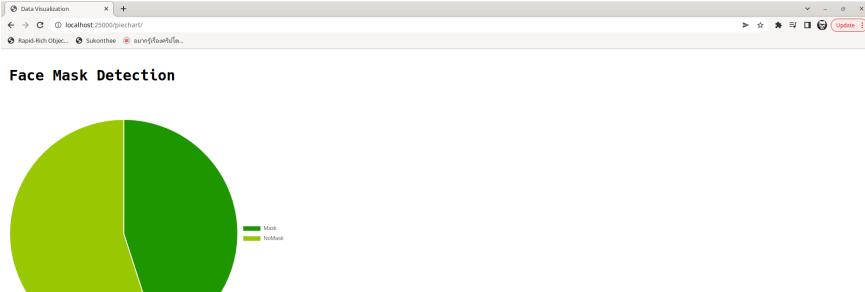
4. เมื่อรันโปรแกรมสำเร็จ ให้ผู้เรียนเปิดเบราว์เซอร์ใหม่อีกหนึ่งหน้าและพิมพ์ url:localhost:<port> หรือ 10.1.1.212:<port>



รูปที่ 11.11 หน้าต่างการทำงานของ 10.1.1.212:25000

5. ให้ผู้เรียนคลิกเลือกดูหน้าต่างการทำงานของแต่ละแผนภูมิ

ตารางที่ 11.1 การแสดงผลในลักษณะแผนภูมิต่าง ๆ

กราฟ	ผลลัพธ์
Barchart	
Linechart	
Piechart	

บรรณานุกรม

ไม่ปรากฏผู้เขียน. (2564). การใช้งานโมดูล RANDOM สร้างเกมสุ่มตัวเลข. สีบคัน 13 มิถุนายน

2565, จาก <https://www.mindphp.com/developer/tips-python/7869-module-random-game.html>

ไม่ปรากฏผู้เขียน. (2560). LAN คืออะไร. สีบคัน 10 มิถุนายน 2565,

จาก <https://www.mindphp.com/คุณวี/73-คืออะไร/2222-lan-คืออะไร.html>

Adrian Rosebrock, (2021), *OpenCV Getting and Setting Pixels*, Retrieved 13 June 2022, from

<https://pyimagesearch.com/2021/01/20/opencv-getting-and-setting-pixels/>

Alexander Mordvintsev and Abid Rahman K., *OpenCV-Python Tutorials*, (2022),

Retrieved 11 June 2022, from https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html

AWS, (n.d), *What is RESTful API?*, Retrieved 20 August 2022 ,

from https://aws.amazon.com/what-is/restful-api/?nc1=h_ls

Brien Posey, (2018), Real Time Streaming Protocol (RTSP), Retrieved 14 June 2022,

from <https://www.techtarget.com/searchvirtualdesktop/definition/Real-Time-Streaming-Protocol-RTSP>

Conex, (n.d), *Machine Vision Applications*, Retrieved 13 June 2022,

from <https://www.cognex.com/what-is/machine-vision/applications>

iNNEKT. (2017). IP Camera คืออะไร. สีบคัน 11 มิถุนายน 2565,

จาก <https://innekt.com/?page=ReviewDetail&Review=12>

IQS Directory, (n.d.), Machine Vision Systems, Retrieved 13 June 2022, from

<https://www.iqsdirectory.com/articles/machine-vision-system.html#what-are-machine-vision-systems>

Jedsada Saengow. (2561). [Vue Native] คือ อะไร. สีบคัน 30 พฤษภาคม 2565,

จาก <https://medium.com/jed-ng/vue-native-คืออะไร-ทำความรู้จัก-และเริ่มต้นสร้าง-project-5913eda9b7dc>

Junaid Rehman, (2021), *What is local area network (LAN) in computer*. Retrieved 10

June 2022, from <https://www.itrelease.com/2021/04/what-is-local-area-network-lan-in-computer/>

KT Smarttech, (2562). แลน (Local Area Network หรือ LAN) คืออะไร ?. สีบคัน 10 มิถุนายน 2565, จาก <https://www.ksmarttech.com/แลน-local-area-network-หรือ-lan-คืออะไร/>

Machine Vision (Thailand). (ม.ป.ป). ระบบ Machine Vision. สีบคัน 13 มิถุนายน 2565,

จาก <https://machinevision.co.th/technical-knowledge/ระบบ-machine-vision/>

Mostori, (2563), *What is Machine Vision ?*, Retrieved 13 June 2022,

from https://www.mostori.com/blog_detail.php?b_id=79

Pagon Gatchalee. (2562). Confusion Matrix. สีบคัน 31 พฤษภาคม 2565,

จาก <https://medium.com/@pagongatchalee/confusion-matrix-เครื่องมือสำคัญในการประเมินผลลัพธ์ของการทํานาย-ใน-machine-learning-fba6e3f9508c>

Saixiii. (2560). RESTful หรือ REST คือ. สีบคัน 10 มิถุนายน 2560,

จาก <https://saixiii.com/what-is-restful/>

- Sakul Montha (2562). REST กับ RESTful API ต่างกันนะรึยัง. สีบคันเมื่อ 20 สิงหาคม 2565,
จาก <https://iameique.medium.com/restful-api-ต่างกันนะรึยัง-2c70c42990e3>
- Sarang Narkhede. (2018). Understanding Confusion Matrix. Retrieved 1 June 2022,
from <https://towardsdatascience.com/understanding-confusion-matrix- a9ad42dcfd62>
- VECLThai, (2011), CNB IP CAMERA (IP Surveillance), Retrieved 11 June 2022,
from <http://www.vecthai.com/main/?p=4777>
- 1stCraft Team. (2563). Data Visualization คืออะไร?. สีบคัน 30 พฤษภาคม 2565,
จาก <https://1stcraft.com/what-is-data-visualization/>