



VAMStack

Better Platform.
Better AI.



VAMStack

Design House

Name : VAMStack Co., Ltd.

TAX ID : 0105561080465

Founded : May, 10 2561

Paid-up Capital : 7,000,000 ฿

Member of :TESA, DEPA, DGTi

Products :

- Video Analytic Platform Provider
 - Image Processing, ML, DL and AI Modules
 - AI for Retail, Manufacturers, Security, Surveillance and Medical Solution

Sale Team (Bangkok)



Developer Team

(Bangsaen Design House)



VAM Platform

“เปิดโลกใหม่ด้วย AI ก้าวสู่การทำงานใน
เชิงพาณิชย์ด้วย AI Analytics Platform”



VAMSTACK

The collage consists of six screenshots from the VAM Platform:

- Top Left:** A video feed showing multiple people in a room, with green and blue lines drawn around them, indicating tracked objects.
- Top Right:** A screenshot of the "Analytics View Output" section, showing a camera feed with overlaid analysis results.
- Middle Left:** The "Analytics Store" page, featuring a banner for "VAMStack" and several thumbnail previews for different analytics modules: Face Mask Detection, People Detection, Vehicle Detection, Living Activity Detection, and Intrusion Detection.
- Middle Right:** The "Facial Recognition" module, showing a dashboard with "Member Registration" and "Member Management" sections, and a preview of detected faces at an entrance.
- Bottom Left:** The "Searching" module, displaying a search interface with a table of results and a timeline graph.
- Bottom Right:** Another view of the "Searching" module, showing a timeline graph with red and white bars.

Core Technology



Home



Fall Out Detection



Loitering Detection



People Monitoring



Intrusion Detection



Activity Recognition



Fire and Smoke Detection



Vehicle Monitoring

Building

Office Department



- Face Recognition
- Access Control
- People Monitoring
- Workforce Management



Hotel Walkway

- Suspect Searching and Tracking
- Action Recognition



Hotel Care Center

- Management and Service



Public Area

- Intrusion Detection
- Alarm Detector



Parking

- Vehicle detection and Monitoring
- Parking Available (Outdoor)



Lobby

- Face Recognition
- Member Registration
- People Counting / in-out

City Surveillance

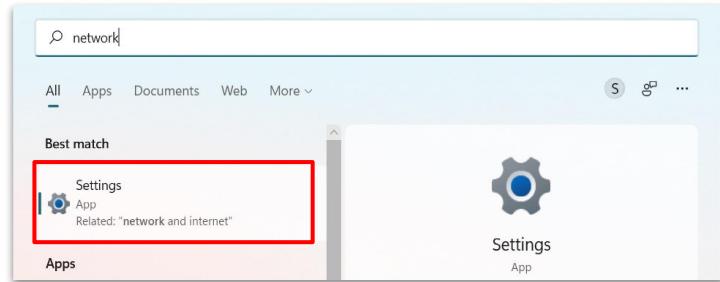




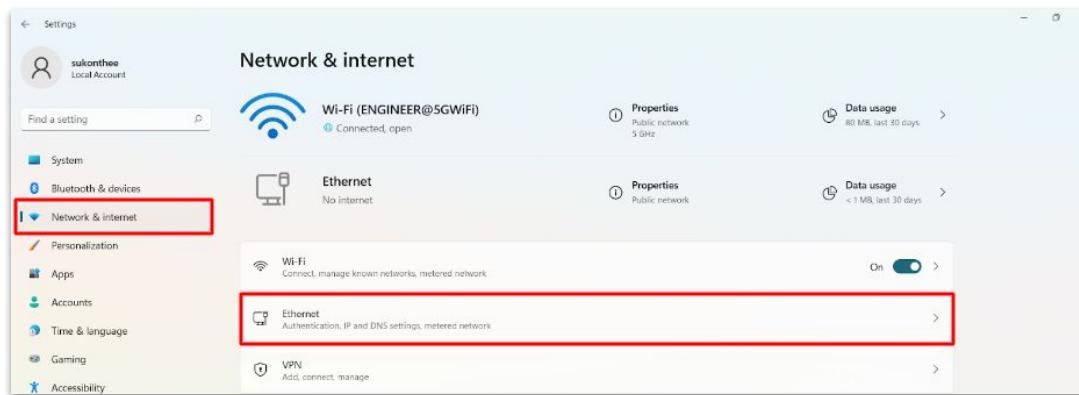
1. Connect NUC
 - 1.1. Windows OS
 - 1.2. Linux OS
 - 1.3. Mac OS
2. VAM Platform
3. VAM Studio
4. VAM SDK
 - 4.1. ฟังก์ชันสำหรับการรับข้อมูลของกล้องที่อยู่ใน VAM Platform
 - 4.1.1. listSource
 - 4.1.2. get_vamsnapshot
 - 4.1.3. get_vamroi
 - 4.1.4. draw_roi
 - 4.2. ฟังก์ชันพื้นฐานสำหรับการประมวลผลและวิเคราะห์ข้อมูล
 - 4.2.1. people_detection
 - 4.2.2. people_counting
 - 4.3. ฟังก์ชันสำหรับการส่งข้อมูลที่ประมวลผลเข้า VAM Platform
 - 4.3.1. set_vamlivefigure
 - 4.3.2. set_data2vamMetadata
5. VAM AlaaS
 - 5.1. Face Registration
 - 5.1.1. EnrollFace
 - 5.2. Face Recognition
 - 5.2.1. FaceFeature
 - 5.3. Face Detection and Quality Verification
 - 5.3.1. CompareFace



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

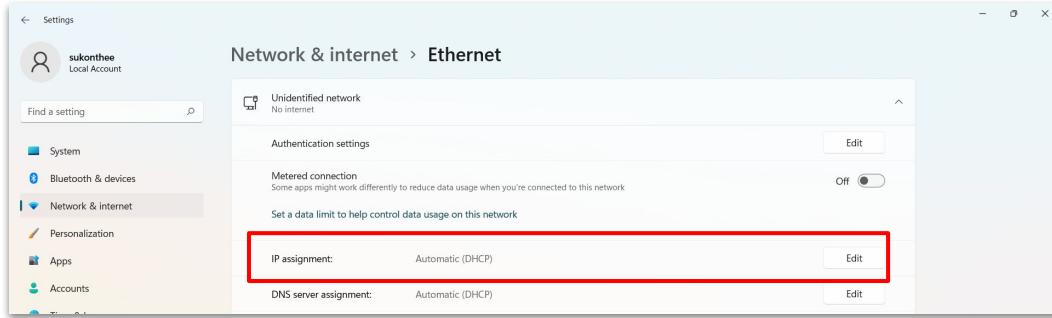


- คลิก “Network & Internet” บริเวณแถบด้านซ้าย และเลือก “Ethernet”

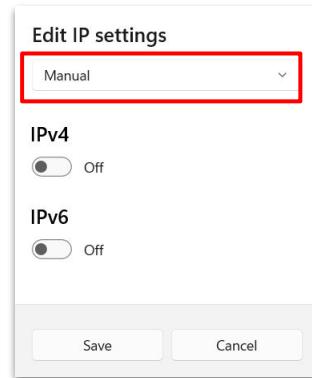




3. กดปุ่ม “Edit” ในตัวเลือก “IP assignment”



4. เปลี่ยนตัวเลือกจาก “DHCP” เป็น “Manual”





5. ปรับ IPv4 ให้เป็น “On”
 - 5.1. กำหนด “IP Address” ของเครื่องผู้เรียนเป็นวงแลน 10.1.1.xxx
 - 5.2. กำหนด “Subnet mask” เป็น 255.255.255.0
 - 5.3. 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

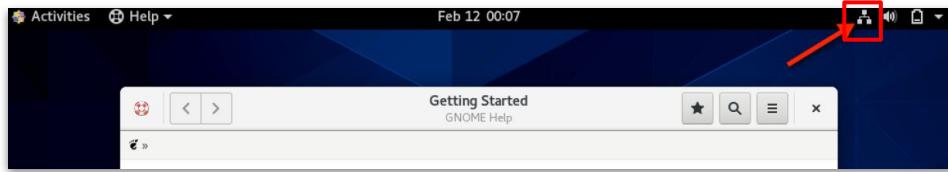
Save **Cancel**

**หมายเหตุ

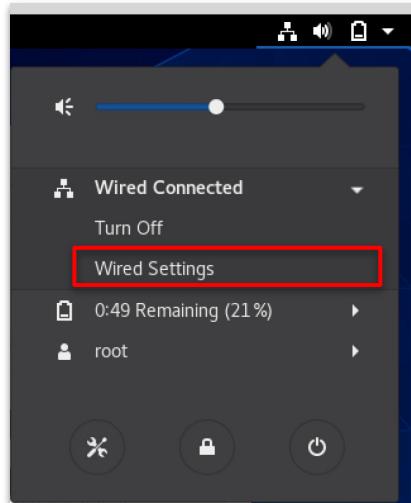
10.1.1.xxx หมายถึง ให้ผู้เรียนใส่ชุดหมายเลขใด ๆ โดยไม่จำกันและห้ามใช้ชุดหมายเลข 10.1.1.1, 10.1.1.212, 10.1.1.165, 10.1.1.166, และ 10.1.1.224 เนื่องจากชุดหมายเลขดังกล่าวถูกกำหนดไว้ในการเขื่อมต่อเครื่องประมวลผลและกล้อง



- ตรวจสอบการเชื่อมต่อสายแลนระหว่างเครื่องของผู้เรียนกับ POE Switch เดียวกันกับเครื่องประมวลผล จากนั้นคลิกไอคอน “Network” บริเวณมุมบนด้านขวา



- คลิกที่ “Wire Connected” แล้วเลือก “Wired Settings”



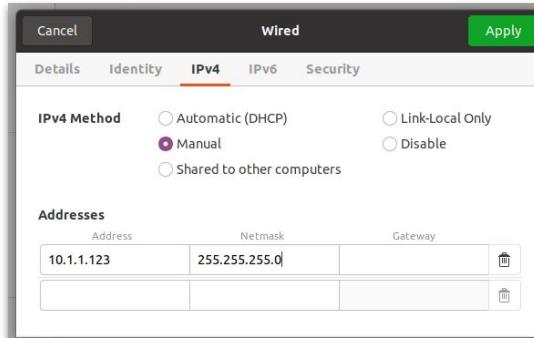


3. คลิกปุ่มรูปฟันเฟืองในตัวเลือก “Wired”



4. คลิกเลือก “IPv4”

- 4.1. เปลี่ยน checkbox จาก “Automatic (DHCP)” เป็น “Manual”
- 4.2. กำหนด “Address” ของเครื่องผู้เรียนเป็นวงแลน 10.1.1.xxx
- 4.3. กำหนด “Netmask” เป็น 255.255.255.0
- 4.4. คลิก “Apply”

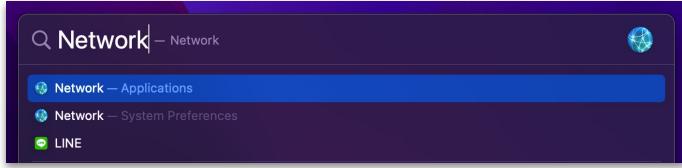


**หมายเหตุ

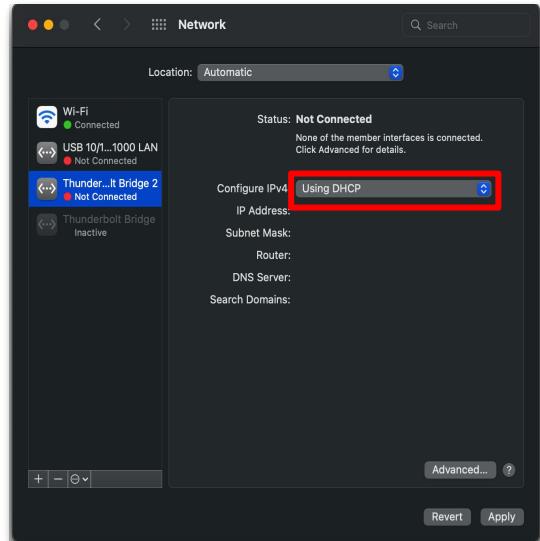
10.1.1.xxx หมายถึง ให้ผู้เรียนใส่ชุดหมายเลขใด ๆ โดยไม่จำกัดและห้ามใช้ชุดหมายเลข 10.1.1.1, 10.1.1.212, 10.1.1.165, 10.1.1.166, และ 10.1.1.224 เนื่องจากชุดหมายเลขดังกล่าวถูกนำไปใช้ในการเขื่อมต่อเครื่องประมวลผลและกล้อง



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



- เลือกการเชื่อมต่อผ่านสาย LAN และเปลี่ยนตัวเลือก Configure IPv4 ให้เป็น “Manually”





3. กำหนด “IP Address” ของเครื่องผู้เรียนเป็นว่างแลน 10.1.1.xxx และ “Subnet Mask” เป็น 255.255.255.0 จากนั้นคลิก “Apply”

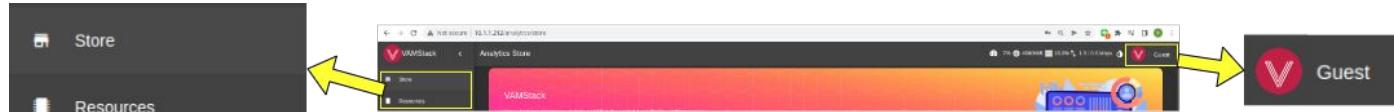


หมายเหตุ

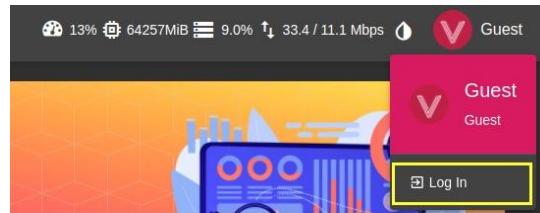
10.1.1.xxx หมายถึง ให้ผู้เรียนใส่ชุดหมายเลขใด ๆ โดยไม่ซ้ำกันและห้ามใช้ชุดหมายเลข 10.1.1.1, 10.1.1.212, 10.1.1.165, 10.1.1.166, และ 10.1.1.224 เนื่องจากชุดหมายเลขดังกล่าวถูกกำหนดไว้ในการเชื่อมต่อเครื่องประมวลผลและกล้อง



- เข้าสู่หน้า VAM Platform โดยพิมพ์ “10.1.1.20” บนเบราว์เซอร์ จะปรากฏหน้าเริ่มต้นการเข้าใช้งานของ VAM Platform ในสถานะผู้เยี่ยมชม (Guest)



- คลิกที่สถานะการใช้งานบริเวณมุมบนด้านขวาและเลือก “Log In”



- เข้าสู่ระบบโดยการใช้บัญชีที่ทางทีมผู้พัฒนาจัดเตรียมให้และคลิก “Log in”

- username: team01, password: study01
- username: team02, password: study02
- username: team03, password: study03
- username: team04, password: study04
- username: team05, password: study05



VAMStack

- Store
- Assignment
- Report
- System Configuration
- Sources Management
- Users Management
- Resources

Analytics Store

6% 4913MiB 18.0% 1.3 / 0.0 Mbps team01

**VAM FACE MASK DETECTION**

Face Mask Detection is the one most popular analytics that can detect mask on any face using deep learning technique. Because this analytic can be used in screening people task for reducing and preventing COVID-19 disease.

[Apply to Camera](#)**VAM HUMAN DETECTION AND ANALYTICS**

Human Detection and Capture is the one of analytics in VAM Platform. It's used for detecting and cropping many people in the image and make data values in the future.

[Apply to Camera](#)**VAM HUMAN SUIT COLOR DETECTION**

Human Suit Color Detection is an analytics to find color cloth or jeans in the people image including, firstly, detection upper body and lower body of the human image, finally, classify color of

[Apply to Camera](#)**VAM PEOPLE COUNTING IN ROI**

People Counting in ROI is an analytics to count and crop people in Region of Interest (ROI). Moreover, the user can make and edit ROI for counting people with themselves.

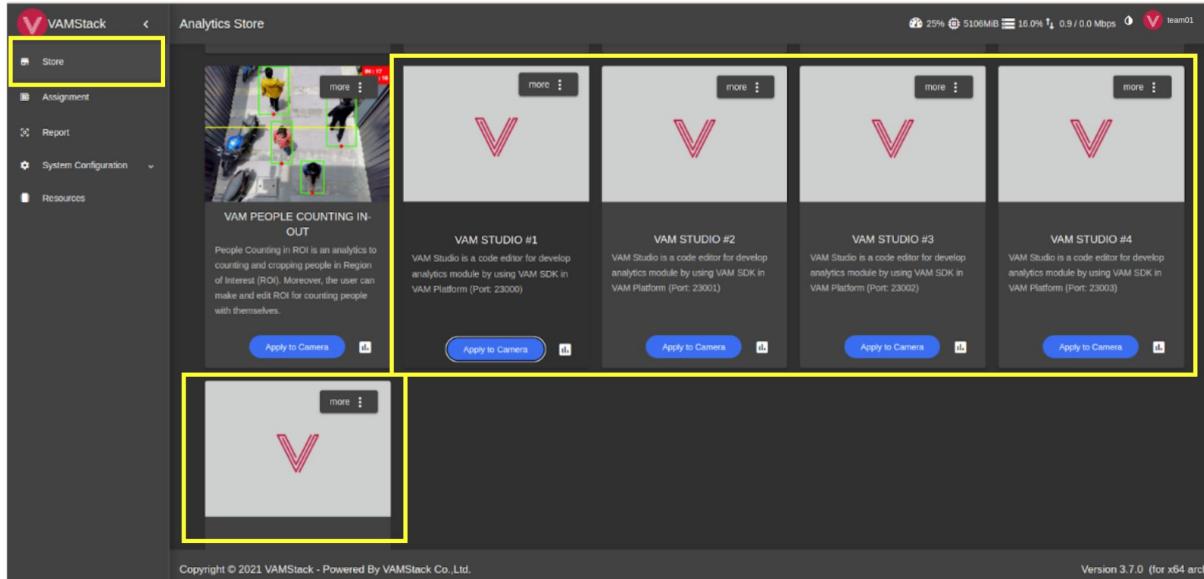
[Apply to Camera](#)**VAM INTRUDER DETECTION AND MONITORING**

Intruder Detection and Monitoring is a simple monitoring intruder application. It can be used for monitoring intruder situations and alarms.

[Apply to Camera](#)



- เข้าสู่หน้า VAM Platform และให้คลิก “Store” เพื่อค้นหาการ์ด “VAM STUDIO#X”





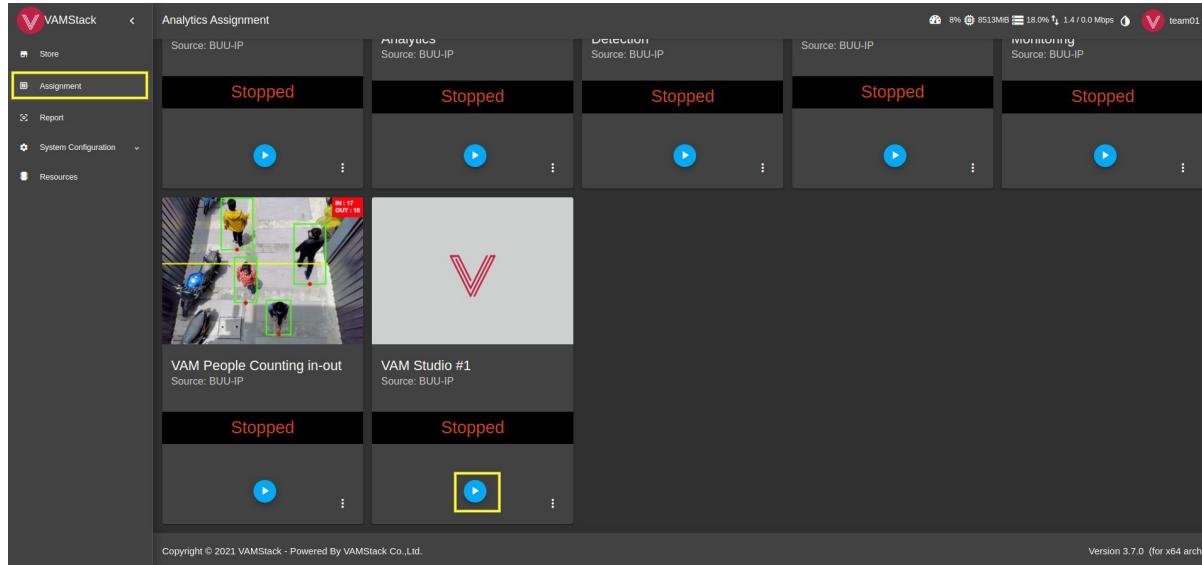
2. คลิกปุ่ม “Apply to Camera” ของкар์ด “VAM STUDIO#X” เพื่อผูกกล้อง โดยมีเงื่อนไขดังนี้
 - 2.1. บัญชี team01 ใช้การ์ด “VAM STUDIO#1”
 - 2.2. บัญชี team02 ใช้การ์ด “VAM STUDIO#2”
 - 2.3. บัญชี team03 ใช้การ์ด “VAM STUDIO#3”
 - 2.4. บัญชี team04 ใช้การ์ด “VAM STUDIO#4”
 - 2.5. บัญชี team05 ใช้การ์ด “VAM STUDIO#5”

The screenshot shows the VAM Stack Analytics Store interface. On the left, there's a sidebar with options like Site, Assignment, Report, System Configuration, and Resources. The main area displays five cards for different VAM STUDIO modules:

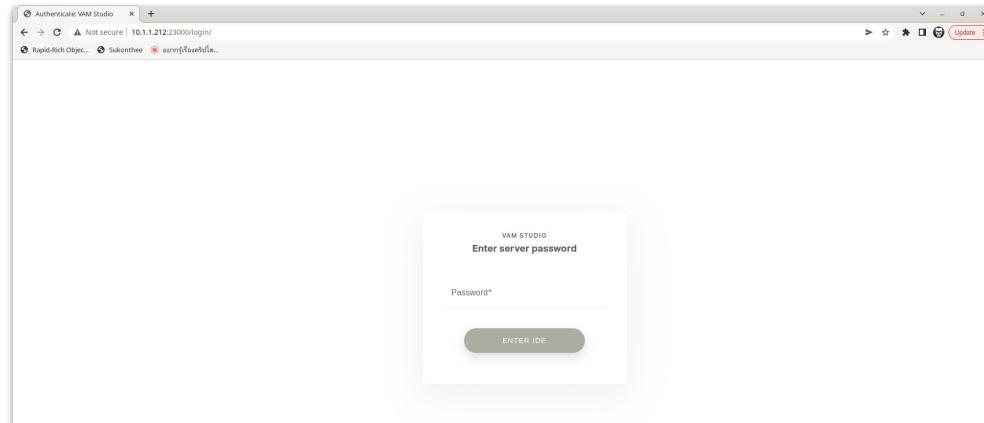
- VAM PEOPLE COUNTING IN-OUT**: People Counting in ROI is an analytics module for counting and tracking people in Region of Interest (ROI). It includes a preview image showing two people walking, and a button labeled "Apply to Camera".
- VAM STUDIO #1**: VAM Studio is a code editor for developing analytics module by using VAM SDK in VAM Platform (Port: 23001).
- VAM STUDIO #2**: VAM Studio is a code editor for developing analytics module by using VAM EDAK in VAM Platform (Port: 23001).
- VAM STUDIO #3**: VAM Studio is a code editor for developing analytics module by using VAM EDAK in VAM Platform (Port: 23001).
- VAM STUDIO #4**: VAM Studio is a code editor for developing analytics module by using VAM SDK in VAM Platform (Port: 23003). This card is highlighted with a yellow box, and an arrow points from it to a larger, detailed view on the right.

The bottom of the screen shows copyright information: Copyright © 2021, VAMStack - Powered By VAMStack Co., Ltd. Version 3.7.0 (for x64 arch).

3. หลังผูกกล้องสำเร็จ ให้คลิกเมนู “Assignment” และคลิกที่ปุ่มการทำงานของ “VAM STUDIO#X” เพื่อเปลี่ยนสถานะการทำงาน



4. เมื่อ “VAM STUDIO#X” เริ่มทำงาน ให้เปิดหน้าต่างในเบราว์เซอร์ใหม่อีก 1 หน้าและทำการพิมพ์ url เพื่อเข้าสู่ระบบ VAM Studio ดังนี้
 - 4.1. การ์ด “VAM STUDIO#1” ใช้ localhost:23000 หรือ 10.1.1.212:23000
 - 4.2. การ์ด “VAM STUDIO#2” ใช้ localhost:23001 หรือ 10.1.1.212:23001
 - 4.3. การ์ด “VAM STUDIO#3” ใช้ localhost:23002 หรือ 10.1.1.212:23002
 - 4.4. การ์ด “VAM STUDIO#4” ใช้ localhost:23003 หรือ 10.1.1.212:23003
 - 4.5. การ์ด “VAM STUDIO#5” ใช้ localhost:23004 หรือ 10.1.1.212:23004
5. เมื่อเข้าสู่หน้า url สำเร็จ จะปรากฏหน้าให้ใส่รหัสผ่านเข้าใช้งาน ซึ่งทุก url จะใช้รหัสเดียวกันคือ P@ssw0rd





The screenshot displays the VAM Studio interface, which is a dark-themed code editor. The top navigation bar includes File, Edit, Selection, View, Go, Debug, Terminal, and Help. The title bar shows "Welcome - app - vam-studio".

EXPLORER sidebar:

- OPEN EDITORS
- TS Welcome
- APP
 - examples_code
 - examples_image
 - models
 - modules
 - protos
 - vam
 - app.py

Welcome - app - vam-studio main area:

VAM Studio

AI เพื่อการเขียนโค้ด

Start

- New file...
- Open folder...
- Add workspace folder...

Recent

No recent folders

Customize

- Tools and languages
 - Install support for [JavaScript](#), [TypeScript](#), [Python](#), [PHP](#), [Azure](#), [Docker](#) and more
- Settings and keybindings
 - Install the settings and keyboard shortcuts of [Vim](#), [Sublime](#), [Atom](#) and others
- Color theme
 - Make the editor and your code look the way you love

Help

- [Printable keyboard cheatsheet](#)
- [Introductory videos](#)
- [Tips and Tricks](#)
- [Product documentation](#)
- [GitHub repository](#)
- [Stack Overflow](#)

Show welcome page on startup

Learn

- Find and run all commands
 - Rapidly access and search commands from the Command Palette (Ctrl+Shift+P)
- Interface overview
 - Get a visual overlay highlighting the major components of the UI
- Interactive playground
 - Try essential editor features out in a short walkthrough



พักรักหน่อย มาเล่นเกมกันเถอะ

```
# 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
```



1. พัฟก์ชันสำหรับการรับข้อมูลของกล้องที่อยู่ใน VAM Platform
 - 1.1. listSource
 - 1.2. get_vamsnapshot
 - 1.3. get_vamroi
 - 1.4. draw_roi
2. พัฟก์ชันพื้นฐานสำหรับการประมวลผลและวิเคราะห์ข้อมูล
 - 2.1. people_detection
 - 2.2. people_counting
3. พัฟก์ชันสำหรับการส่งข้อมูลที่ประมวลผลเข้า VAM Platform
 - 3.1. set_vamlivefigure
 - 3.2. set_data2vamMetadata

listSource

The screenshot shows the VAM Studio IDE interface. The top menu bar includes File, Edit, Selection, View, Go, Debug, Terminal, and Help. The left sidebar has sections for EXPLORER, APP, and PROBLEMS. The APP section contains several Python files and image files. The central workspace shows two open editors: 'example_01.py' and 'example_mv_01.py'. The code in 'example_01.py' is as follows:

```
1 #import vamsdk for get listsources examples
2 from vamsdk.source import SourceManagement
3 import pprint
4
5 #calling source management sdk
6 sm = SourceManagement()
7
8 #calling listsources function
9
10 MACHINE_IP = "10.1.1.212" #machine ip example
11 listsources_data = sm.listSource(machine_ip=MACHINE_IP)
12 pprint.pprint(listsources_data)
```

To the right of the code editor, there is a terminal window with the command:

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

Below the terminal, another terminal window shows the command:

```
python3 example_01.py
```

The bottom status bar indicates the terminal is in bash mode. The footer of the IDE shows the current file is 'example_01.py'.



1. สร้างไฟล์แล้วตั้งชื่อเป็น snapshot-xx.py โดยให้แทน xx ด้วยหมายเลขกลุ่ม เช่น snapshot-01.py และห้ามตั้งชื่อซ้ำกัน
2. คัดลอกชุดโค้ดของโปรแกรมดังต่อไปนี้ลงในไฟล์ที่สร้างไว้

```
# 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)
```

get_vamsnapshot

The screenshot shows a VAM Studio interface with the following components:

- File Explorer:** On the left, it lists project files: `snapshot-01.py`, `results.jpg`, and `snapshot-01.py` (selected).
- Code Editor:** The main editor window contains the following Python code:

```
1 # import vamsdk for get snapshot examples
2 from vam.vamsdk import get_vamsnapshot
3 import cv2
4
5 # SDK: Platform get camera source
6 (img, assignmentId, sourceId, ts) = get_vamsnapshot()
7
8
9 # return result from get vam snapshot sdk
10 cv2.imwrite("results.jpg", img)
```
- Terminal:** At the bottom, the terminal window shows the command and its execution:

```
root@d933f59ec2db:/usr/src/app# python3 snapshot-01.py
2022-06-16 11:40:34.015 | DEBUG    | modules.analytics_user_authenticator:_check_existing_user:56 - Matched user vamanalytics7
2022-06-16 11:40:34.140 | INFO     | vam.vamsdk:get_vamsnapshot:60 - Get snapshot status: Done
root@d933f59ec2db:/usr/src/app#
```
- Image Preview:** On the right, there is a preview of the captured image named `results.jpg`, which shows a person in a room with tables and chairs.
- Bottom Status Bar:** The status bar at the bottom indicates: Ln 10, Col 32, Spaces: 4, UTF-8, LF, Python.



1. ตั้งค่าพื้นที่ที่สูงในกล้องที่ผูกอยู่กับการ์ด “VAM STUDIO#X” ในหน้า “Assignment”
2. สร้างไฟล์แล้วตั้งชื่อเป็น drawROI-xx.py โดยให้แทน xx ด้วยหมายเลขกลุ่ม เช่น drawROI-01.py
3. คัดลอกชุดโค้ดของโปรแกรมดังต่อไปนี้ลงในไฟล์ที่สร้างไว้

```
# 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)
```

get_vamroi & draw_roi

File Edit Selection View Go Debug Terminal Help

results-ROI.jpg - app - vam-studio

EXPLORER

OPEN EDITORS

APP

- examples_code
 - ai-as-a-service
 - basic-opencv
 - basic-sdk
 - classification
 - data_visualization
 - object_detection
- examples_image
- models
- modules
- protos
- vam
- app.py
- drawROI-01.py

results-ROI.jpg

results.jpg

snapshot-01.py

drawROI-01.py

```
1 # import vamsdk for get roi examples
2 from vam.vamsdk import get_vamsnapshot
3 from vam.vamsdk import get_vamroi
4 from vam.vamsdk import draw_roi
5
6 import cv2
7 import os
8
9 # SDK: Platform get camera source
10 (img, assignmentId, sourceId, ts) = get_vamsnapshot()
11
12 # SDK: Calling roi data from platfrom
13 (confRoiStatus, confRoiList) = get_vamroi(assignmentId=assignmentId)
14 print(confRoiList)
15
16 # draw roi sdk
17 roi_frame = draw_roi(img, confRoiList)
18 cv2.imwrite("results-ROI.jpg", roi_frame)
```

RESULTS

1970-01-17 Sat 17:46:42

Zone 1

24x7 Care Center

python3 drawROI-xx.py

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
root@d933f59ec2db:/usr/src/app# python3 drawROI-01.py
2022-06-16 11:54:10.558 | DEBUG | modules.analytics_user_authenticator:_check_existing_user:56 - Matched user vamanalytics7
2022-06-16 11:54:10.727 | INFO | vam.vamsdk:get_vamsnapshot:60 - Get snapshot status: Done
2022-06-16 11:54:10.948 | DEBUG | modules.analytics_user_authenticator:_check_existing_user:56 - Matched user vamanalytics7
[{'source_id': 251, 'assignment_id': 21, 'coordinate': [[[35, 505], [34, 80], [936, 94], [924, 513]]], 'name': 'Zone 1', 'width': 970, 'height': 582}]
root@d933f59ec2db:/usr/src/app#
```

OUTLINE

0 0 0 Whole Image 1920x1080 339.86KB



- สร้างไฟล์แล้วตั้งชื่อเป็น people-detection-xx.py โดยให้แทน xx ด้วยหมายเลขกลุ่ม เช่น people-detection-01.py
- คัดลอกชุดโค้ดของโปรแกรมดังต่อไปนี้ลงในไฟล์ที่สร้างไว้

```
# 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)
```

people_detection

File Edit Selection View Go Debug Terminal Help detection-01.py - app - vam-studio

EXPLORER

- OPEN EDITORS
- APP
 - examples_code
 - examples_image
 - models
 - modules
 - protos
 - vam
 - app.py
 - detection-01.py
 - drawROI-01.py
 - results-ROI.jpg
 - results.jpg
 - snapshot-01.py
 - test.png

detection-01.py x

```
1 # import vamsdk for set get analytics results examples
2 from vam.vamsdk import people_detection
3
4 import cv2
5 # select image
6 IMAGE_PATH = "/../../../../examples_image/person.png"
7 image = cv2.imread(IMAGE_PATH)
8
9 # calling detection people function
10 # SDK: People detection
11 people_bboxes, results_frame_people = people_detection(frame=image, conf=0.6)
12 print(people_bboxes)
13
14 # imwrite people detection results
15 cv2.imwrite(["test.png", results_frame_people])
```

test.png x



python3 people-detection-xx.py

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
root@d933f59ec2db:/usr/src/app# python3 detection-01.py
[(58, 76, 208, 372, 0.872821033000946)]
root@d933f59ec2db:/usr/src/app# []
```

1: bash

Ln 15, Col 46 Spaces: 4 UTF-8 LF Python



1. สร้างไฟล์แล้วตั้งชื่อเป็น people-cnt-xx.py โดยให้แทน xx ด้วยหมายเลขกลุ่ม เช่น people-cnt-01.py
2. คัดลอกชุดโค้ดของโปรแกรมดังต่อไปนี้ลงในไฟล์ที่สร้างไว้

```
# 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_bboxes)
print(cnt)
```

people_counting

The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows a tree view of files and folders. The "APP" folder is expanded, showing "examples_code", "examples_image", "models", "modules", "proto", "vam", and "app.py". "counting-01.py" is selected and highlighted.
- Code Editor:** Displays the content of "counting-01.py". The code imports vamsdk, selects an image file ("person.png"), and calls the "people_detection" function from vamsdk to find bounding boxes. It then uses the "objects_counting" function to count the objects (people) in the frame and prints the result.

```
File Edit Selection View Go Debug Terminal Help
counting-01.py - app - vam-studio

EXPLORER
OPEN EDITORS
APP
examples_code
examples_image
models
modules
proto
vam
app.py
counting-01.py
detection-01.py
drawROI-01.py
results-ROI.jpg
results.jpg
snapshot-01.py
test.png

1 # import vamsdk for set get analytics results examples
2 from vam.vamsdk import people_detection
3 from vam.vamsdk import objects_counting
4 import cv2
5
6 # select image
7 IMAGE_PATH = "examples_image/person.png"
8 image = cv2.imread(IMAGE_PATH)
9
10 # calling counting people function
11 # SDK: People detection
12 people_bboxes, results_frame_people = people_detection(frame=image, conf=0.6)
13
14 # SDK: Counting object with bounding box
15 cnt = objects_counting(people_bboxes)
16 print([cnt])
```

python3 people-counting-xx.py

The screenshot shows a terminal window with the following details:

- Terminal Tab:** The "TERMINAL" tab is selected.
- Output:** The terminal shows the command "python3 counting-01.py" being run by a root user. The output is "1", indicating one person was detected.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: bash
root@d933f59ec2db:/usr/src/app# python3 counting-01.py
1
root@d933f59ec2db:/usr/src/app#
```





1. สร้างไฟล์แล้วตั้งชื่อเป็น vamlivefigure-xx.py โดยให้แทน xx ด้วยหมายเลขกลุ่ม เช่น vamlivefigure-01.py
2. คัดลอกชุดโค้ดของโปรแกรมดังต่อไปนี้ลงในไฟล์ที่สร้างไว้

```
# 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)
```

set_vamlivefigure

The screenshot shows the Vam Studio IDE interface. The top bar includes File, Edit, Selection, View, Go, Debug, Terminal, and Help menus. The title bar indicates the current file is vamlivefigure-01.py - app - vam-studio.

The Explorer sidebar on the left lists project files under APP, including examples_code, ai-as-a-service, basic-opencv, basic-sdk, classification, data_visualization, object_detection, examples_image, models, modules, protos, vamlivefigure-01.py, app.py, counting-01.py, detection-01.py, drawROI-01.py, results-ROI.jpg, results.jpg, snapshot-01.py, test.png, and vamlivefigure-01.py.

The main code editor window displays the following Python script:

```
# 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)
```

A terminal window titled "python3 vamlivefigure-xx.py" shows the execution of the script with the following log output:

```
root@d933f59ec2db:/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   | vam.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   | vam.vamsdk:set_vamlivefigure:84 - Live figure status: OK
root@d933f59ec2db:/usr/src/app#
```

The bottom status bar shows Ln 15, Col 73, Spaces: 4, UFT-8, LF, Python, and a small icon.



vamstack < Analytics Raw Output

ANALYTICS OUTPUT

LIVE FIGURE

Processed Images

05-28-2022 Sat 19:05:21

Camera 01

Copyright © 2022 vamstack - Powered By VAMStack Co.,Ltd.

Version 2022.03-01 (for x64 arch.)

25% 13242MiB 15.0% 0.8 / 0.0 Mbps admin

Store Assignment Train Model Report System Configuration Resources

A screenshot of the VAM Stack web application. On the left, a sidebar lists navigation items: Store, Assignment, Train Model, Report, System Configuration (with a dropdown arrow), and Resources. The main content area has two tabs: 'ANALYTICS OUTPUT' (selected) and 'LIVE FIGURE'. The 'ANALYTICS OUTPUT' tab shows a sub-section titled 'Processed Images'. The 'LIVE FIGURE' tab displays a live video feed from 'Camera 01'. The video shows a room with wooden walls and a glass door. A person is seated at a desk on the right side of the frame. A green rectangular box highlights the person. The timestamp '05-28-2022 Sat 19:05:21' is displayed above the video. In the top right corner of the interface, there are system status icons (battery at 25%, RAM usage, CPU usage, network speed) and a user profile for 'admin'. The bottom of the screen includes copyright information and a version number.



1. สร้างไฟล์แล้วตั้งชื่อเป็น set-data-xx.py โดยให้แทน xx ด้วยหมายเลขกลุ่ม เช่น set-data-01.py
2. คัดลอกชุดโค้ดของโปรแกรมดังต่อไปนี้ลงในไฟล์ที่สร้างไว้

```
# 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"))

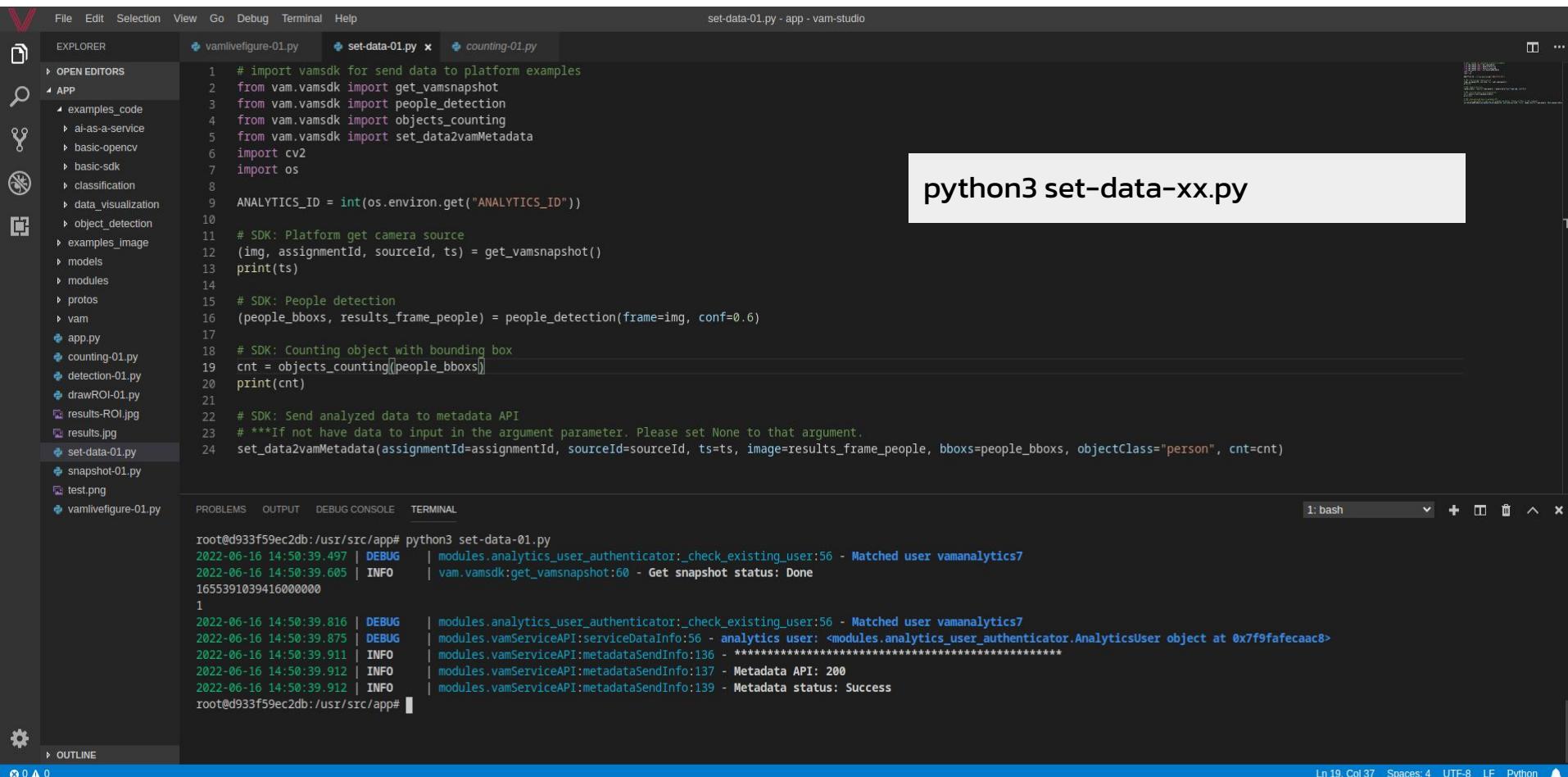
# SDK: Platform get camera source
(img, assignmentId, sourceId, ts) = get_vamsnapshot()
print(ts)

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

# SDK: Counting object with bounding box
cnt = objects_counting(people_bboxes)
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, bboxes=people_bboxes,
objectClass="person", cnt=cnt)
```

set_data2vamMetadata



The screenshot shows a VS Code interface with the following details:

- File Explorer:** Shows files in the current workspace, including `vamlivefigure-01.py`, `set-data-01.py` (the active file), and `counting-01.py`.
- Code Editor:** Displays the content of `set-data-01.py`. The code imports various VAM SDK modules and performs three main operations:
 - SDK: Platform get camera source
 - SDK: People detection
 - SDK: Counting object with bounding boxIt then sends analyzed data to a metadata API.
- Terminal:** Shows the command `python3 set-data-01.py` being run and its output. The output includes DEBUG and INFO log messages from the VAM SDK, indicating successful user authentication, snapshot retrieval, and metadata API calls.
- Status Bar:** Shows the terminal tab is active, and the status bar indicates the file is at line 19, column 37, with 4 spaces, and the language is Python.

```
# import vamsdk for send data to platform examples
from vam.vamsdk import get_vamsnapshot
from vam.vamsdk import people_detection
from vam.vamsdk import objects_counting
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_bboxes, results_frame_people) = people_detection(frame=img, conf=0.6)

# SDK: Counting object with bounding box
cnt = objects_counting([people_bboxes])
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, bboxes=people_bboxes, objectClass="person", cnt=cnt)
```

python3 set-data-xx.py



VAMStack Analytics Raw Output

ANALYTICS OUTPUT | LIVE FIGURE

Processed Images

2022-05-30 09:58:12 2022-05-30T09:57:13.387Z

LIVE FIGURE

05-29-2022 Sun 19:35:58

Camera 01

```
{  
  "ts": "2022-05-30 09:58:12",  
  "analyticsId": 8,  
  "sourceId": 250,  
  "analyticsType": "detection",  
  "probability": 0.871605634689331,  
  "image": {  
    "cropped": "/assets/vamstack/images/analytics/results/8-10-250-165390469252300000.jpeg"  
  },  
  "recognition": null,  
  "data": {  
  }  
}
```

Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd.

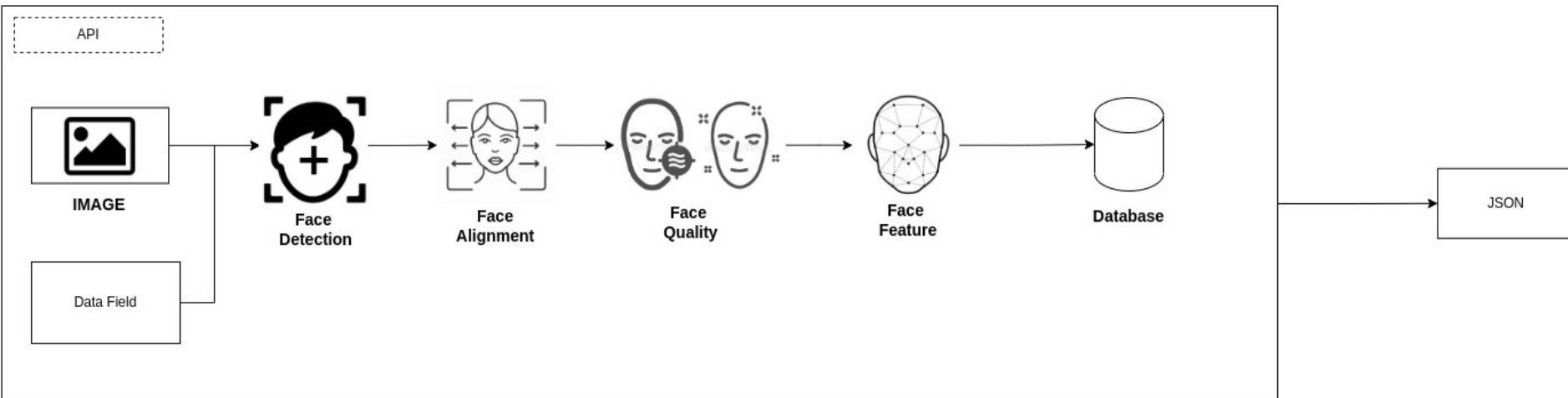
Version 3.7.0 (for x64 arch.)



1. Face Registration
 - 1.1. EnrollFace
2. Face Recognition
 - 2.1. FaceFeature
3. Face Detection and Quality Verification
 - 3.1. CompareFace



ENROLL



EnrollFace

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

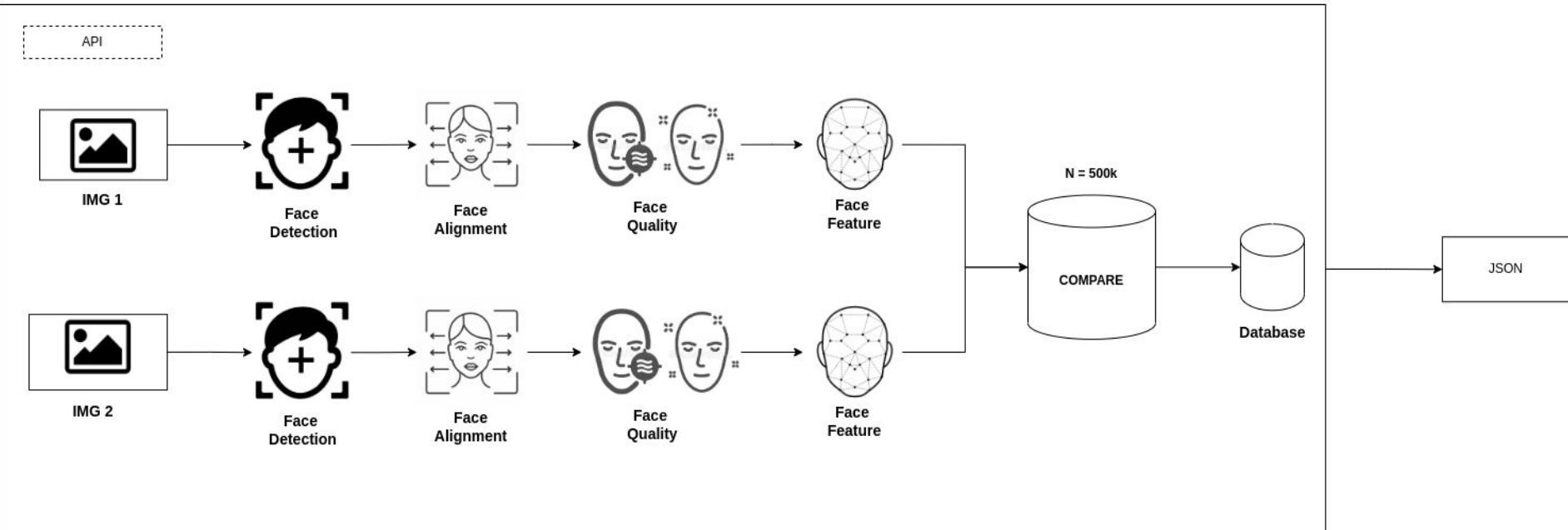
- File Bar:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- Explorer:** Shows the project structure under "OPEN EDITORS" and "APP". The "examples_code" folder contains "ai-as-a-service", "face-api", and "compare.py". The "face-api" folder contains "enroll.py", "feature.py", and several "example_X.py" files (X=1..6). Other folders like "basic-opencv", "basic-sdk", etc., are also listed.
- Code Editor:** The "enroll.py" file is open, displaying Python code for enrolling a face. It imports requests, pprint, and json. It sets parameters for calling the AI service (HOST_IP = "10.1.1.212", PORT = "8500", IMAGE_PATH = "../../../../examples_image/face02.jpg"), defines an endpoint ("EnrollFace"), and specifies a member ID ("test-face"). It then creates a request form with the image path and member data (MemberId: memberId). Finally, it makes a POST request to the service and prints the response status code and JSON results.
- Terminal:** A terminal window titled "1: bash" shows the command "python3 enroll.py" being run. The output indicates a successful enrollment process.
- Bottom Status Bar:** Shows file counts (0), line numbers (Ln 1, Col 1), spaces (Spaces: 2), and file type (UTF-8 LF Python).

Terminal Output (1: bash):

```
root@feb0ad82b42d:/usr/src/app/examples_code# ls
ai-as-a-service basic-opencv basic-sdk classification data_visualization object_detection
root@feb0ad82b42d:/usr/src/app/examples_code# cd ai-as-a-service/
root@feb0ad82b42d:/usr/src/app/examples_code/ai-as-a-service# cd face-api/
root@feb0ad82b42d:/usr/src/app/examples_code/ai-as-a-service/face-api# python3 enroll.py
200
{'message': 'EnrollFaceSuccessful', 'success': True}
root@feb0ad82b42d:/usr/src/app/examples_code/ai-as-a-service/face-api#
```



RECOGNITION



FaceFeature

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

- File Explorer:** Shows the project structure with files like feature.py, example_01.py, and app.py.
- Terminal:** Displays the command `cd examples_code/ai-as-a-service/face-api` and the output of running `python3 feature.py`.
- Code Editor:** The `feature.py` file contains code for calling an AI service to compare faces.
- Output:** The terminal shows the results of the face comparison, including coordinates and a quality score.

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

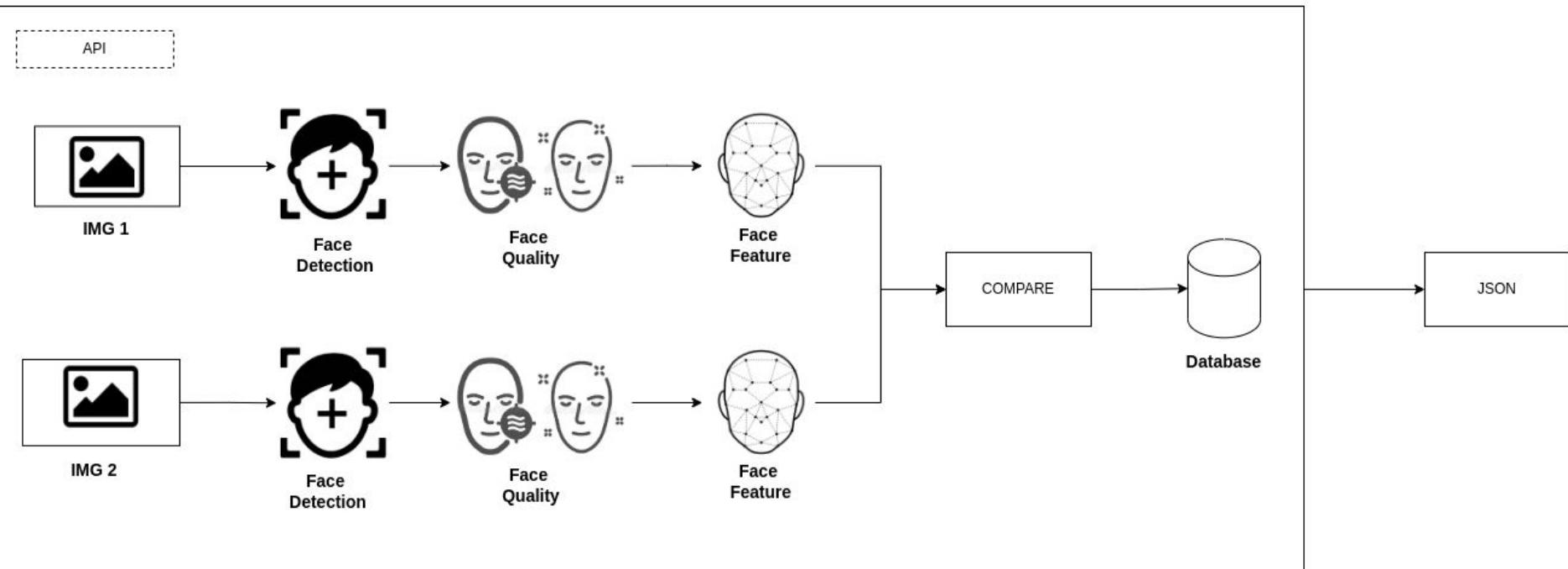
# parameters for calling ai service
HOST_IP = "10.1.1.212"
PORT = "8500"
IMAGE_PATH = "../../examples_image/face-compare01.jpg"
endpoint = "/FaceFeature"

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

0.328125,
-0.265625,
-0.578125,
-0.359375,
-1.0,
-0.46875,
0.28125,
-0.59375,
-0.21875,
-0.875,
0.375,
0.515625,
1.65625,
0.25,
0.234375,
-0.015625,
-0.546875,
0.1875,
-1.21875,
0.265625,
0.609375,
-1.15625,
-0.375,
-1.0625,
-0.640625,
1.0,
-1.140625,
0.609375,
-0.40625,
-0.390625,
0.765625,
-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}}

t: bash

Ln 1, Col 1 Spaces: 2 UTF-8 LF Python

**COMPARE**

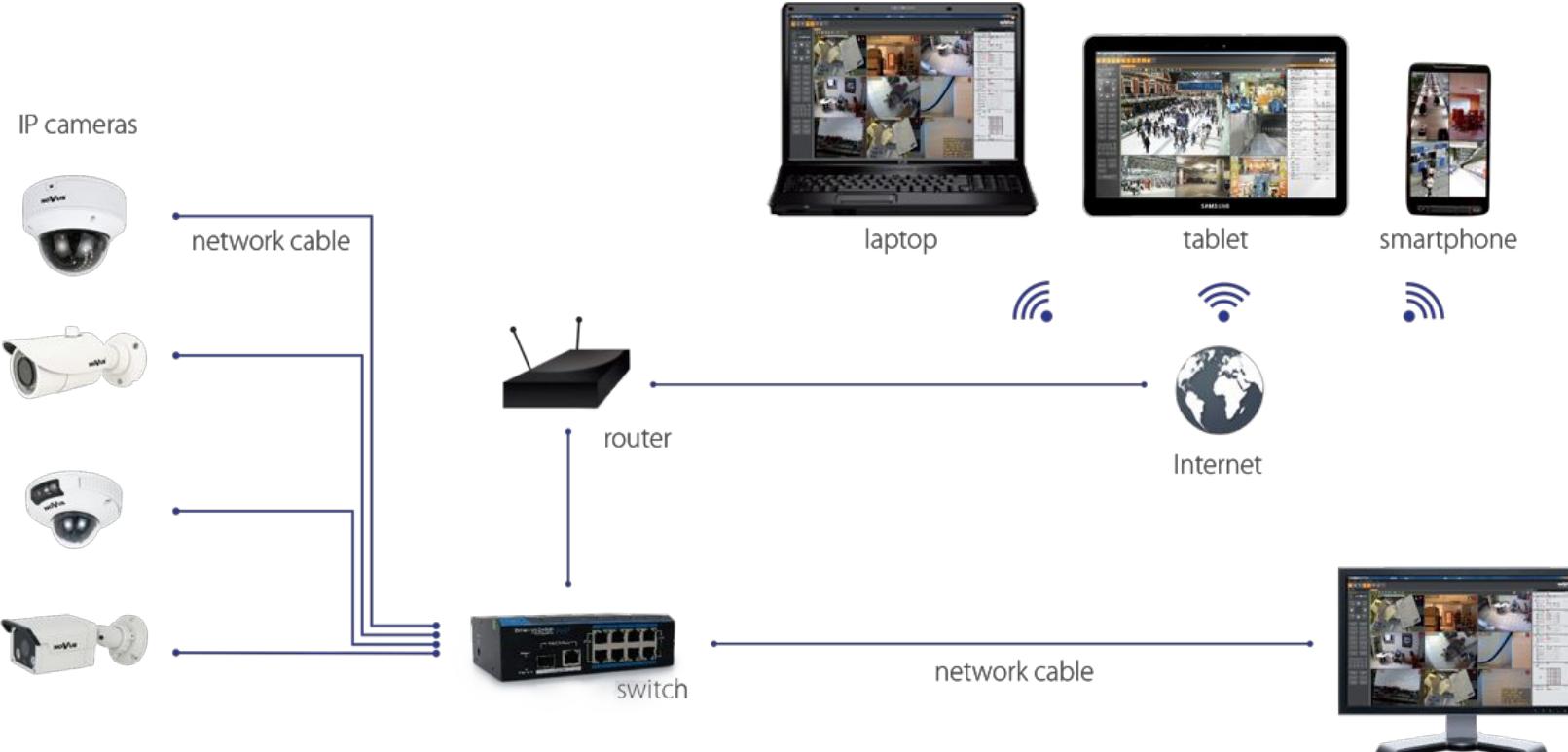
CompareFace

The screenshot shows the Vam Studio IDE interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- Explorer:** Shows the project structure under "OPEN EDITORS" and "APP". The "compare.py" file is selected in the editor.
- Editor:** The "compare.py" file content is displayed:

```
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 = "8500"
9 image01Path = "../../examples_image/face-compare01.jpg"
10 image02Path = "../../examples_image/face-compare01.jpg"
11 endpoint = "/CompareFace"
12
13 request_form = {
14     "ImageA": open(image01Path, "rb"),
15     "ImageB": open(image02Path, "rb")
16 }
17
18 # calling ai service function
19 response = requests.post(f'http://{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())
```
- Terminal:** A terminal window titled "bash" shows the command being run:

```
root@feb0ad82b42d:/usr/src/app# cd examples_code/ai-as-a-service/face-api/
root@feb0ad82b42d:/usr/src/app/examples_code/ai-as-a-service/face-api# python3 compare.py
200
{'score': 99.9989999845351, 'success': True}
root@feb0ad82b42d:/usr/src/app/examples_code/ai-as-a-service/face-api#
```
- Status Bar:** Includes icons for file operations (New, Open, Save, etc.), a search bar, and status information: Ln 1, Col 1, Spaces: 2, UTF-8, LF, Python.



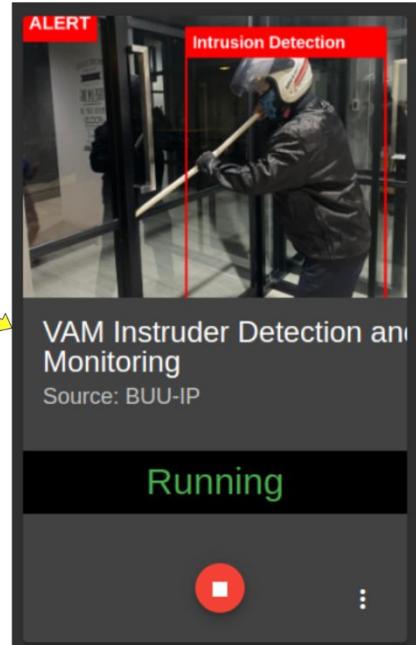


- คลิก “Assignment” ที่แถบเมนูด้านซ้ายของหน้า VAM Platform
- คลิกปุ่มเปลี่ยนสถานะของ “VAM Intrusion Detection” ให้เป็น “Running”

The screenshot shows the VAM Stack Analytics Assignment interface. On the left, there is a sidebar with options: Store, Assignment (selected), Report, System Configuration, and Resources. The main area displays six detection modules:

- VAM Face Mask Detection (Source: BUU-IP): Status: Stopped.
- VAM Human Detection and Analytics (Source: BUU-IP): Status: Stopped.
- VAM Human Suit Color Detection (Source: BUU-IP): Status: Stopped.
- VAM People Counting in ROI (Source: BUU-IP): Status: Stopped.
- VAM Intruder Detection and Monitoring (Source: BUU-IP): Status: Stopped. This module is highlighted with a yellow box and a yellow arrow points from it to the expanded view on the right.
- VAM People Counting in-out (Source: BUU-IP): Status: Stopped.
- VAM Studio #1 (Source: BUU-IP): Status: Stopped.

At the bottom, the text reads: Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd. Version 3.7.0 (for x64 arch.).





3. คลิกการ์ด “VAM Intrusion Detection” ที่มีสถานะเริ่มทำงาน (Running) เพื่อดูการประมวลผลและวิเคราะห์ข้อมูล

VAMStack

- Store
- Assignment
- Report
- System Configuration
- Sources Management
- Users Management
- Resources

Analytics Raw Output

ANALYTICS OUTPUT

LIVE FIGURE

Processed Images

2022-05-31T01:37:10.584Z 2022-05-31T01:37:08.054Z 2022-05-31T01:37:05.484Z

2022-06-17T02:18:07.905Z

ALERT!!

170-1-18 Sun 6:3:10:55

24x Care Center

{
 "resultId": 7486,
 "analyticsType": "detection",
 "probability": 0.953318,
 "data": {
 "trackId": 28,
 "label": "person",
 "fullImg": "/assets/vamstack/images/analytics/results/5-15-1655432287782000000-fullframe.jpg",
 "cnt": 0
 },
 "currentId": 251
}

Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd.

Version 3.7.0 (for x64 arch.)

Intrusion Detection

The screenshot shows the VAM Studio IDE interface. The top navigation bar includes File, Edit, Selection, View, Go, Debug, Terminal, and Help. The left sidebar has sections for EXPLORER, OPEN EDITORS, APP, and various project files like example_01.py, example_02.py, etc. The main editor area displays a Python script named example_01.py:

```
# 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": 5
}

# 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())
```

The bottom part of the editor shows the JSON response from the API call:

```
{"compute_name": "VAM Intruder Detection and Monitor", "data": {"analyticsId": 5, "analyticsResult": {"cnt": 0, "notification": True, "objsInfo": {"croppedImg": "/assets/vamstack/images/analytics/results/5-4-250-165336008074000000.jpeg", "label": "person", "probability": 0.6050128936767578, "roiName": None, "trackId": 10, "x1": 1300, "x2": 1420, "y1": 422, "y2": 604}}, "assignmentId": 4, "fullImg": "/assets/vamstack/images/analytics/results/5-4-165336008074000000-fullframe.jpg", "imgDim": {"height": 1080, "width": 1920}, "sourceId": 250, "sourceName": "BDH-VAM"}, "origin_name": "VAMSTACK-ONPREMISED", "time": "2022-05-24T02:41:20.908594+00:00"}]
```

The terminal at the bottom shows the command being run:

```
root@feb0ad82b42d:/usr/src/app/examples_code/ai-as-a-service#
```

cd examples_code/ai-as-a-service

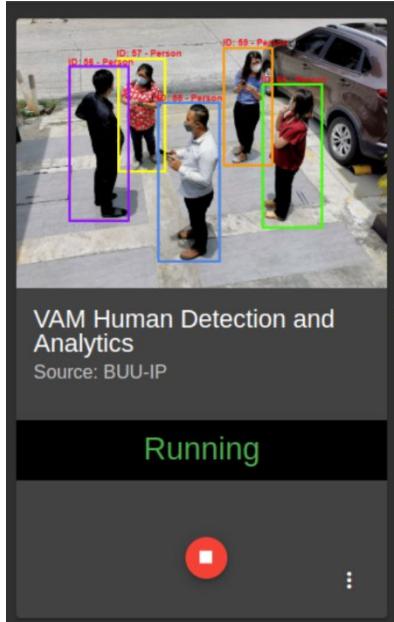
python3 example_01.py

1: bash

Ln 1, Col 1 Spaces: 2 UTF-8 LF Python



- คลิก “Assignment” ที่แถบเมนูด้านซ้ายของหน้า VAM Platform
- คลิกปุ่มเปลี่ยนสถานะของ “VAM Human Detection” ให้เป็น “Running”



VAM Stack

Analytics Assignment

Assignment 1

VAM Face Mask Detection
Source: BUU-IP
Stopped

VAM Human Detection and Analytics
Source: BUU-IP
Stopped

VAM Human Suit Color Detection
Source: BUU-IP
Stopped

VAM People Counting in ROI
Source: BUU-IP
Stopped

VAM Intruder Detection and Monitoring
Source: BUU-IP
Stopped

VAM Studio #1
Source: BUU-IP
Stopped

Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd.

Version 3.7.0 (for x64 arch.)



3. คลิกที่การ์ด “VAM Human Detection” เพื่อดูการประมวลผลและวิเคราะห์ข้อมูล

VAMStack Analytics Raw Output

Analytics Output

LIVE FIGURE

Processed Images

1970-01-18 Sun 08:05:31

ID: 15

24x7 Care Center

Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd.

Version 3.7.0 (for x64 arch.)

The screenshot displays the VAMStack interface for "Human Detection". On the left, a sidebar lists navigation options: Store, Assignment, Report, System Configuration, Sources Management, Users Management, and Resources. The main area is divided into two sections: "Analytics Raw Output" and "LIVE FIGURE".

Analytics Raw Output: This section shows a grid of processed images. Each image includes a timestamp and a unique ID. The images are categorized by date:

- 2022-06-17T01:35:17.083Z (Top Left)
- 2022-06-17T01:37:08.596Z (Top Right)
- 2022-05-31T01:34:32.219Z (Second Row, Left)
- 2022-05-31T01:37:10.598Z (Second Row, Right)
- 2022-05-31T01:37:07.517Z (Third Row, Left)
- 2022-05-31T01:37:07.517Z (Third Row, Right)

LIVE FIGURE: This section shows a live video feed from a camera. A person wearing a white t-shirt and a face mask is highlighted with a blue bounding box. The text "ID: 15" is displayed above the box. The timestamp "1970-01-18 Sun 08:05:31" is shown at the top left of the video frame. The background shows an indoor setting with tables, chairs, and computer monitors. The text "24x7 Care Center" is visible in the bottom right corner of the video frame.

Bottom Content: Below the main interface, there is a JSON-like code snippet showing the detection results:

```
{  
    "resultId": 7476,  
    "analyticsType": "detection",  
    "probability": 0.950269,  
    "data": {  
        "trackId": 15,  
        "label": "person",  
        "fullImg": "/assets/vamstack/images/analytics/results/2-12-1655431960723000000-fullframe.jpg",  
        "cnt": 0  
    },  
    "sourceId": 251  
}
```

Human Detection

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

- File Explorer:** Shows the project structure with files like `example_02.py`, `example_01.py`, and `app.py`.
- Code Editor:** Displays the `example_02.py` file containing Python code for making a request to an AI service for human detection.
- Terminal:** Shows two commands:
 - `cd examples_code/ai-as-a-service`
 - `python3 example_02.py`
- Output:** Shows the JSON response from the AI service, indicating a person was detected with a bounding box and confidence level.
- Status Bar:** Shows the current terminal tab is `1: bash`.

```
example_02.py x
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 = "/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.pprint(response.json())
```

```
'data': {'analyticsId': 2,
  'analyticsResult': {'cnt': 0,
    'objsInfo': {'color_name': 'black',
      'color_rgb': '(31, 29, 33)',
      'croppedImg': '/assets/vamstack/images/analytics/results/2-2-250-165381763023500000.jpeg',
      'label': 'person',
      'probability': 0.7118918895721436,
      'roiName': None,
      'trackId': 46,
      'x1': 1176,
      'x2': 1444,
      'y1': 585,
      'y2': 940}},
  'assignmentId': 2,
  'fullImg': '/assets/vamstack/images/analytics/results/2-2-165381763023500000-fullframe.jpg',
  'imgDim': {'height': 1080, 'width': 1920},
  'sourceId': 250,
  'sourceName': 'BDH-VAM',
  'origin_name': 'VAMSTACK-ONPREMISED',
  'time': '2022-05-29T09:45:06.999606+00:00'}]
```



- คลิก “Assignment” ที่แถบเมนูด้านซ้ายของหน้า VAM Platform
- คลิกปุ่มเปลี่ยนสถานะของ “VAM Human Suit Color Detection” ให้เป็น “Running”

VAMStack Analytics Assignment

Assignment 1

Report System Configuration Resources

VAM Face Mask Detection Source: BUU-IP
Stopped

VAM Human Detection and Analytics Source: BUU-IP
Stopped

VAM Human Suit Color Detection Source: BUU-IP
Stopped 2

VAM People Counting in RC1 Source: BUU-IP
Stopped

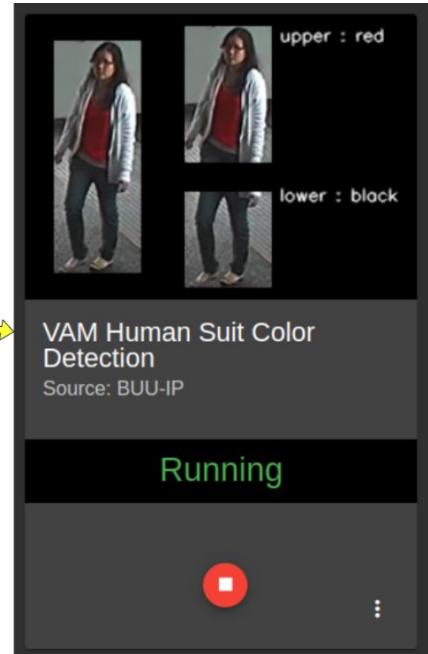
VAM Intruder Detection and Monitoring Source: BUU-IP
Stopped

VAM People Counting in-out Source: BUU-IP
Stopped

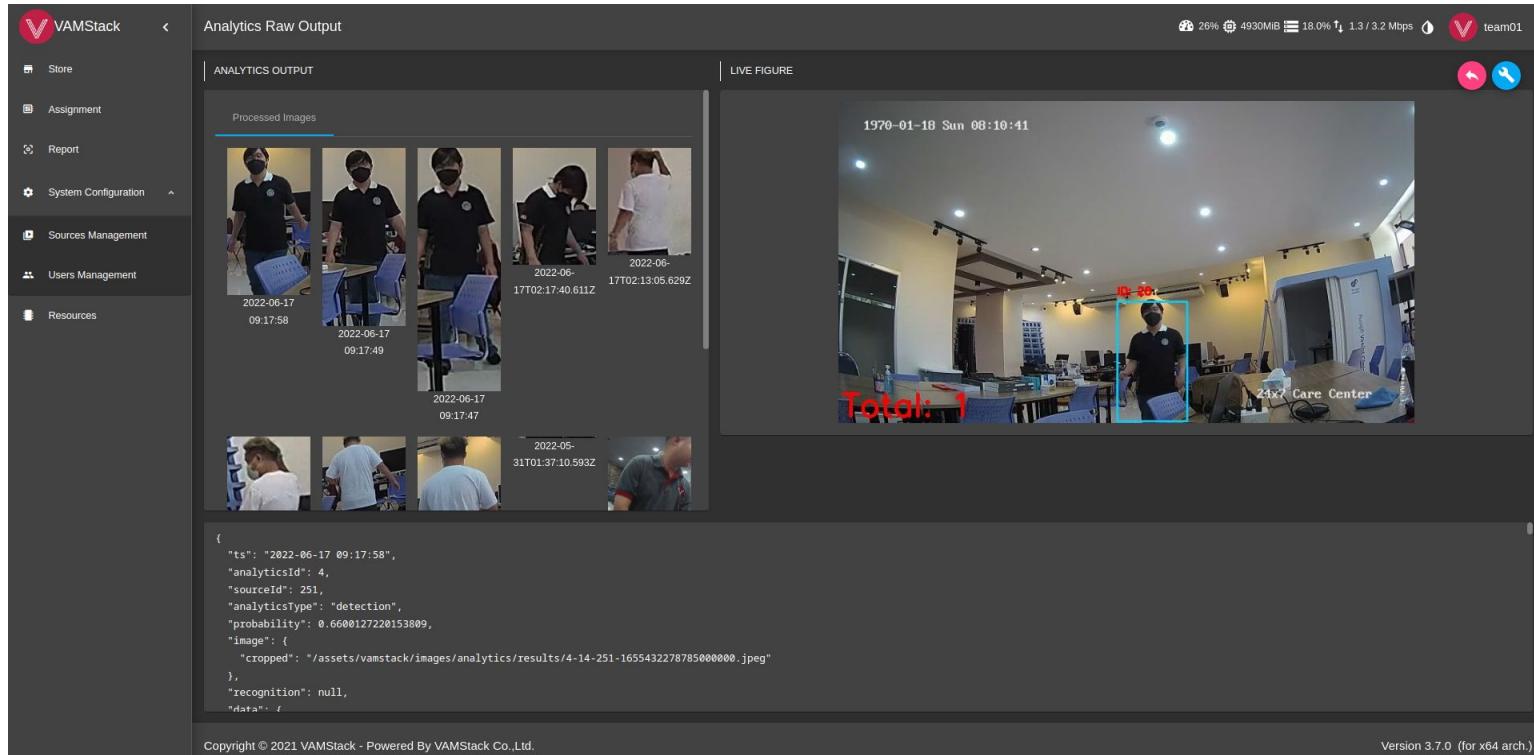
VAM Studio #1 Source: BUU-IP

Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd.

Version 3.7.0 (for x64 arch.)



3. คลิกที่การ์ด “VAM Human Suit Color Detection” เพื่อดูการประมวลผลและวิเคราะห์ข้อมูล



The screenshot shows the VAMStack interface for "Analytics Raw Output". On the left, there's a sidebar with navigation options: Store, Assignment, Report, System Configuration, Sources Management, Users Management, and Resources. The main area has two sections: "ANALYTICS OUTPUT" and "LIVE FIGURE".

ANALYTICS OUTPUT: This section displays a grid of processed images under the heading "Processed Images". The images show various individuals in different settings, with timestamps like "2022-06-17 09:17:58", "2022-06-17 09:17:49", and "2022-06-17 09:17:47". Below this grid, there's a JSON snippet showing a single detection record:

```
{  
  "ts": "2022-06-17 09:17:58",  
  "analyticsId": 4,  
  "sourceId": 251,  
  "analyticsType": "detection",  
  "probability": 0.6600127220153809,  
  "image": {  
    "cropped": "/assets/vamstack/images/analytics/results/4-14-251-1655432278785000000.jpeg"  
  },  
  "recognition": null,  
  "data": {}  
}
```

LIVE FIGURE: This section shows a live video feed from a camera. The timestamp in the top right corner is "1970-01-18 Sun 08:10:41". A person in a black shirt and mask is highlighted with a blue bounding box and the ID "ID: 38". The text "Total: 1" is displayed in red at the bottom left of the video frame. In the bottom right corner of the video, there's a watermark that says "24x7 Care Center".

At the bottom of the screen, there are copyright and version information: "Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd." and "Version 3.7.0 (for x64 arch.)".

Human Suit Color Detection

The screenshot shows the VAM Studio IDE interface. The top navigation bar includes File, Edit, Selection, View, Go, Debug, Terminal, and Help. The title bar indicates the current file is example_03.py - app - vam-studio. The left sidebar (EXPLORER) lists open editors (example_03.py, examples_code/ai-as-a-service/app), examples_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-opencv, basic-sdk, classification, data_visualization, object_detection, examples_image, models, modules, protos, vam, and app.py. The main editor area displays the Python code for example_03.py, which imports requests, pprint, and json, and demonstrates calling an AI service for human suit color detection. The right side features two terminal panes: one for navigating to the examples_code directory and another for running the script with python3 example_03.py. The bottom section shows the terminal tab with the command 1: bash and the output of the script execution.

```
# 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.get(f'http://{HOST_IP}:{PORT}{endpoint}', params=request_form)

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


```

```
cd examples_code/ai-as-a-service
python3 example_03.py
```

```
'objsInfo': {'color_name': 'black',
              'color_rgb': '(16, 16, 18)',
              'croppedImg': '/assets/vamstack/images/analytics/results/3-13-251-1653961030394000000.jpeg',
              'label': 'person',
              'probability': 0.6138097047805786,
              'roiName': None,
              'trackId': 5,
              'x1': 624,
              'x2': 1636,
              'y1': -14,
              'y2': 1080},
    'assignmentId': 13,
    'fullImg': '/assets/vamstack/images/analytics/results/3-13-1653961030394000000-fullframe.jpg',
    'imgDim': {'height': 1080, 'width': 1920},
    'sourceId': 251,
    'sourceName': 'BUU-IP',
    'origin_name': 'VAMSTACK-ONPREMISED',
    'time': '2022-05-31T01:37:10.618873+00:00'}]
```



- คลิก “Assignment” ที่แถบเมนูด้านซ้ายของหน้า VAM Platform
- คลิกปุ่มเปลี่ยนสถานะของ “VAM Face Mask Detection” ให้เป็น “Running”



VAMStack

Analytics Assignment

Assignment 1

Report

System Configuration

Resources

VAM Face Mask Detection
Source: BUU-IP
Stopped

VAM Human Detection and Analytics
Source: BUU-IP
Stopped

VAM Human Suit Color Detection
Source: BUU-IP
Stopped

VAM People Counting in ROI
Source: BUU-IP
Stopped

VAM Intruder Detection and Monitoring
Source: BUU-IP
Stopped

VAM People Counting in-out
Source: BUU-IP

VAM Studio #1
Source: BUU-IP

Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd.

Version 3.7.0 (for x64 arch.)



3. คลิกที่การ์ด “VAM Face Mask Detection” เพื่อดูการประมวลผลและวิเคราะห์ข้อมูล

VAMStack Analytics Raw Output ANALYTICS OUTPUT LIVE FIGURE

28% 5004MB 18.0% 1.8 / 5.8 Mbps team01

Processed Images

ANALYTICS OUTPUT

LIVE FIGURE

1970-01-18 Sun 08:04:48

24x7 Care Center

ID: 15

Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd.

Version 3.7.0 (for x64 arch.)

The screenshot shows the VAMStack interface with the 'Analytics Raw Output' card selected. On the left, a sidebar lists 'Store', 'Assignment', 'Report', 'System Configuration', 'Sources Management', 'Users Management', and 'Resources'. The main area has two sections: 'ANALYTICS OUTPUT' and 'LIVE FIGURE'. The 'ANALYTICS OUTPUT' section displays a grid of processed images showing people wearing face masks. Below the images is a JSON object representing the analysis results. The 'LIVE FIGURE' section shows a live video feed from a camera. A person in the video is highlighted with a blue bounding box and labeled 'ID: 15'. The video feed also includes the text '24x7 Care Center'. The top right of the interface shows system status: battery at 28%, memory usage at 5004MB, network speed at 18.0%, and connection details (1.8 / 5.8 Mbps). The bottom of the screen shows copyright information and the version (3.7.0).

```
        "probability": 0.8648382425308228,
        "image": {
          "cropped": "/assets/vamstack/images/analytics/results/1-11-251-1655431915959000000.jpeg"
        },
        "recognition": null,
        "data": {
          "trackId": 15,
          "label": "Mask",
          "fullImg": "/assets/vamstack/images/analytics/results/1-11-1655431915959000000-fullframe.jpg"
        }
      }
```

Face Mask Detection

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

- File Menu:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- EXPLORER:** Shows the project structure with **OPEN EDITORS** containing `example_04.py` and **APP** containing multiple example files (e.g., `example_01.py`, `example_02.py`, `example_03.py`, `example_04.py`, `example_05.py`, `example_06.py`) and various service modules.
- CODE EDITOR:** The `example_04.py` file is open, displaying Python code for face mask detection using an AI service. It imports requests, pprint, and json, sets parameters for the AI service (HOST_IP: "10.1.1.212", PORT: "20000", endpoint: "/face-mask-detection"), creates a request form with machine_ip and limit_data, calls the service, and prints the JSON response.
- TERMINAL:** Two terminal windows are shown:
 - Terminal 1 (bash): `cd examples_code/ai-as-a-service`
 - Terminal 2 (bash): `python3 example_04.py`The output of the second terminal shows the JSON results of the face mask detection, including crop dimensions and confidence levels.
- PROBLEMS, OUTPUT, DEBUG CONSOLE:** Standard IDE tabs for managing build errors, logs, and debugging.
- STATUS BAR:** Shows the current file is `example_04.py`, the line is 1, column 0, and the encoding is UTF-8.



- คลิก “Assignment” ที่แถบเมนูด้านซ้ายของหน้า VAM Platform
- ตั้งค่าพื้นที่ที่สนใจของ “VAM People Counting in ROI”
- คลิกปุ่มเปลี่ยนสถานะของ “VAM People Counting in ROI” ให้เป็น “Running”

VAMStack

Analytics Assignment

1

Assignment

Report

System Configuration

Resources

VAM Face Mask Detection
Source: BUU-IP

Stopped

VAM Human Detection and Analytics
Source: BUU-IP

Stopped

VAM Human Suit Color Detection
Source: BUU-IP

Stopped

VAM People Counting in ROI
Source: BUU-IP

Stopped

VAM Instruder Detection and Monitoring
Source: BUU-IP

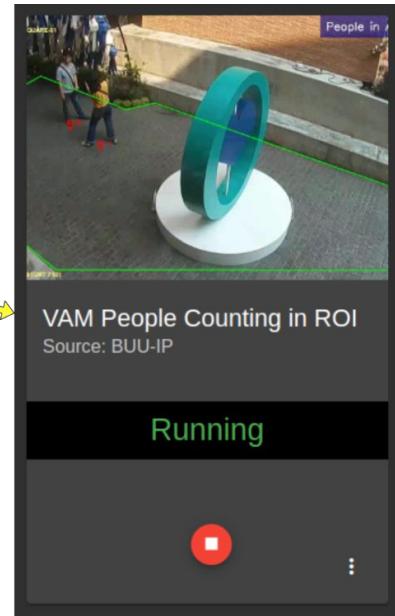
Stopped

VAM People Counting in-out
Source: BUU-IP

VAM Studio #1
Source: BUU-IP

Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd.

Version 3.7.0 (for x64 arch.)





3. คลิกที่การ์ด “VAM People Counting in ROI” เพื่อดูการประมวลผลและวิเคราะห์ข้อมูล

VAMStack Analytics Raw Output ANALYTICS OUTPUT LIVE FIGURE

Processed Images

1970-01-18 Sun 08:12:48

Total: 2

ID: 31 ID: 32

24x2 Care Center

Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd.

Version 3.7.0 (for x64 arch.)

The screenshot shows the VAMStack interface for "Analytics Raw Output". On the left, there's a sidebar with navigation links: Store, Assignment, Report, System Configuration, Sources Management, Users Management, and Resources. The main area has two sections: "ANALYTICS OUTPUT" and "LIVE FIGURE". The "ANALYTICS OUTPUT" section displays a grid of processed images from June 17, 2022, at various times. Below the images is a JSON log entry:

```
{  
  "ts": "2022-06-17 09:20:05",  
  "analyticId": 4,  
  "sourceId": 251,  
  "analyticsType": "detection",  
  "probability": 0.6785496473312378,  
  "image": {  
    "cropped": "/assets/vamstack/images/analytics/results/4-14-251-165543240552800000.jpeg"  
  },  
  "recognition": null,  
  "data": {}  
}
```

The "LIVE FIGURE" section shows a live video feed of an office environment. Two people are detected and tracked with blue bounding boxes. The text "ID: 31 ID: 32" is displayed above the boxes, and "Total: 2" is shown below. The timestamp in the video feed is 1970-01-18 Sun 08:12:48. The video feed is labeled "24x2 Care Center".

People Counting

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

- File Bar:** File, Edit, Selection, View, Go, Debug, Terminal, Help.
- Explorer:** Shows the project structure under "OPEN EDITORS" and "APP".
- Code Editor:** The file "example_05.py" is open, containing Python code for interacting with an AI service for people counting. The code imports requests, pprint, and json, defines parameters for the AI service (HOST_IP, PORT, endpoint), creates a request form, sends a GET request, and prints the JSON response.
- Terminal:** A terminal window titled "1: bash" shows the command "cd examples_code/ai-as-a-service" followed by "python3 example_05.py".
- Output:** The terminal output shows the JSON response from the AI service, which includes data like analyticsId, analyticsResult, objsInfo, assignmentId, fullImg, imgDim, sourceId, sourceName, origin_name, and time.
- Bottom Status:** Shows file status (0 ▲ 0), Ln 1, Col 1, Spaces: 2, UTF-8, LF, Python.



1. คลิก “Assignment” ที่แถบเมนูด้านซ้ายของหน้า VAM Platform
2. ตั้งค่าพื้นที่ที่สนใจออกเป็น 2 ส่วนคือ พื้นที่เข้า (IN) และพื้นที่ข้อออก (OUT)

VAMStack

Analytics Assignment

VAM Face Mask Detection
Source: BUU-IP

VAM Human Detection and Analytics
Source: BUU-IP

Stopped Stopped

VAM People Counting in-out
Source: BUU-IP

VAM Studio #1
Source: BUU-IP

Stopped Stopped

Configuration

Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd.

VAMStack

Configuration

1970-01-18 Sun 08:18:37

Configuration

Line Notification API

Your Token Line Notification

Save

Polygon Set ROI name Click ROI's name OUT OK Delete

Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd.

Version 3.7.0 (for x64 arch.)



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

The screenshot displays the VAM Stack software interface. On the left, a camera feed titled "VAM People Counting in-out" shows several people walking through a doorway. The feed is labeled "Source: BUU-IP". Below the feed, the status is shown as "Running" in green text. A red circular button with a white square icon is located at the bottom left. A yellow arrow points from this button towards the main interface.

The main interface consists of a grid of cards. The first card, highlighted with a yellow border, is titled "VAM People Counting in-out" and "Source: BUU-IP". It shows the same scene as the camera feed but with a different overlay. The top right corner of the card displays "IN : 17" and "OUT : 16". The status below the card is "Stopped" in red text. To the right of this card is another card titled "VAM Studio #1" and "Source: BUU-IP", also showing a "Stopped" status.

Below the main grid, there is a footer bar with the text "Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd." and "Version 3.7.0 (for x64 arch.)".



3. คลิกที่การ์ด “VAM People Counting in-out” เพื่อดูการประเมินผลและวิเคราะห์ข้อมูล

VAMStack Analytics Raw Output ANALYTICS OUTPUT LIVE FIGURE

Processed Images

2022-06-17 09:26:37

2022-06-17 09:26:30

1970-01-18 Sun 08:19:25

People in: 2
People Out: 0

IN
OUT

24x7 Care Center

Copyright © 2021 VAMStack - Powered By VAMStack Co.,Ltd.

Version 3.7.0 (for x64 arch.)

```
{ "ts": "2022-06-17 09:26:37", "analyticsId": 6, "sourceId": 251, "analyticsType": "detection", "probability": 0.7067449688911438, "image": { "cropped": "/assets/vamstack/images/analytics/results/6-16-251-165543279797700000.jpeg" }, "recognition": null, "data": {} }
```

People Counting in-out

The screenshot shows the VAM Studio IDE interface. On the left is the Explorer sidebar with a tree view of project files. The main area displays a code editor for `example_06.py`. The terminal at the bottom shows the execution of the script.

Code Editor Content:

```
example_06.py x
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 = "/people-counting-in-out"
10
11 # parameter for requests people-counting in-out ai services
12 request_form = {
13     "machine_ip": "10.1.1.212",
14     "limit_data": 1
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.pprint(response.json())
23
```

Terminal Output:

```
cd examples_code/ai-as-a-service
python3 example_06.py
```

Terminal Log:

```
1: bash
analyticResult': {'cntIn': 0,
                  'cntOut': 1,
                  'objsInfo': {'croppedImg': '/assets/vamstack/images/analytics/results/6-6-250-165381835162800000.jpeg',
                               'label': 'person',
                               'probability': 0.7262669205665588,
                               'roiName': 'OUT',
                               'trackId': 74,
                               'x1': 1094,
                               'x2': 1242,
                               'y1': 388,
                               'y2': 815}},
'assignmentId': 6,
'fullImg': '/assets/vamstack/images/analytics/results/6-6-165381835162800000-fullframe.jpg',
'imgDim': {'height': 1080, 'width': 1920},
'sourceId': 250,
'sourceName': 'BDH-VAM',
'origin_name': 'VAMSTACK-ONPREMISED',
'time': '2022-05-29T09:56:16.686696+00:00'}]
root@d933f59ec2db:/usr/src/app/examples_code/ai-as-a-service#
```

Bottom Status Bar:

Ln 1, Col 1 Spaces: 2 UTF-8 LF Python