



CE65-12



IPAU TSoNS

ร้านค้าสำหรับแอพพลิเคชันการประมวลผลภาพซึ่งจัดการงานบนระบบประมวลผลแบบกลุ่ม
Marketplace for Image Processing Application using Task Management on Cluster Computing System

สมาชิก
นาย พศิน จันทร์กัน 63015121
นาย สุรี สาระพันธ์ 63015190

อาจารย์ที่ปรึกษา
รศ. ดร. อรฉัตร จิตต์ไสภัคตร

หัวข้อการนำเสนอ



01.
ที่มา & ปัญหา

02.
Feature

03.
เป้าหมายหลักและขอบเขต
ของโครงการ

04.
Design &
Development

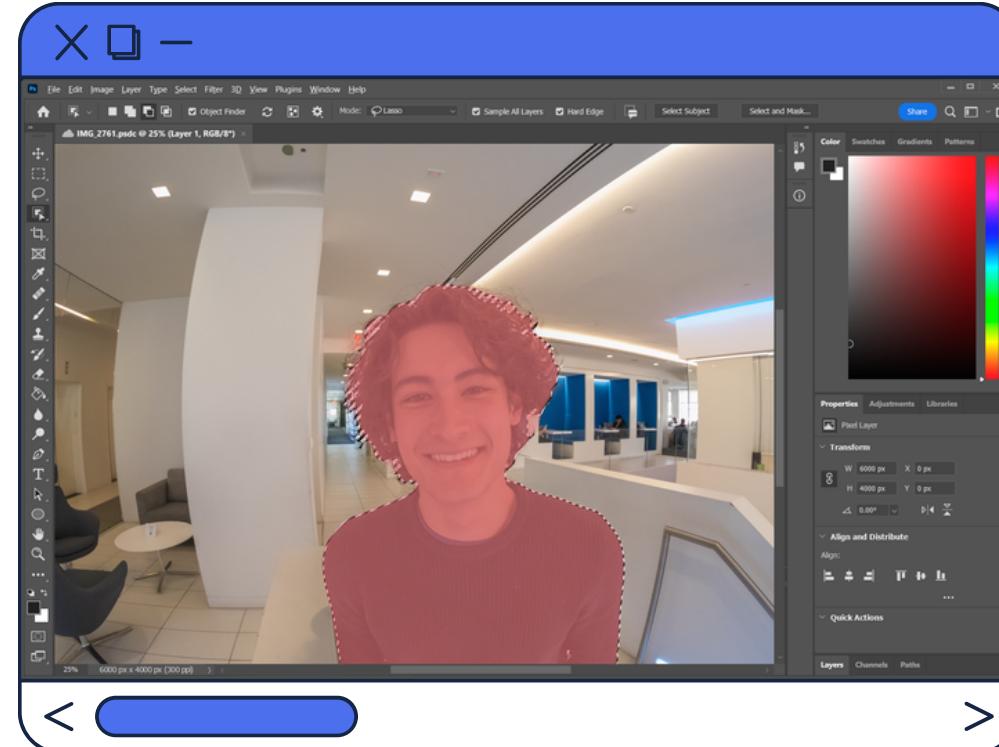
05.
ปัญหาที่เกิดขึ้น
แนวทางแก้ไข

06.
สรุป

ที่มา & ปัจจุบัน



Image Processing



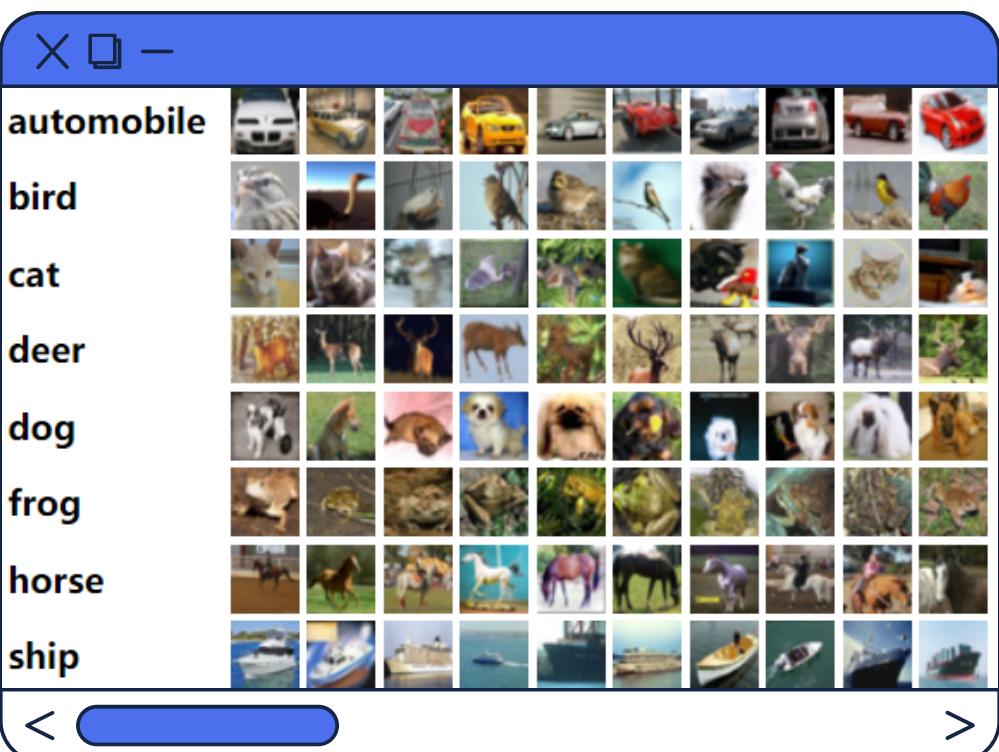
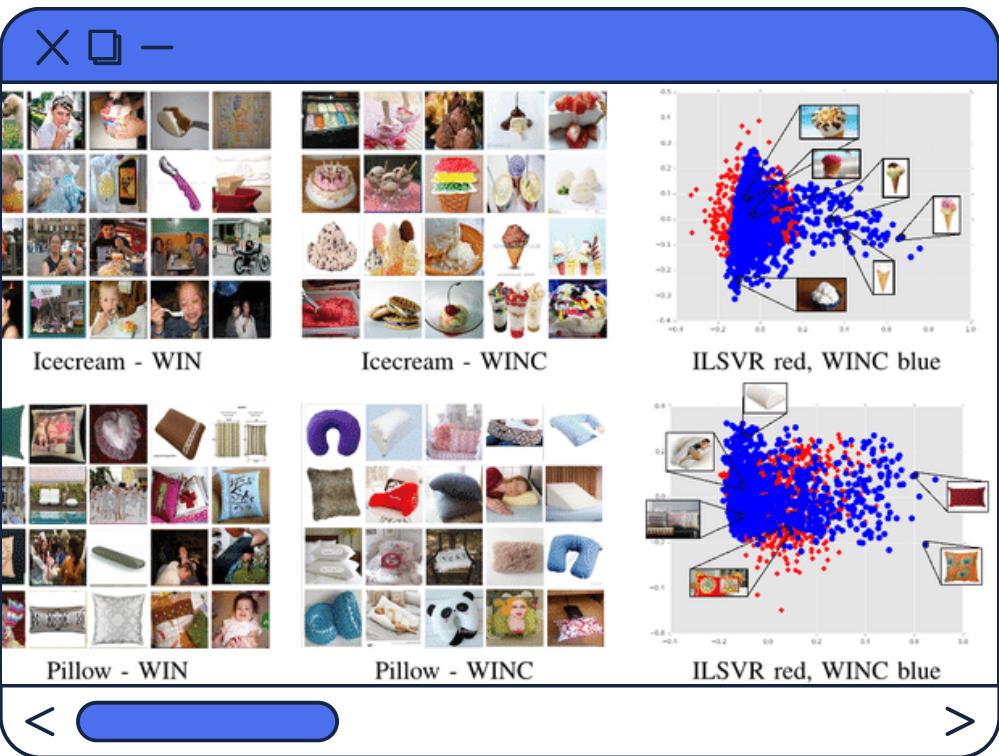
ชีวิตประจำวัน

แต่งรูปภาพต่างๆ, ถ่ายรูปหน้าชัด
หลังเบลอ, แสกน QR code ฯลฯ

อุตสาหกรรม

แยกแยะวัตถุ, ตรวจจับขนาด สี ของ
วัตถุ, Monitoring การเปลี่ยนแปลง
ของสินค้า ฯลฯ

Weight Model



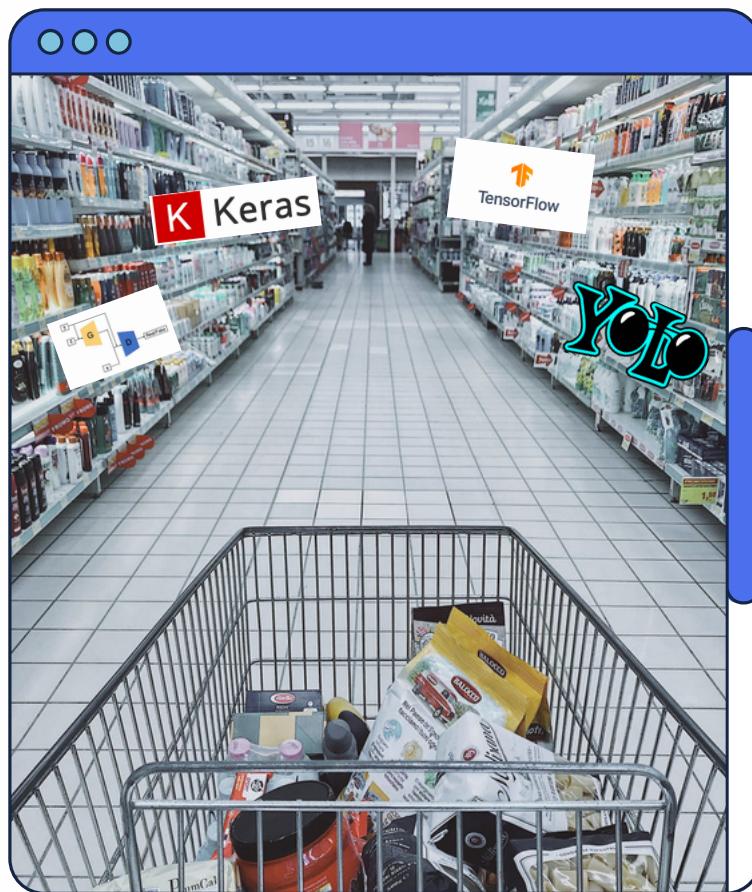
Model

Deep Learning Model ประกอบด้วย การประมวลผลเชิงคณิตศาสตร์ที่ซับซ้อน ใช้สำหรับแก้ปัญหาต่างๆ ด้วยการวิเคราะห์ข้อมูลภาพ เช่น การแยกวัตถุที่มีความซับซ้อนสูง

Weight

น้ำหนักภายในโมเดล เป็นส่วนประกอบหนึ่งของการคณิตศาสตร์ในโมเดล น้ำหนักของโมเดลที่ได้จากการสอนโมเดล จะทำให้โมเดล มีความสามารถในการทำงานเฉพาะด้านที่ต้องการ การเปลี่ยนน้ำหนัก เป็นการควบคุมให้โมเดลทำงาน เปลี่ยนไปตามที่ต้องการ

ที่มา & ปัญหา



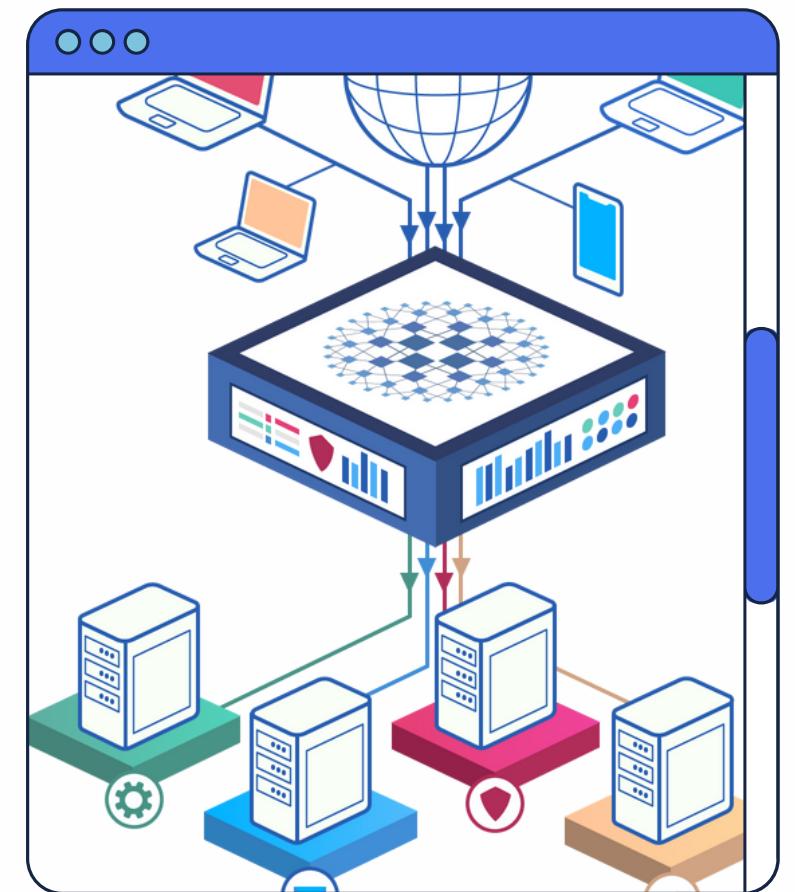
1

ยังไม่มีตลาดซื้อขาย
Weight Model ในปัจจุบัน



2

จำเป็นต้อง R&D
Image Processing Application
เพื่อเป็นบันไดในการพัฒนาส่วนอื่น



3

จำเป็นต้องมี
Task Management

Feature

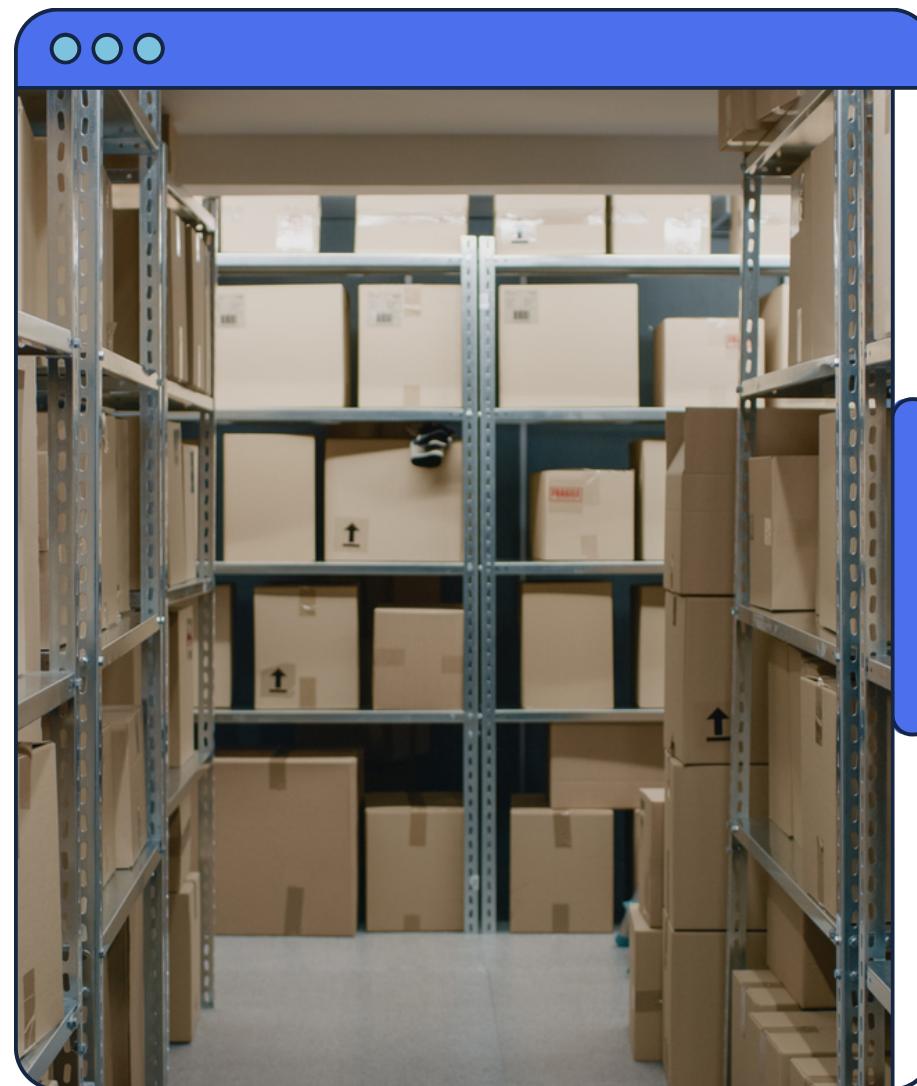
Feature ของแต่ละส่วน



Marketplace

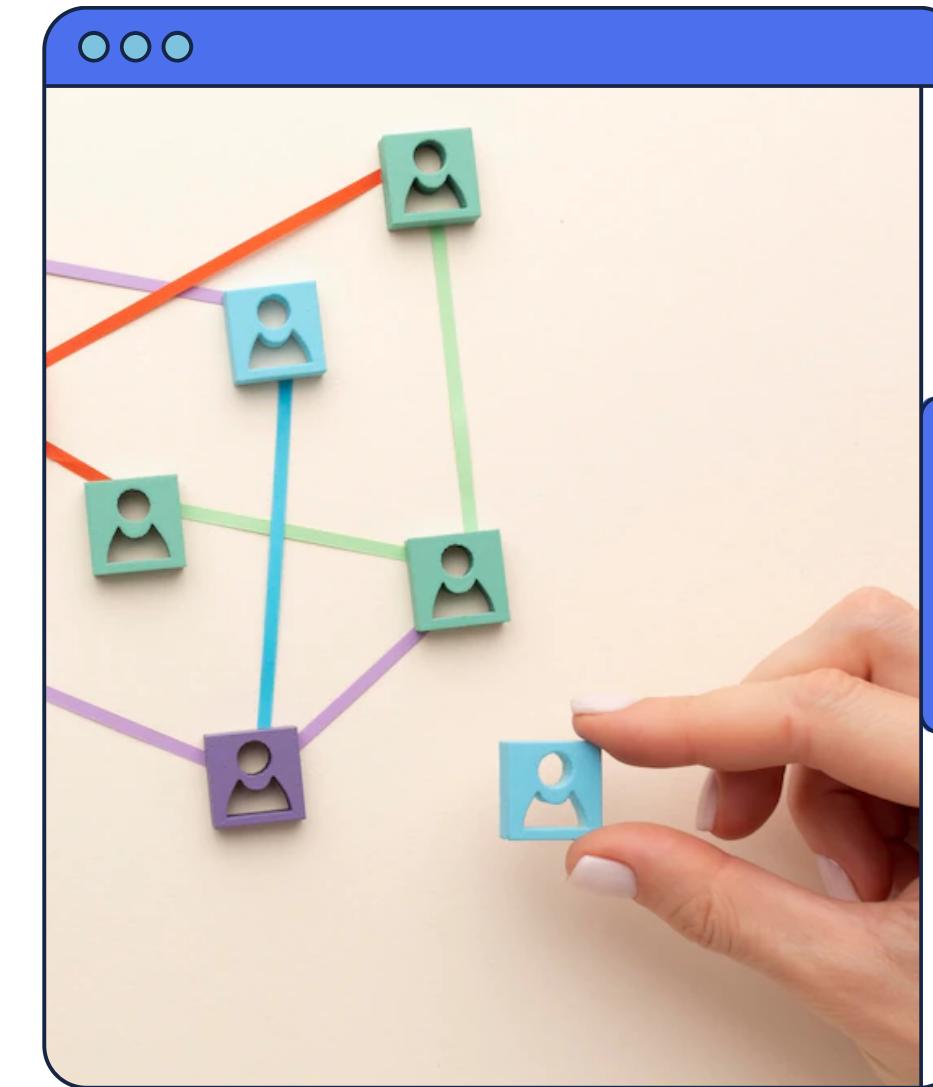


จัดทำพื้นที่แสดงผลงาน
Preview ให้ทดลองใช้
เพื่อให้สามารถดูผลลัพธ์ของการ
ประมวลผลก่อนที่จะดำเนินการ
ประมวลผล



ให้บริการพื้นที่จัดเก็บผลงาน
Weight Model กับ Train มา
เพื่อให้สามารถจัดเก็บผลงานของ
ตนเอง และ สะดวกมากยิ่งขึ้นใน
การใช้งาน

Image Processing Application

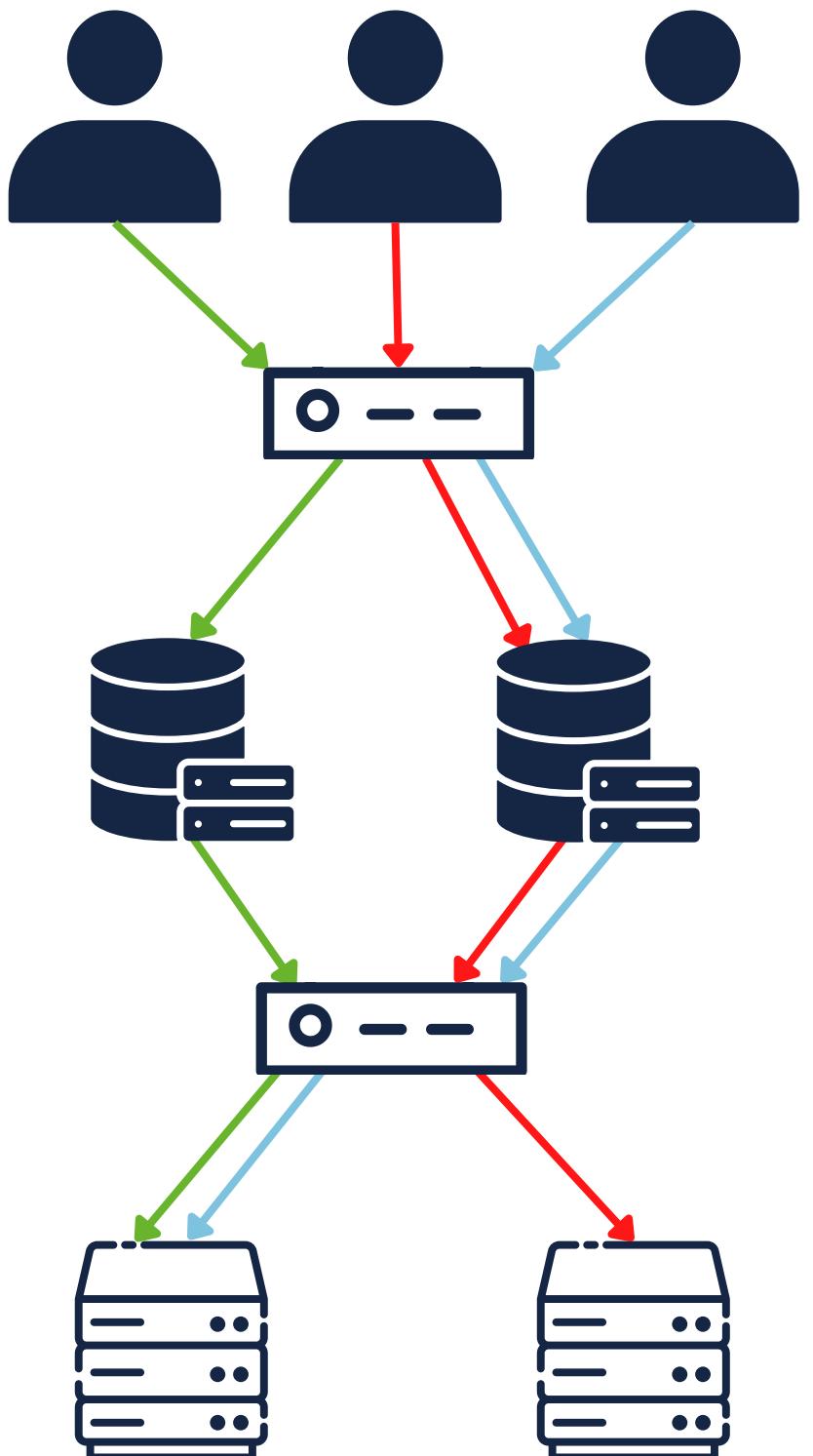


ให้บริการประมวลผลภาพ

- ประมวลผลภาพจาก App (Basic & Advance)
- ประมวลผลภาพจาก Weight ที่ผู้ใช้ได้ทำการซื้อขาย

ให้บริการเตรียม Model ที่จะใช้กับ Weight ไว้ให้ใช้งาน เพื่อลดเวลาการเตรียมการลงไปเพื่อให้กระบวนการทำงานสัดส่วนเร็วขึ้น

Task management



Load balance Web Application Server

เพื่อลดการเกิดภาวะคอขวดที่
สาเหตุมาจากการที่ผู้ใช้บันทึ้นเข้าใช้
Web application จำนวนมาก

Load Balance งานประมวลผล

เพื่อให้รองรับการประมวลผลภาระจำนวนมาก
มากได้ โดยที่ใช้ประสิทธิภาพของ
ทรัพยากรได้คุ้มค่า

งาน & นวัตกรรมที่ใกล้เคียงกัน



Google Colab ให้บริการใช้งาน Notebook Code บนระบบ Cloud ในการเขียน Code และ Execute บน Browser

- ต้องเตรียม Code และ Model เองตั้งแต่ต้นในการใช้งาน
- ทรัพยากรในการประมวลผลค่อนข้างต่ำ



AWS Marketplace ให้บริการใช้งาน Model บนระบบ Cloud ในงาน Image Processing และ ML

- มีค่าใช้จ่ายที่สูง คิดเป็นเวลาในการใช้งาน
- ต้องจัดเตรียม Model เพิ่มเติมเองเพื่อใช้งานทำให้มีเวลาไปเป็นผลกระทบกับราคาให้สูงขึ้น

นอกจากนี้ก็มี Cloud service อื่นๆ ที่ให้ใช้งานในการ Pre-Trained Model และ Train Model ในลักษณะคล้ายๆ กัน แต่ว่าก็ยังขาดในส่วนของที่เป็น Marketplace ไป

เป้าหมายหลักและขอบเขต ของโครงการ

Main Goal

Main Goal



Marketplace



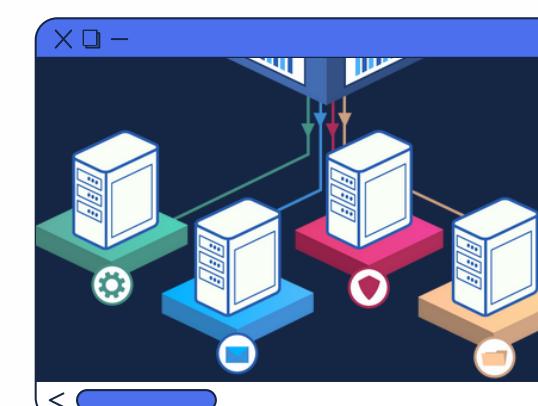
Task Management



ให้บริการ Basic & Advance
Image App



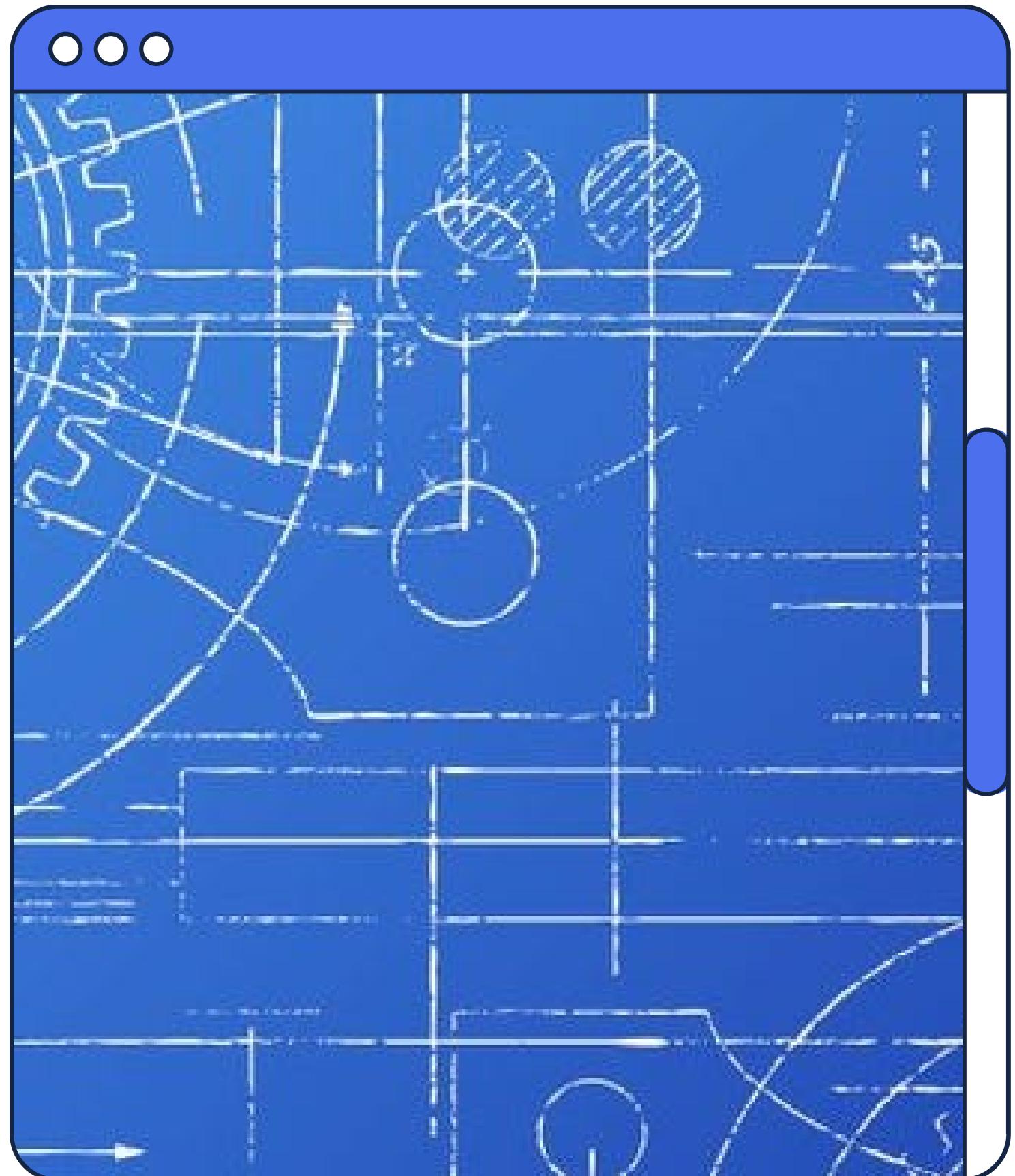
ให้บริการซื้อขาย Weight Model
ใน Marketplace



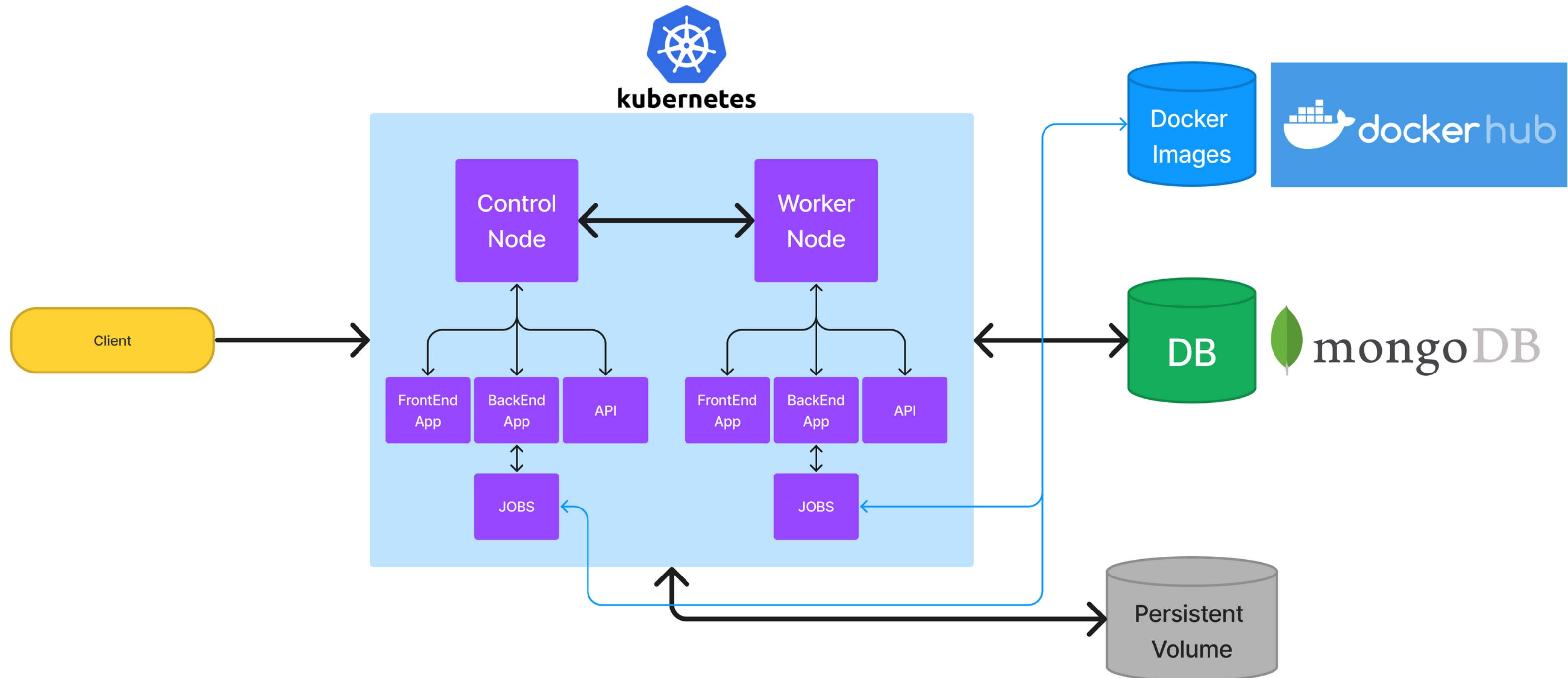
มีระบบการจัดการ
งานประมวลผลภาพ

Design & Development

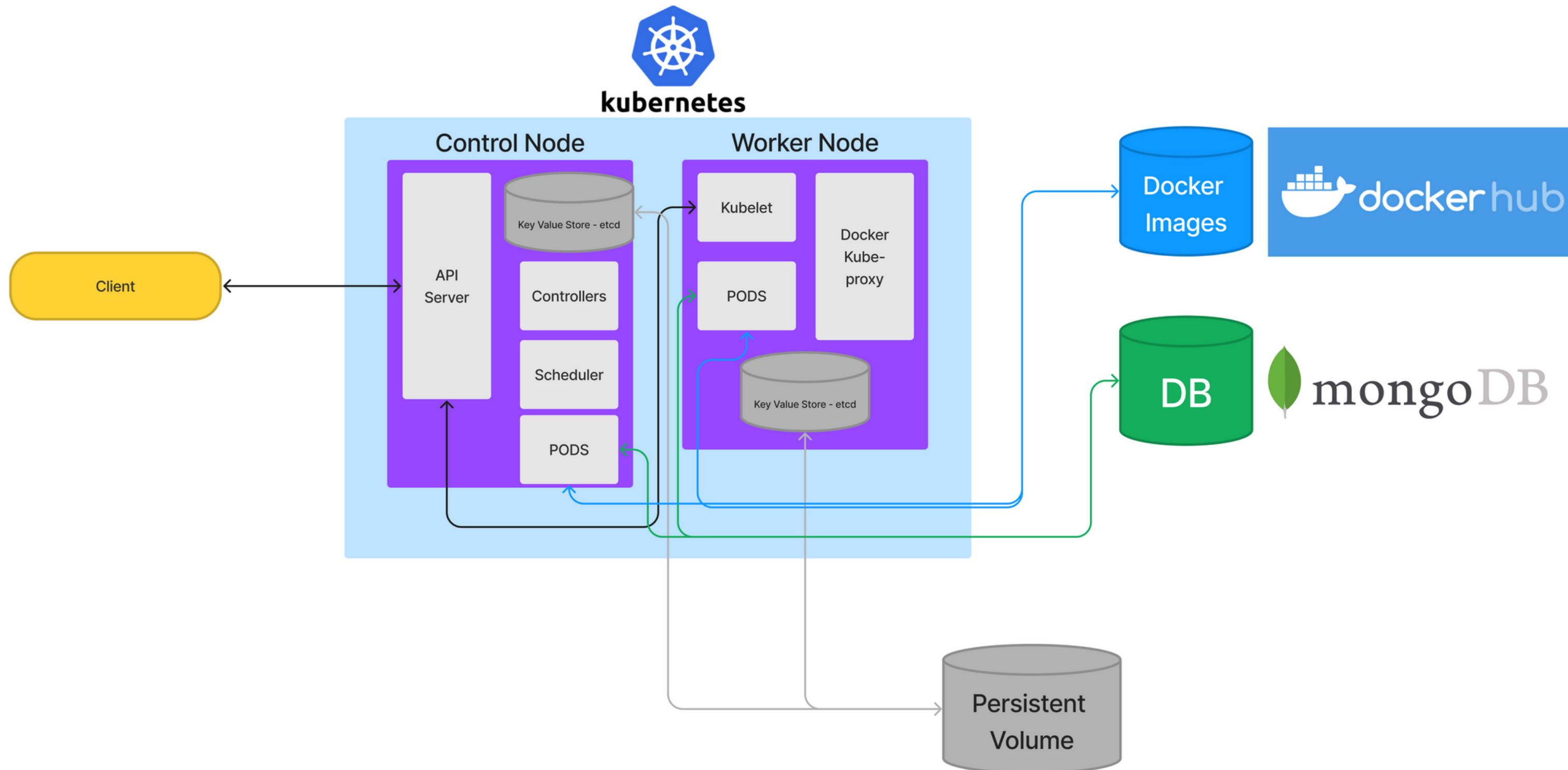
การออกแบบระบบและการพัฒนา



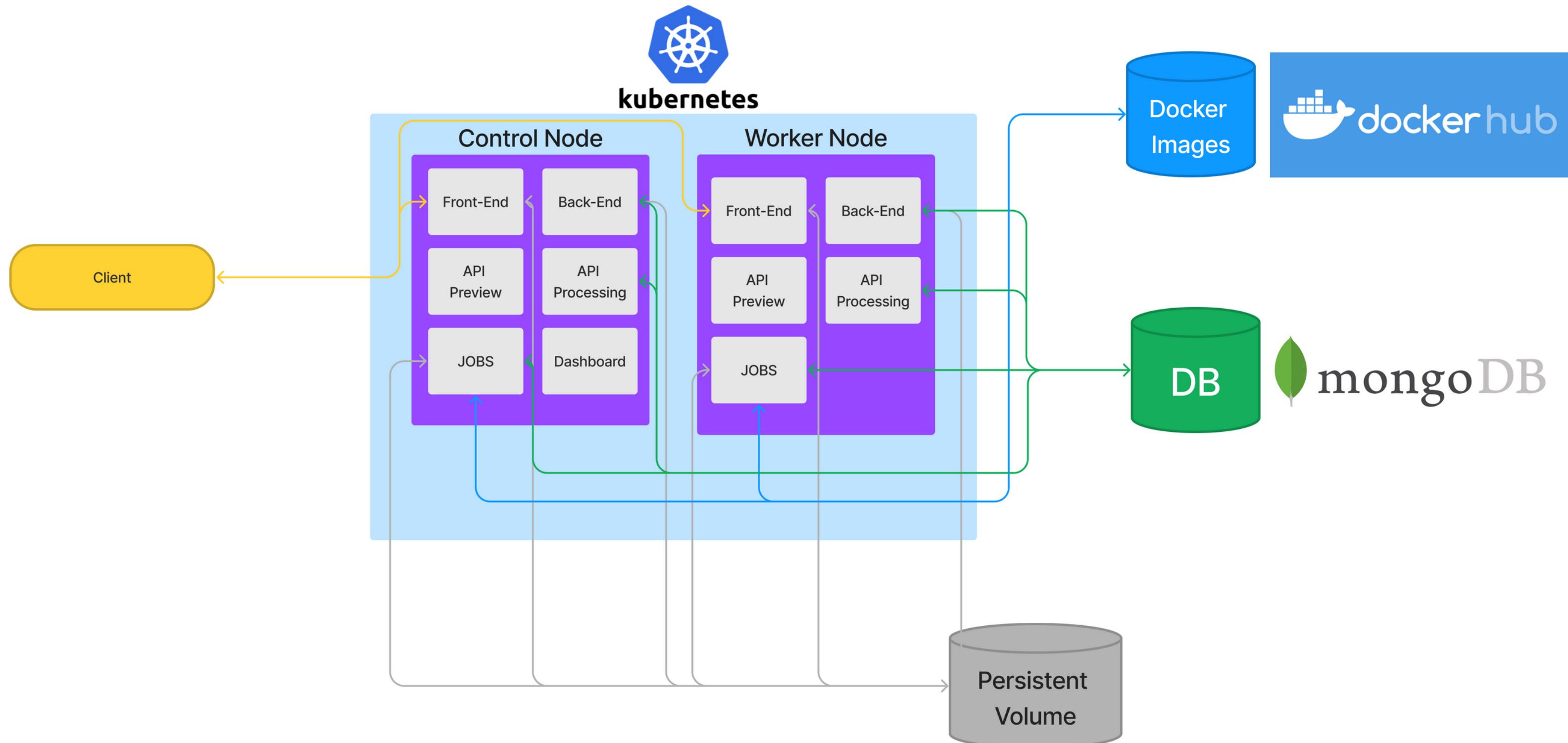
System Design Overview



System Design Node

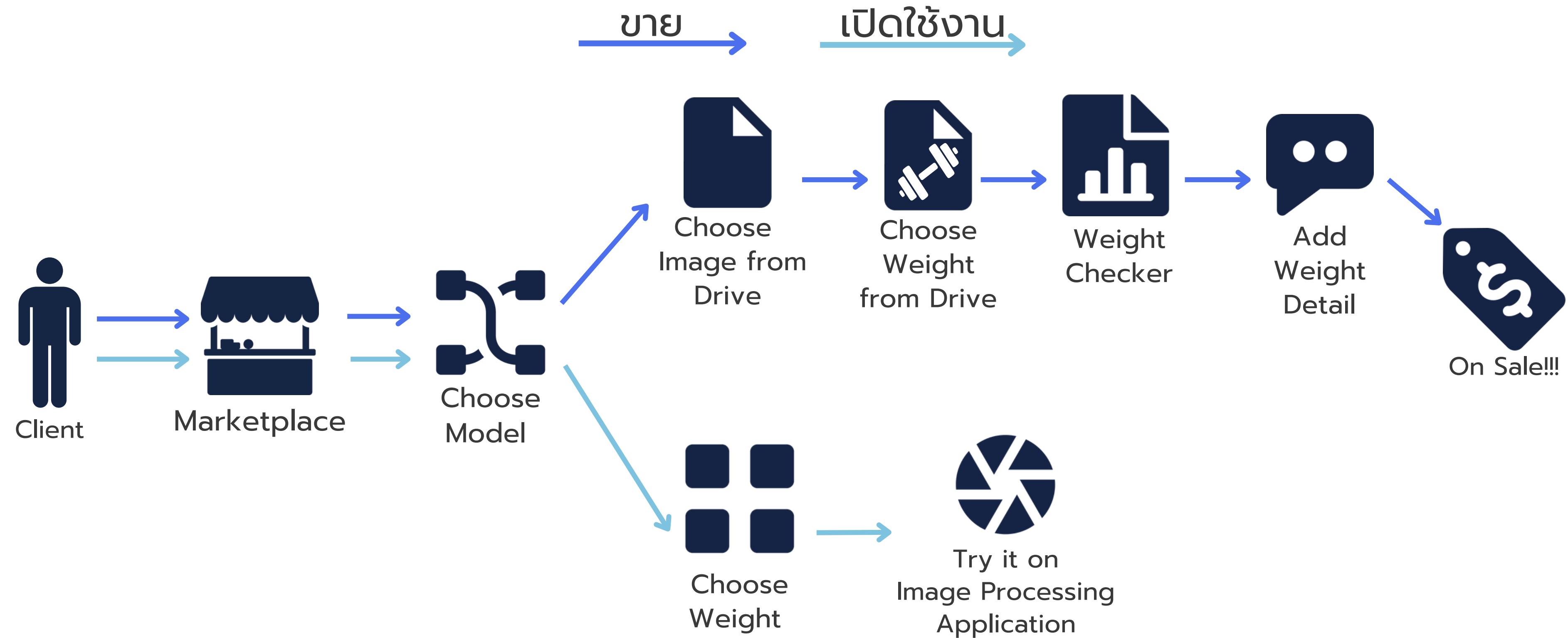


System Design PODS



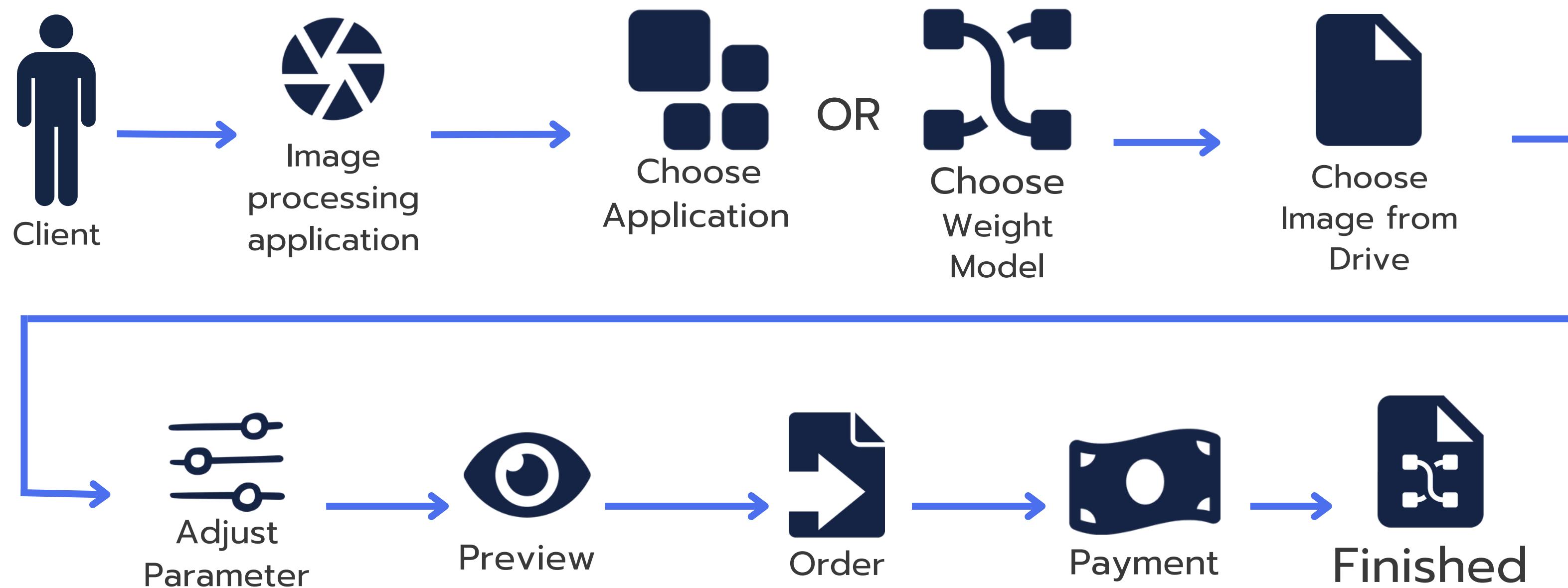
การทำงานของระบบ

ในส่วนของการ เปิดใช้ และ ขาย Weight Model



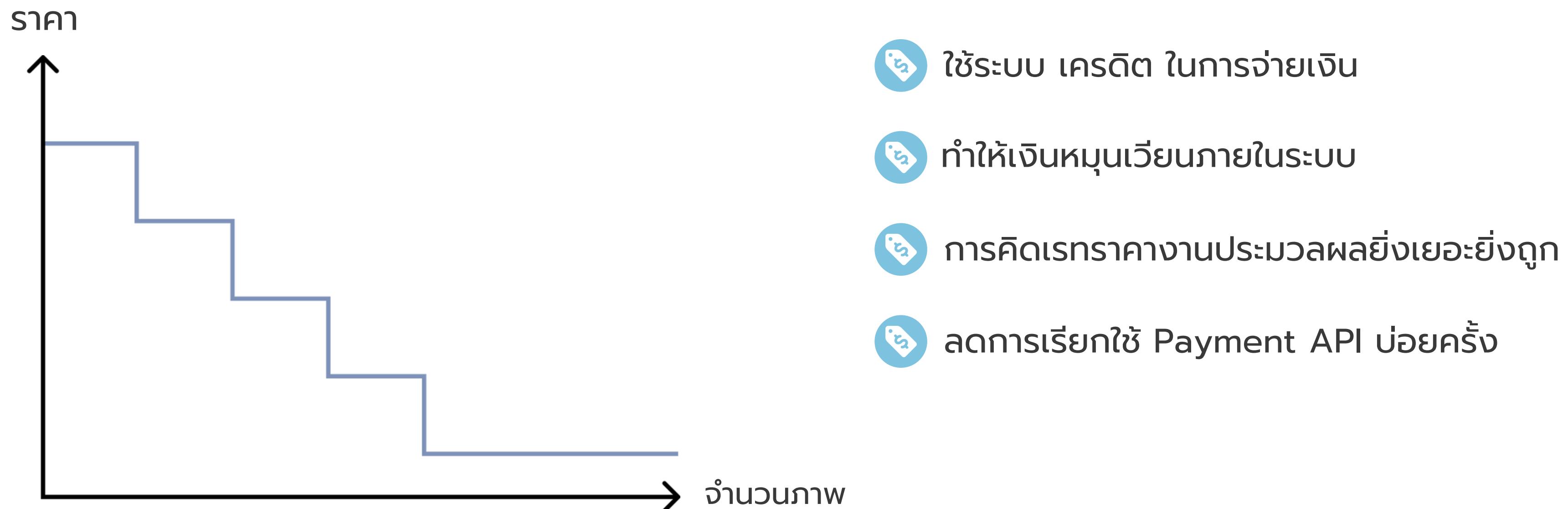
การทำงานของระบบ

ในส่วนของการใช้งาน Weight Model และ Application
uu Image Processing Application

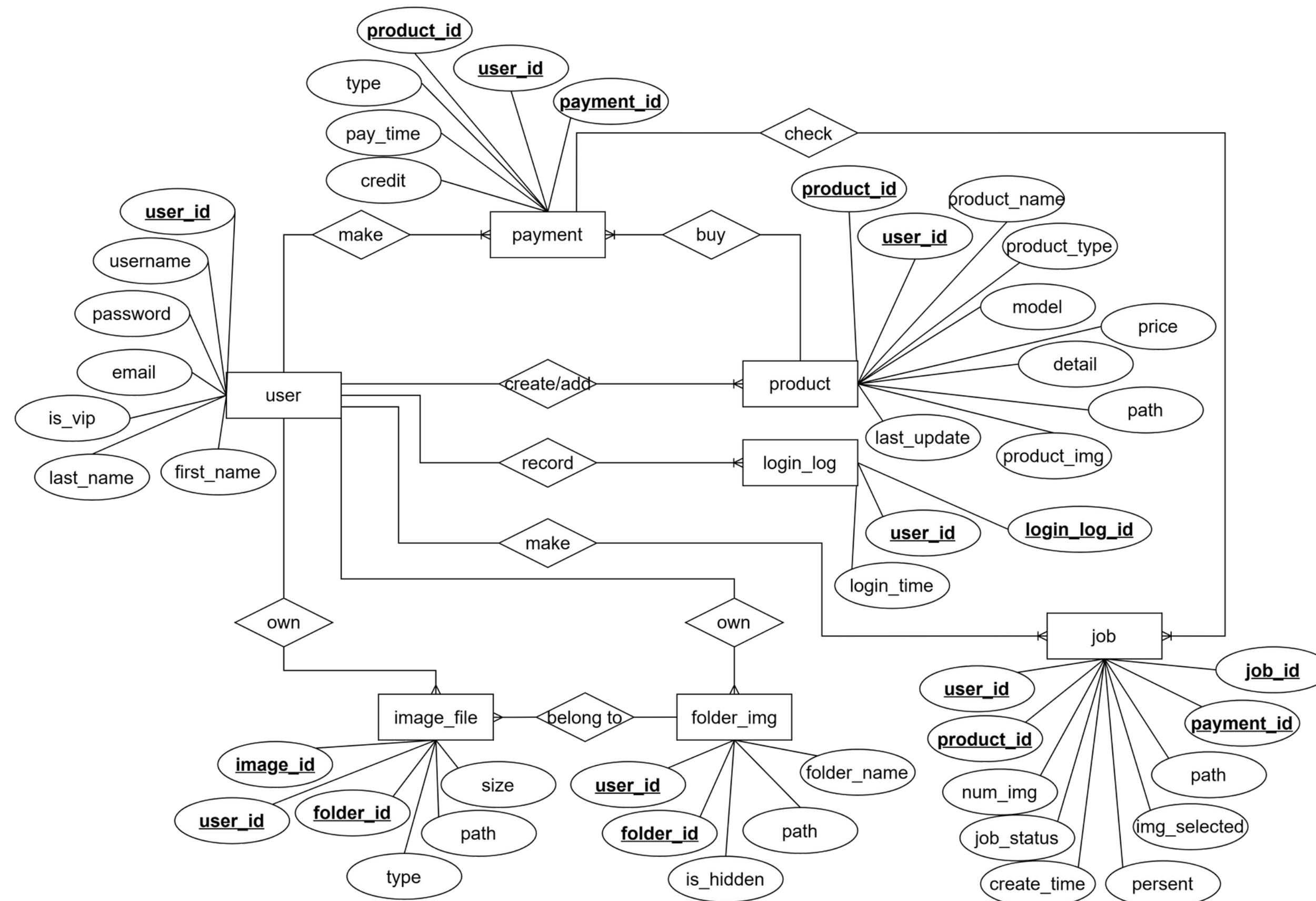


Payment

Price rate



ER diagram





Web Application

កុំ My Drive

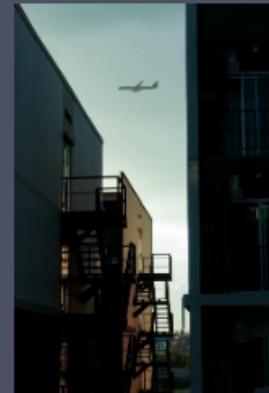


Drive

Folder name : test

Choose Files | No file chosen

Upload



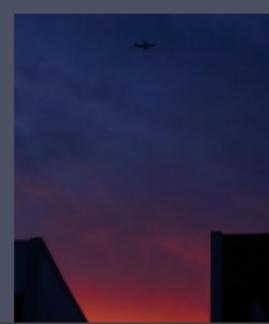
DSC00161.JPG



DSC00148.jpeg



DSC00145.JPG





Web Application

ՀԱՅ Application page (Basic & Advance App)

≡

Image processing application

BASIC PRODUCT

- Black and White
- ASCII
- PixelArt
- Mosaic

MARKETPLACE PRODUCT

Don't have product from marketplace.

Drive > test

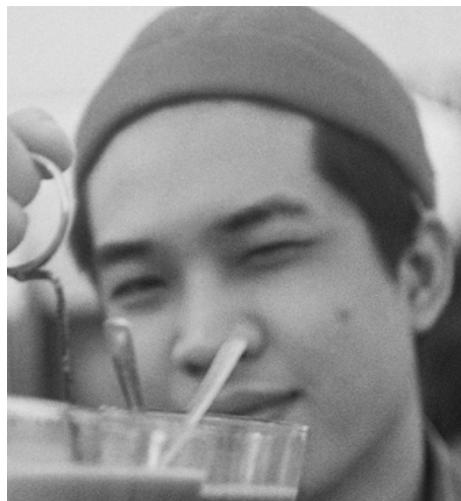
Export



Image Processing Application

ตัวอย่างการประมวลผลภาพโดยใช้ Application ที่พัฒนาขึ้นมาโดยเป็นการประมวลผลภาพที่ใช้ทรัพยากรไม่สูง เช่น CPU ก็เพียงพอต่อการประมวลผล

Black and White



ASCII

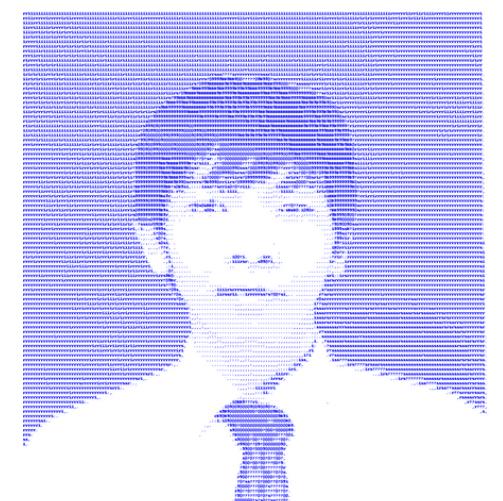
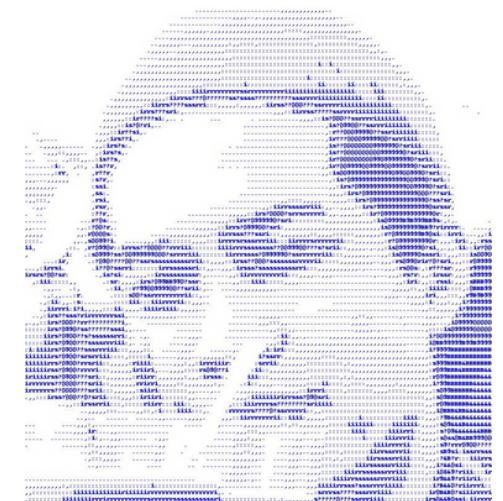




Image Processing Application

ตัวอย่างการประมวลผลภาพโดยใช้ Application ที่พัฒนาขึ้นมาโดยเป็นการประมวลผลภาพที่ใช้กรวยการสูง เช่น จำเป็นต้องใช้ GPU ช่วยในการประมวลผล

Mosaic



PixelArt



ML



DeepNN



Web Application

ក្នុង Marketplace page

☰ Market

Marketplace

តារាងថ្មី-ខាយ

sort by newest sort by oldest
mm/dd/yyyy ☰

Type : All Model : All

+ Add product to sell

Seller - Pasin Chanharathan



Yhaa tub sein

Object detection

Seller - Pasin Chanharathan



365 objects

Object detection



Web Application

ຮູ້ Application page (Weight Model)

≡

Image processing application

BASIC PRODUCT

- Black and White
- ASCII
- PixelArt
- Mosaic

MARKETPLACE PRODUCT

- 365 objects detection

Drive > test

A large main image showing a person working at a desk with multiple monitors and a computer tower.

Export

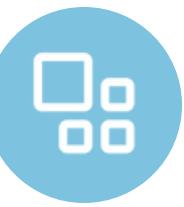


Image Processing Application

ຮະບຸ Preview ໃນສ່ວນຂອງ Application ຈັດກຳເປັນຮູບແບບ Restful API

API ipautsons Basic App 1.0.3 OAS3

[/openapi.json](#)

Basic App

default

GET / Root

POST /pixelart Convert Image To Pixelart

POST /blackwhite Convert Image To Blackwhite

POST /mosaig Convert Image To Mosaic

POST /ascii Convert Image To Ascii

Schemas

Body_convert_image_to_ascii_ascii_post >

Body_convert_image_to_blackwhite_blackwhite_post >

Body_convert_image_to_mosaig_mosaig_post >

Body_convert_image_to_pixelart_pixelart_post >

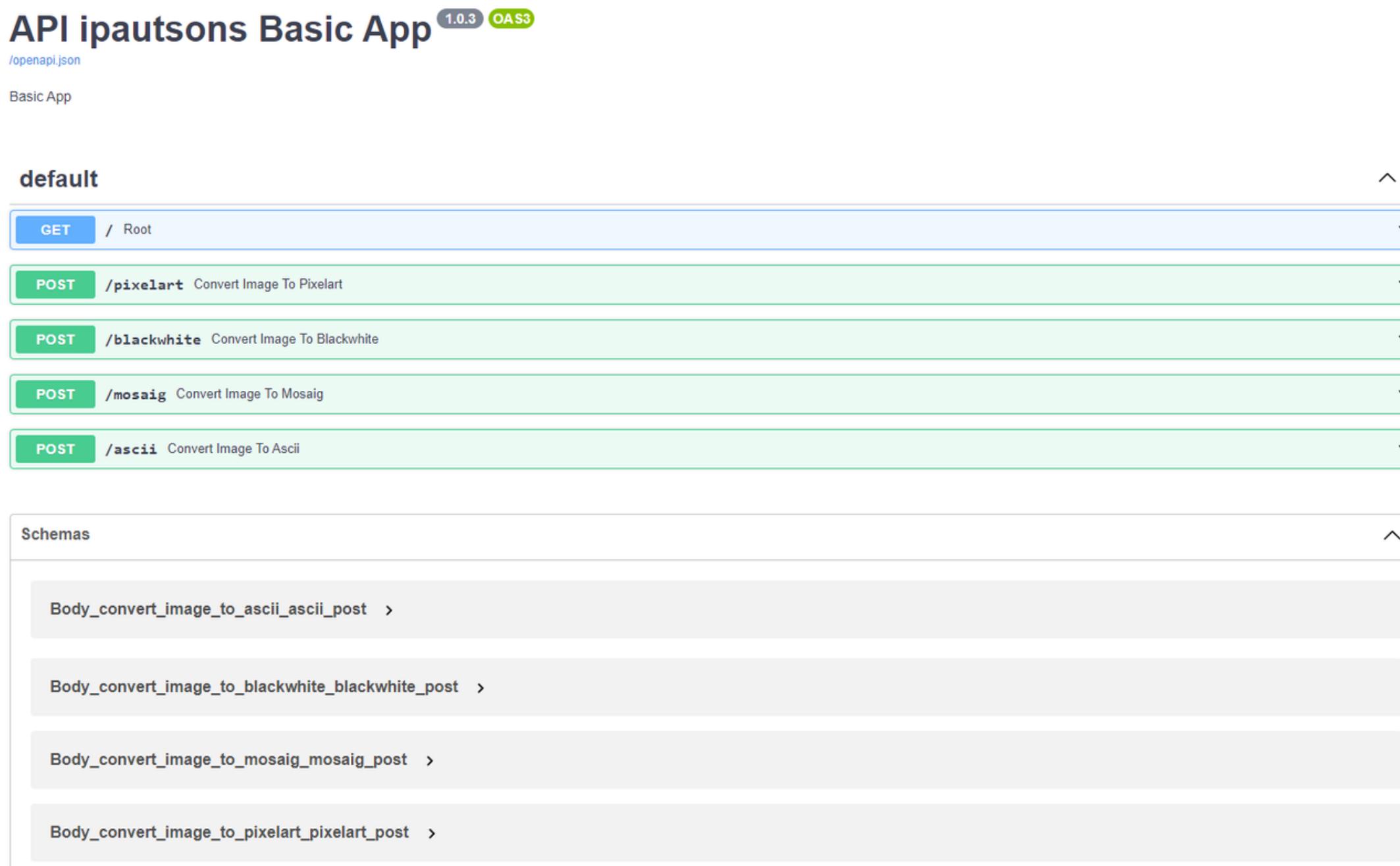
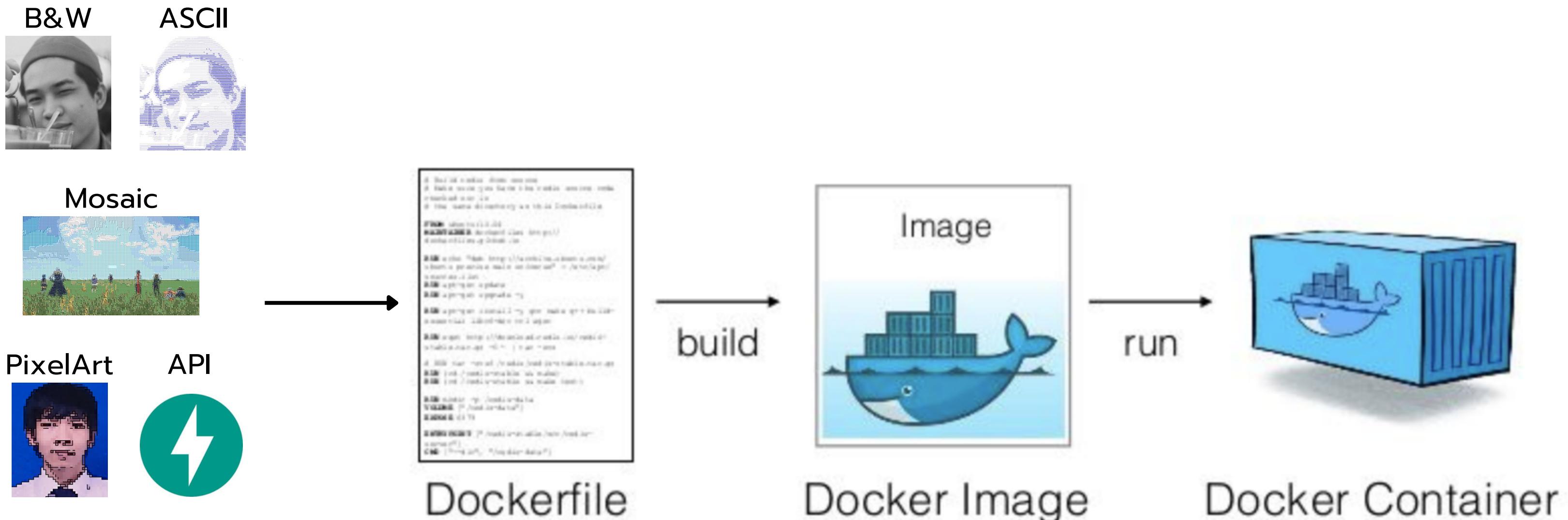




Image Processing Application

จัดเก็บ Image Processing Application ไว้ในรูปแบบ Docker Image

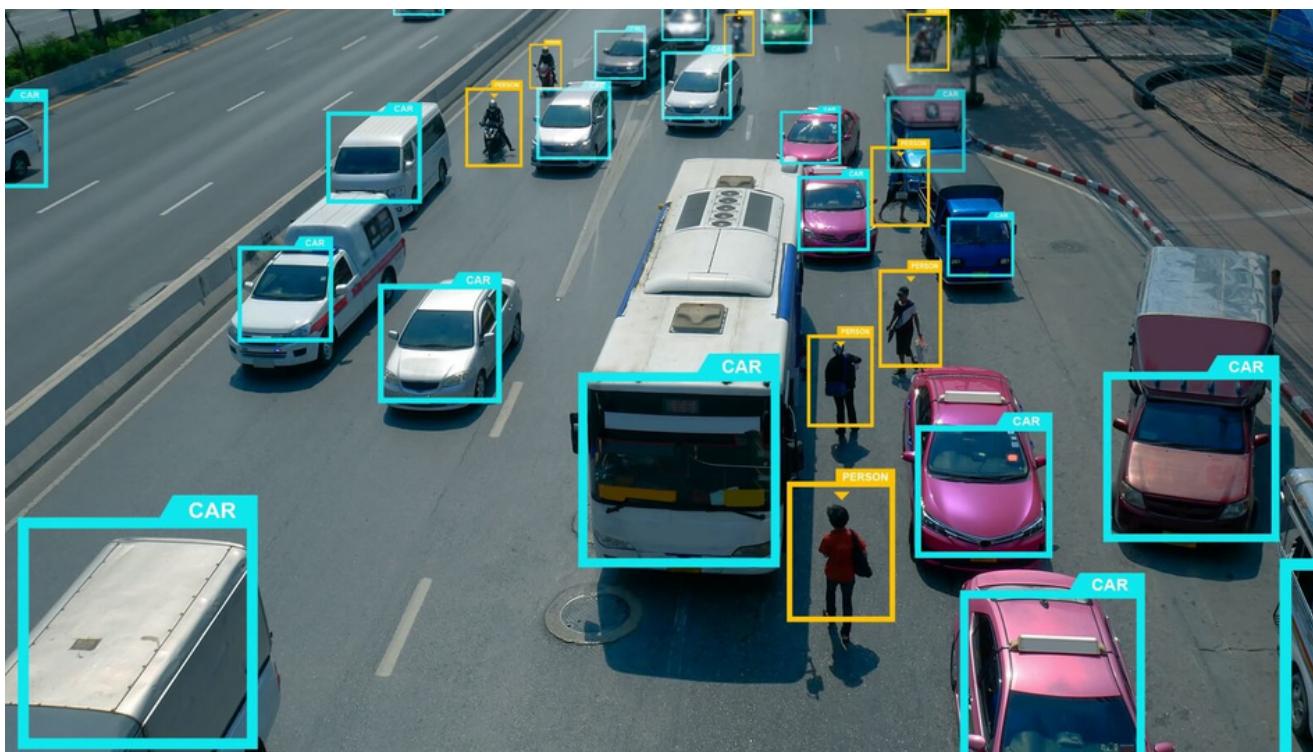




Marketplace

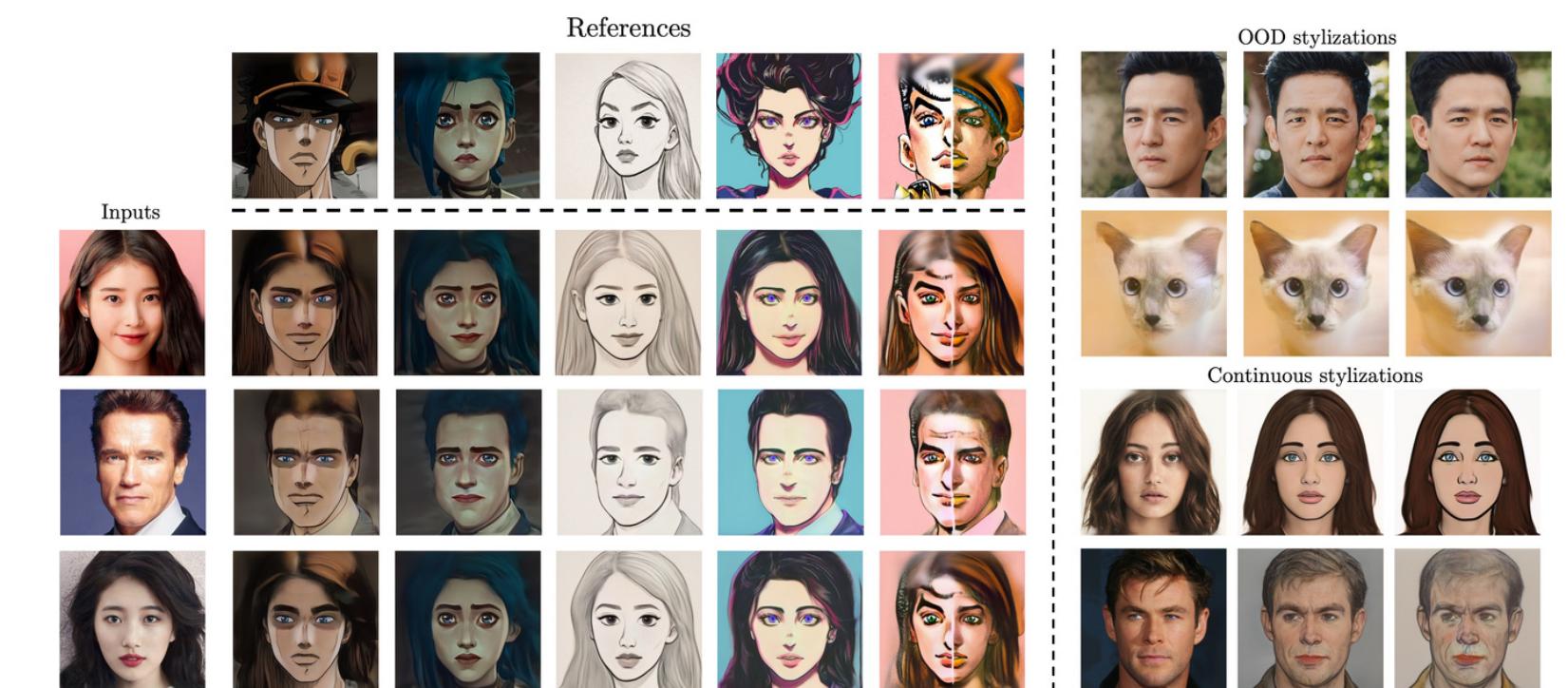
Model กี่จะมีในระบบ

YOLOv5



ML
Object Detection

GAN



DeepNN
Generator



Marketplace

ຮະບຸ Preview ໃນສ່ວນຂອງ Application ພູ Marketplace ຈັດກຳເປັນຮູບແບບ Restful API

Ipautsons API YoloModel Preview 1.0.0 OAS3

/openapi.json

Ipautsons API YoloModel Preview

default

GET / Root

POST /detect-to-json Detect Garbage Return Json Result

POST /detect-to-img Detect Garbage Return Base64 Img

POST /detect Detect Garbage

Schemas

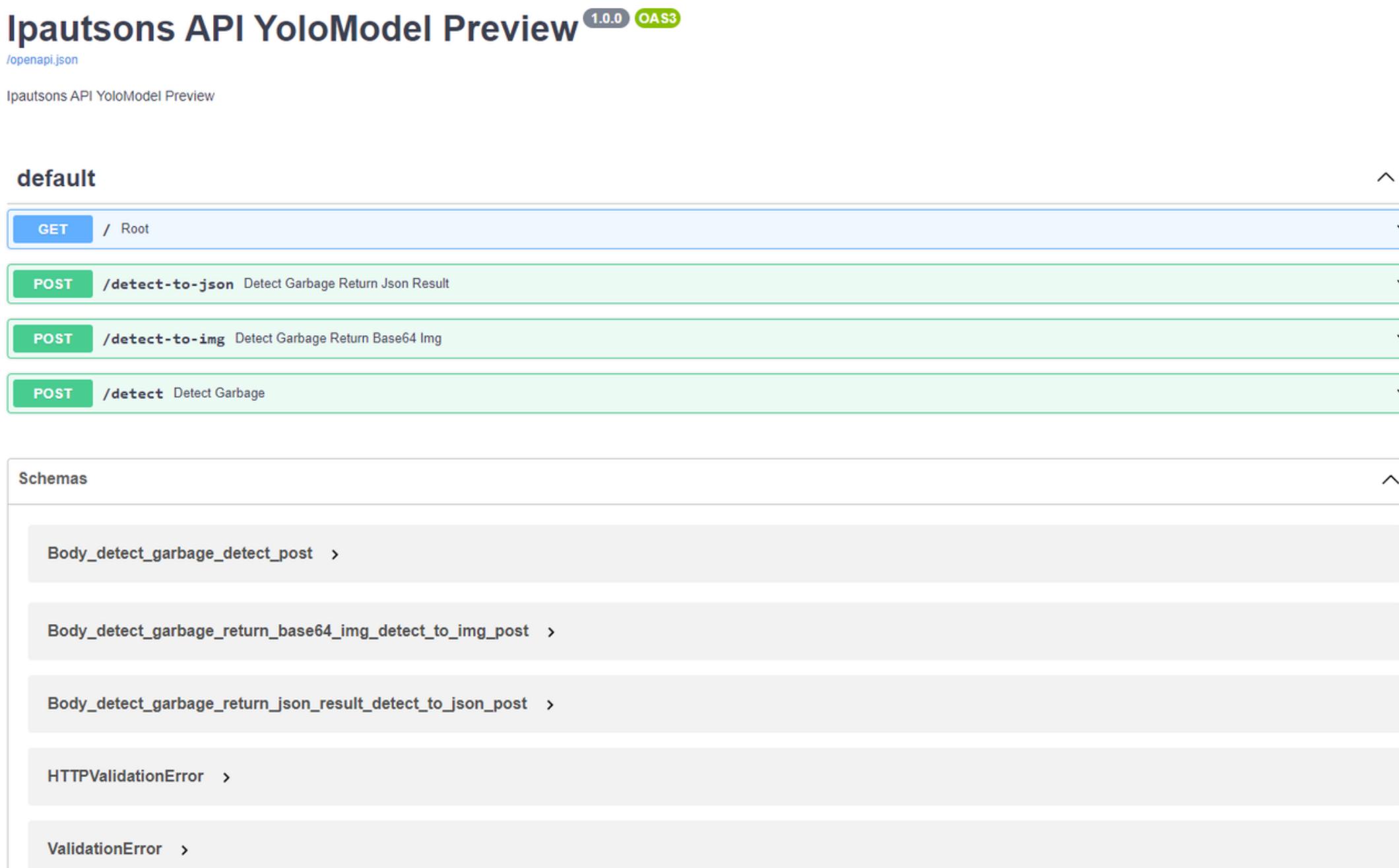
Body_detect_garbage_detect_post >

Body_detect_garbage_return_base64_img_detect_to_img_post >

Body_detect_garbage_return_json_result_detect_to_json_post >

HTTPValidationError >

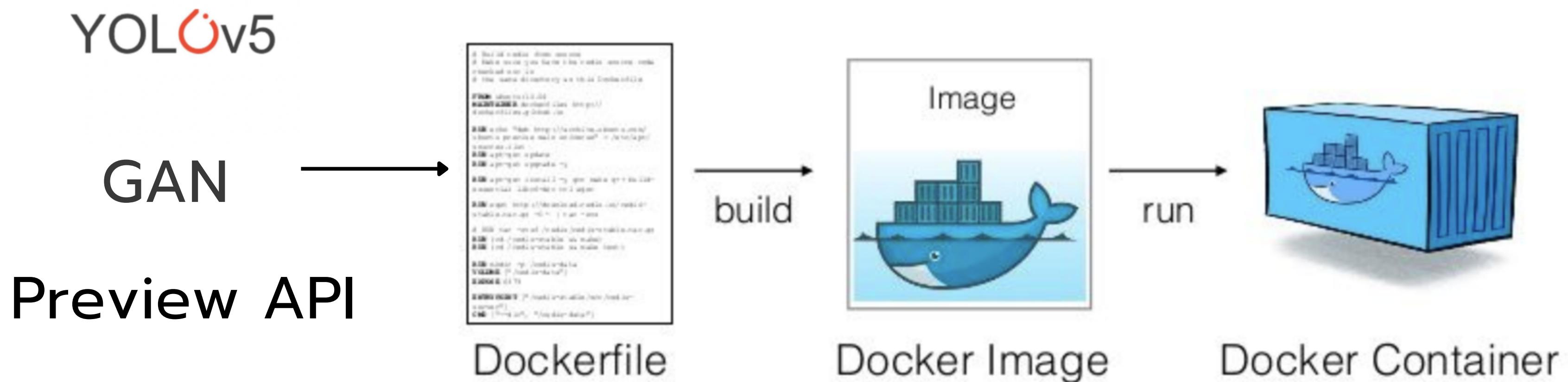
ValidationError >





Marketplace

จัดเก็บ Model ไว้ในรูปแบบ Docker Image เพื่อรองรับการทำงานกับระบบ Cluster



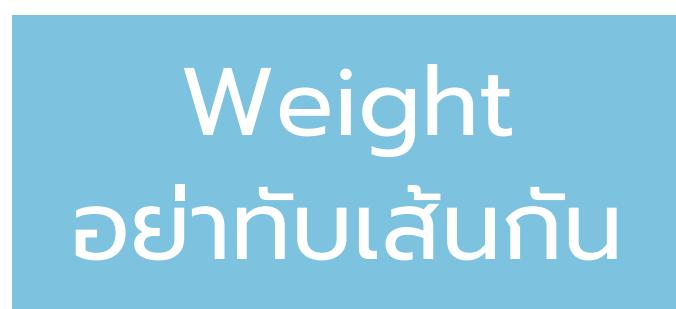


Marketplace

การทำงานร่วมกับ Application ของ Model uu Docker Image Container

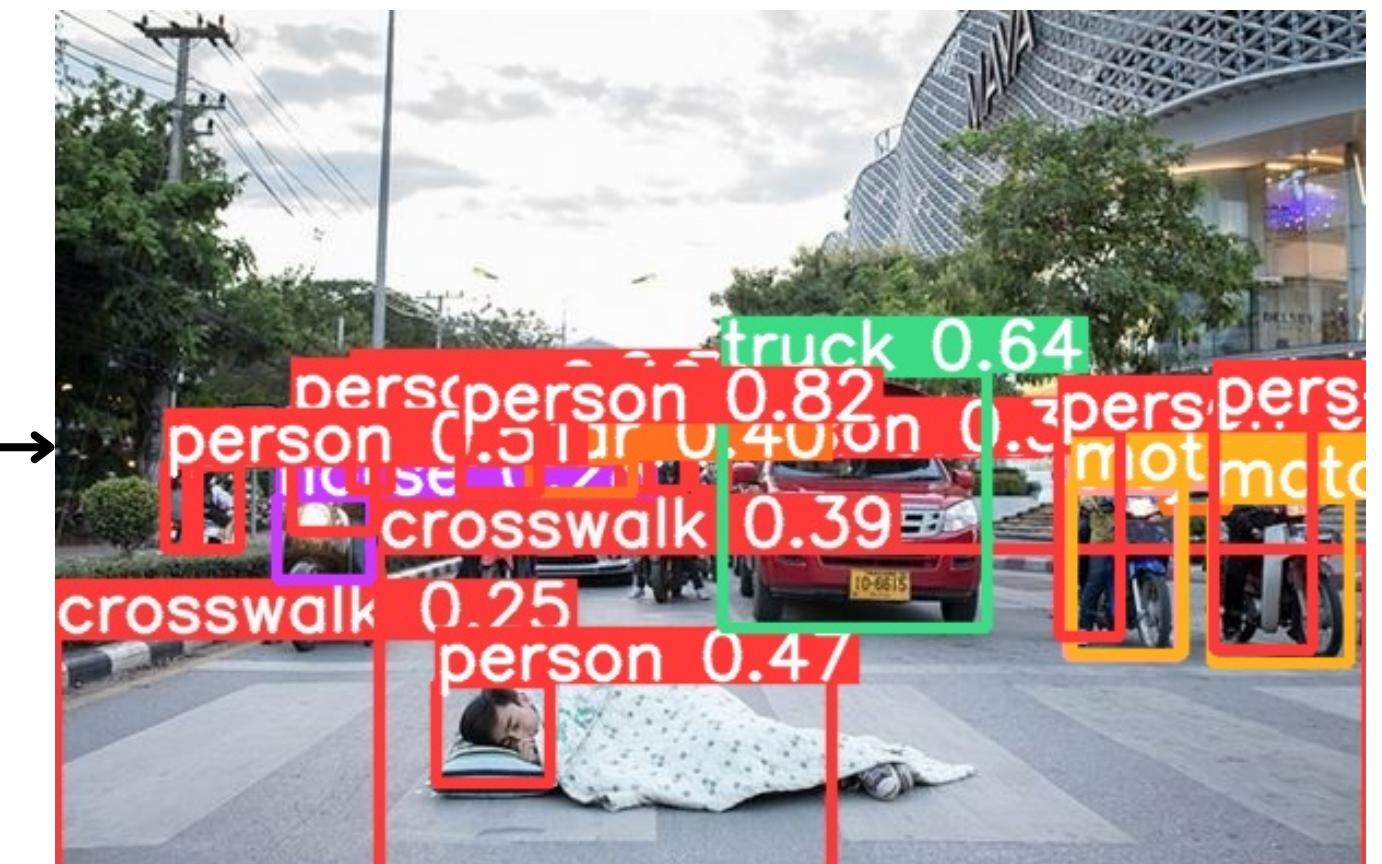


Input
รูปภาพจากผู้ใช้ที่นำเข้ามา



Weight Model
ที่มีให้ใช้บริการใน Marketplace

YOLOv5
Model
Docker Image



Output

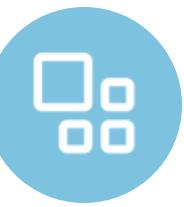


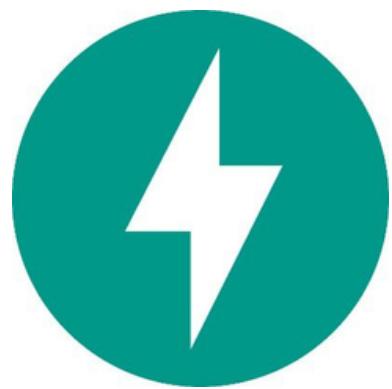
Image Processing Application

Preview API กับ JOBS



JOBS

- สามารถแยก Process ของตัวเองออกจาก Process อื่น ๆ
- มีความเร็วในการประมวลผลภาพจำนวนมากได้ แต่จะซ้ำเมื่อต้องประมวลผลภาพจำนวนมากน้อย



API

Preview API

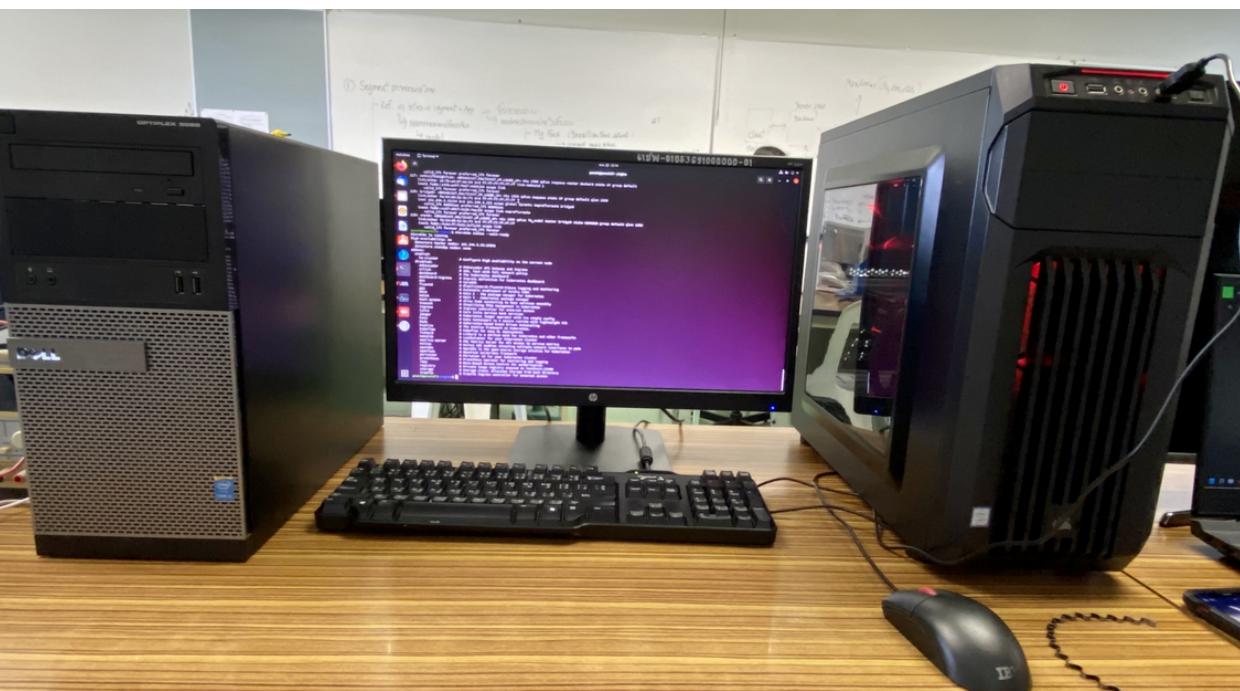
- สามารถรับการ Request จากผู้ใช้ได้ทันทีเมื่อมีการส่งเข้ามา
- มีความเร็วในการประมวลผลภาพจำนวน 1 ภาพได้เร็ว และ Response กลับไปได้ทันทีเมื่อมีการประมวลผลเสร็จสิ้น



Server

ติดตั้ง Server (Master node)

ติดตั้ง OS Ubuntu 22.04 LTS
Spec: i9-9900K, Ram 32 GB



ติดตั้ง Server (Worker node)

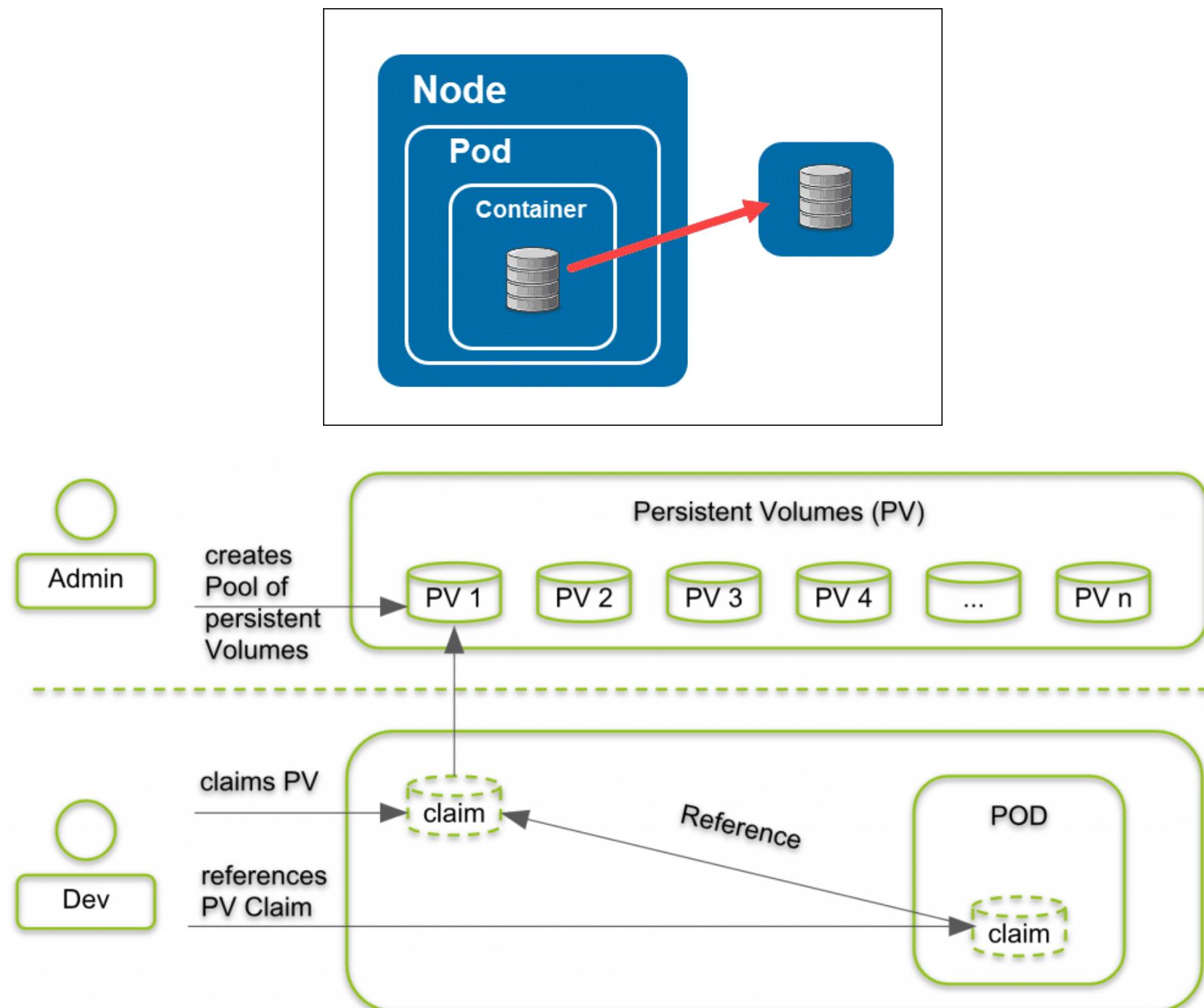
ติดตั้ง OS Ubuntu 22.04 LTS
Spec: i5-8300H, Ram 16 GB





Server

จัดการตั้งค่า Storage โดยใช้ Persistent Volume สำหรับจัดการข้อมูล





Server

NOSQL: MongoDB

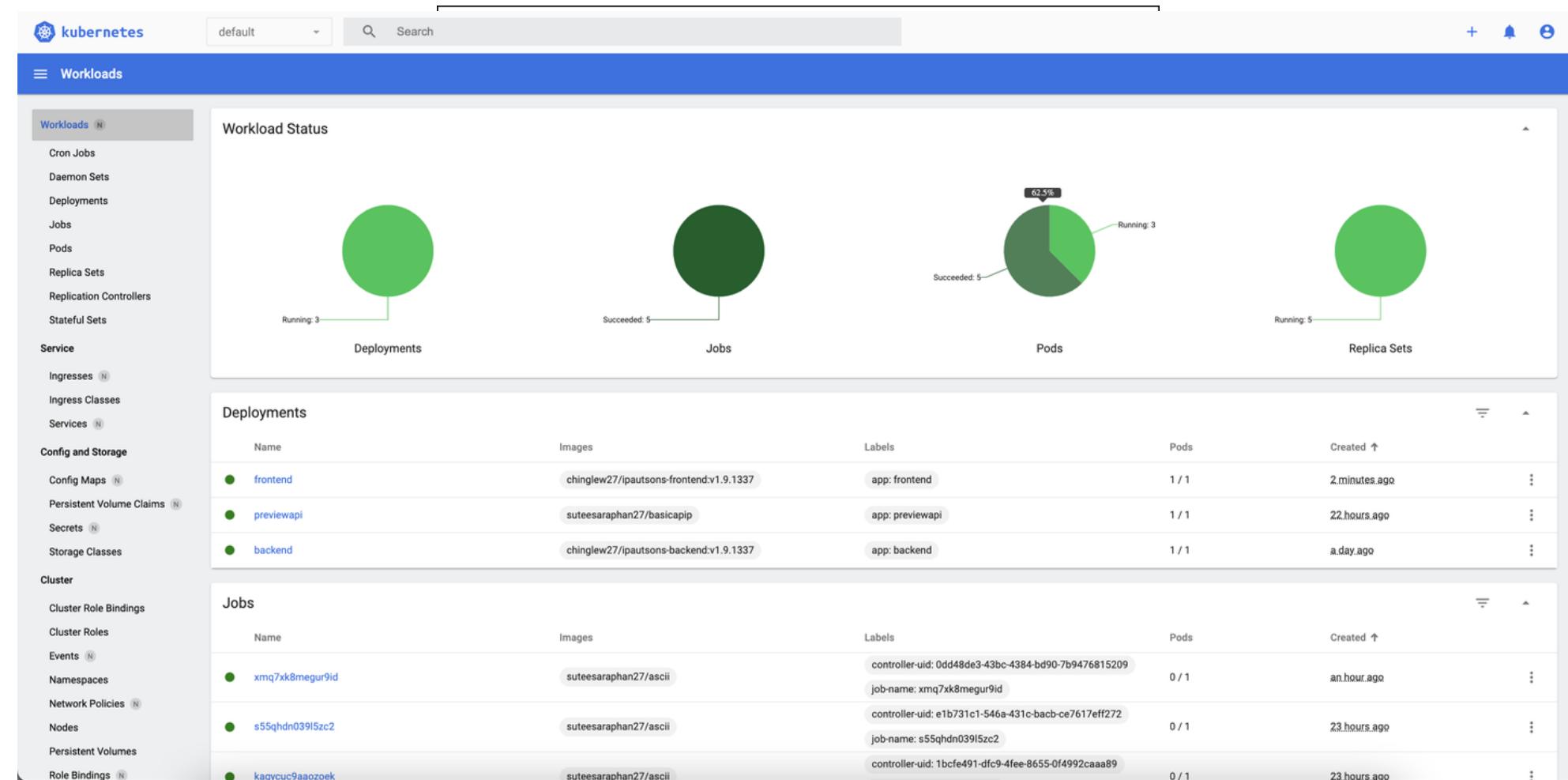


- cloud icon รองรับ Workload จำนวนมากได้ เหมาะแก่การ Scale ในแนวตั้ง(เพิ่มจำนวน Server)
- cloud icon เหมาะแก่การทำงานของข้อมูลที่มีการเปลี่ยนแปลงบ่อยในการแก้ไข หรือ ต้องการอัพเดตในอนาคต
- cloud icon Partition tolerance ระบบจะยังทำงานต่อได้ แม้ว่าจะมีหนึ่งใน Database ขาดการเชื่อมต่อ หรือเสียหายไป



Server

จัดการตั้งค่า Dashboard และ Role ต่าง ๆ ในการทำงานของการสั่งงานบน K8S

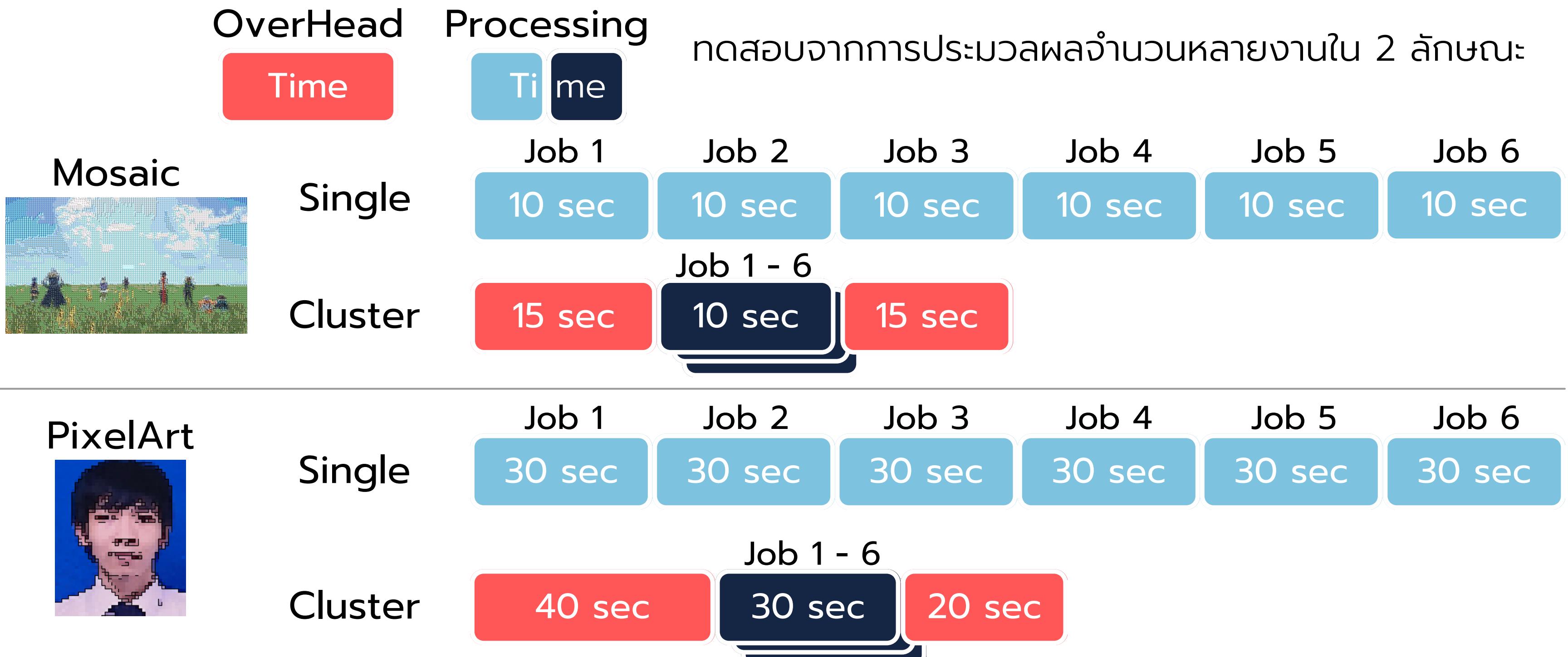


- WorkLoad: Jobs, Pods, Deployment
- Service: Loadbalancer, Expose, Internal
- กรณีที่ใช้งานอยู่



Server

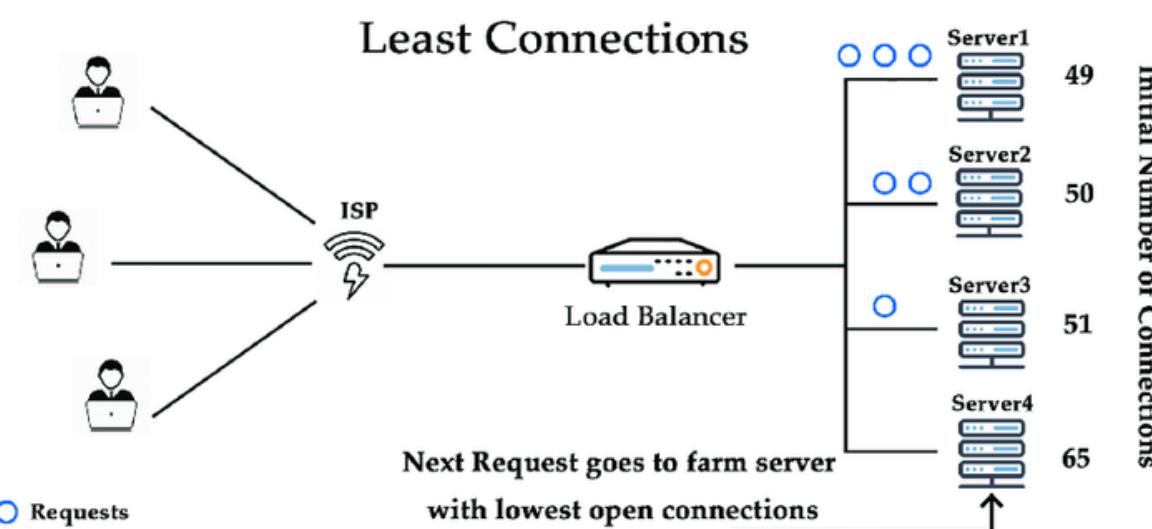
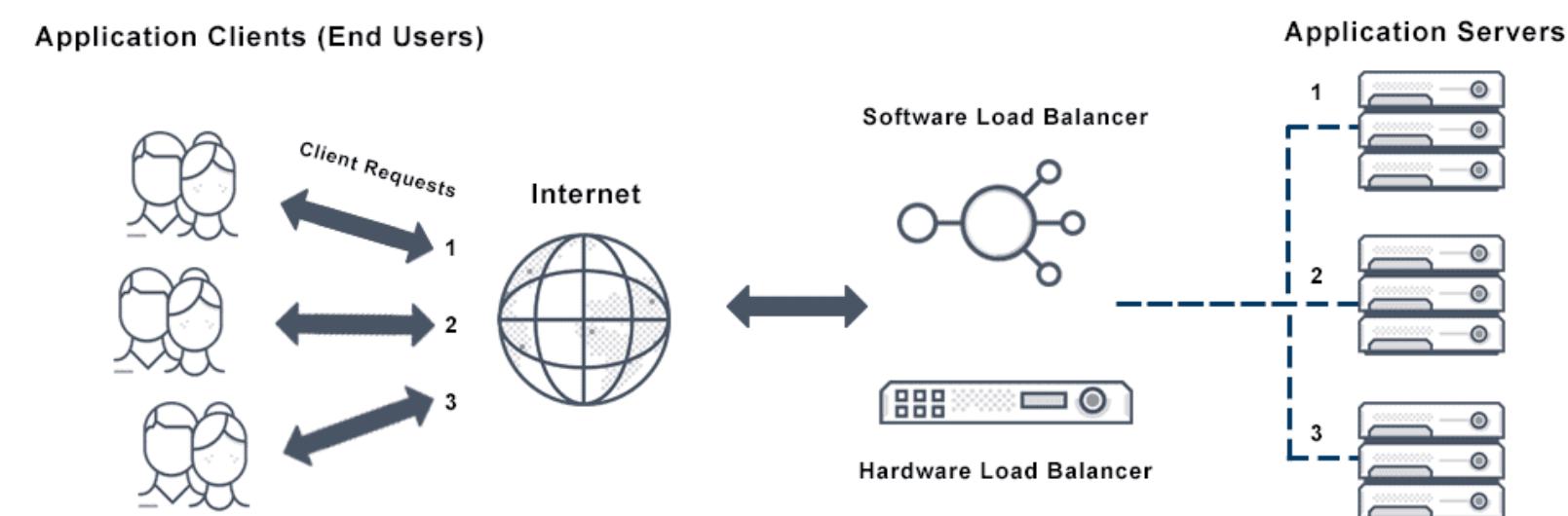
เปรียบเทียบงานการมีระบบ Cluster และ Load Balancer





Server

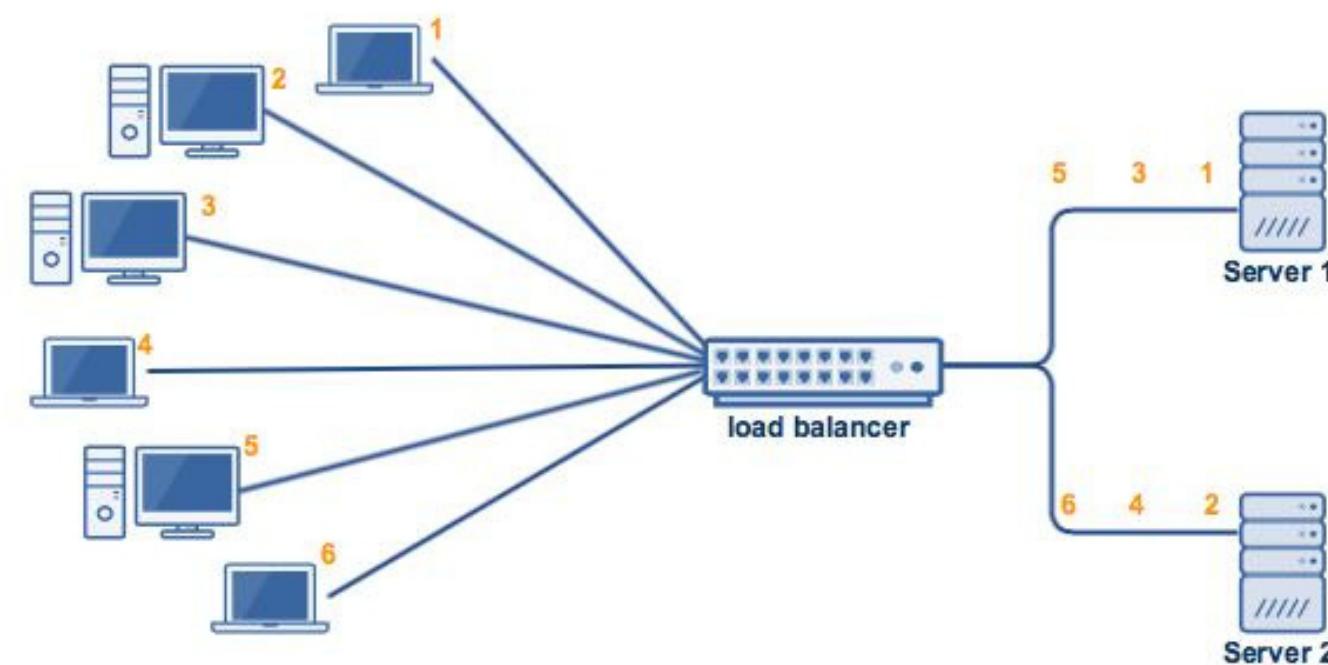
จัดการ Load Balance งานประมวลผลใน Kubernetes ด้วย Round Robin และ Least Connection





Server

Load Balance Round Robin

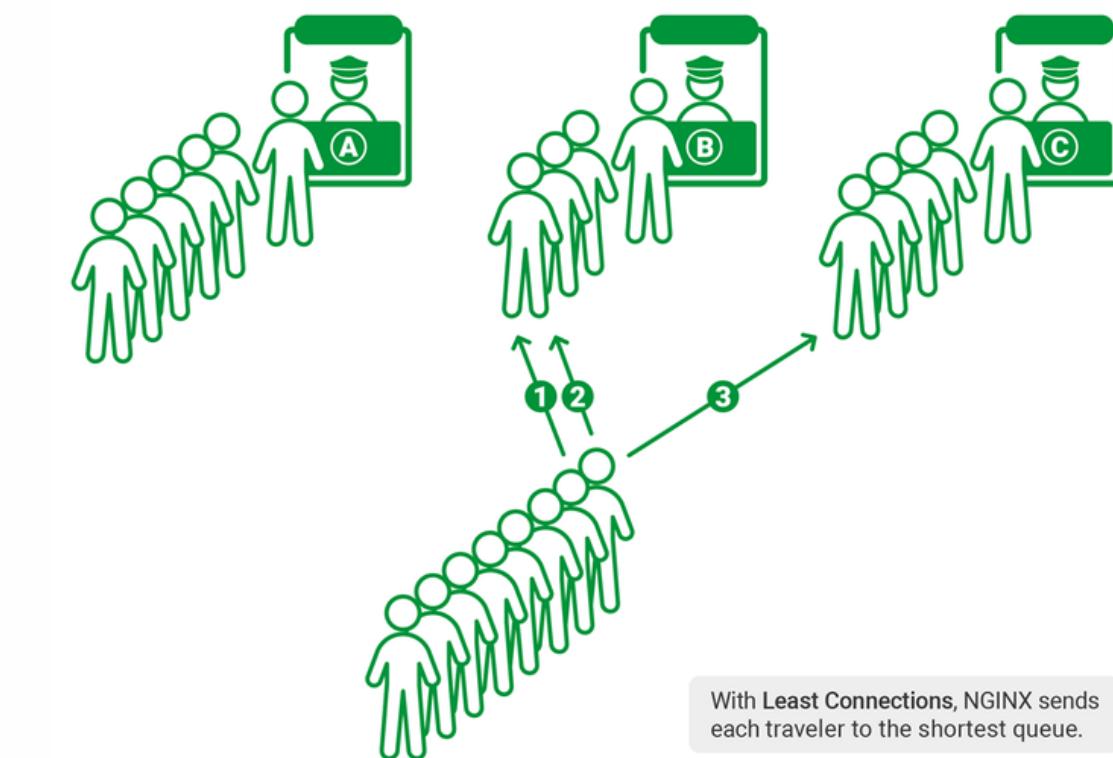
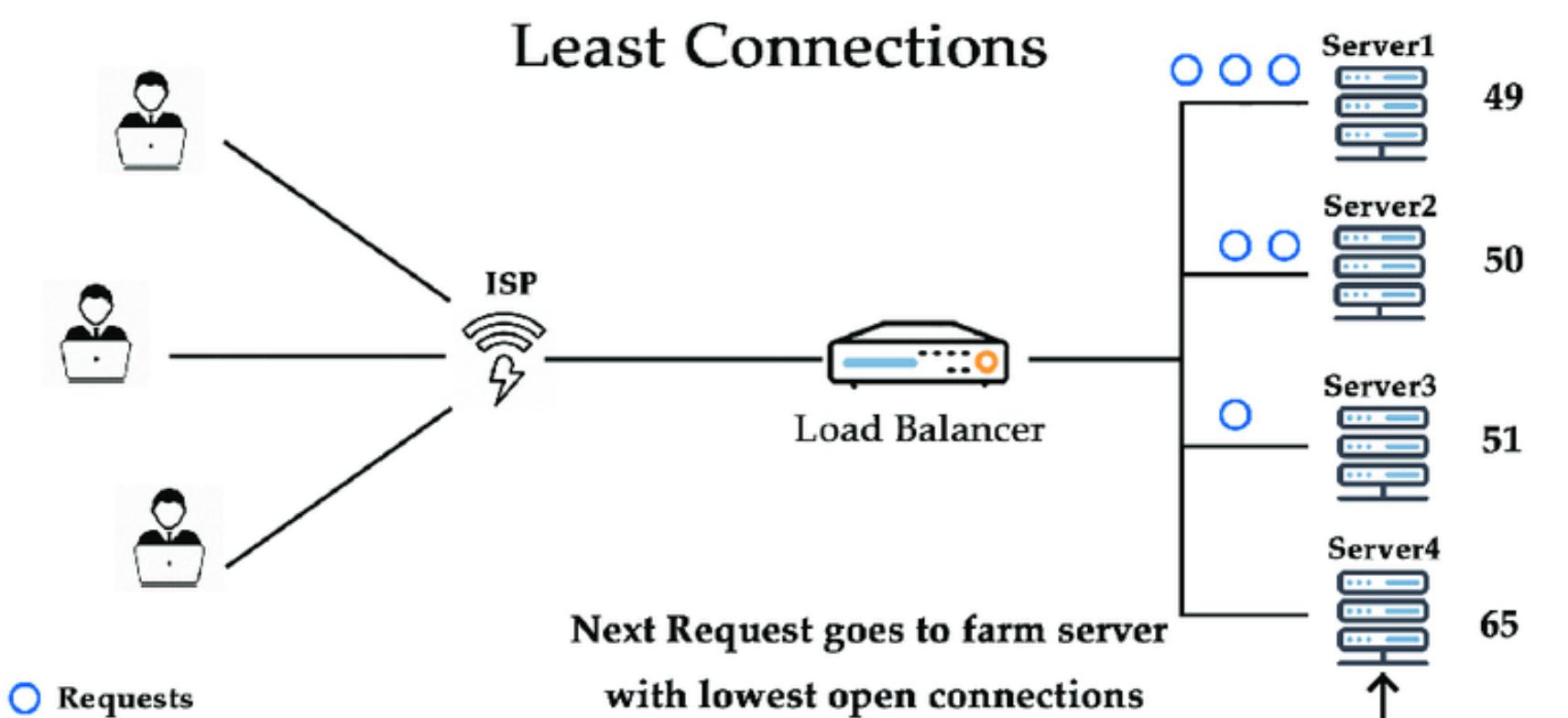


- เป็นวิธีการหนึ่งของ load balancing
- คำขอที่เข้ามามาจะถูกกระจายอย่างเท่าเทียมกันในกลุ่มของเซิร์ฟเวอร์ โดยแต่ละเซิร์ฟเวอร์จะได้รับคำขอตามลำดับ
- เป็นวิธีที่ง่ายและมีประสิทธิภาพในการกระจายโหลดระหว่างกลุ่มของทรัพยากร และทำให้แน่ใจว่าไม่มีเซิร์ฟเวอร์ใดถูกใช้งานมากเกินไป



Server

Load Balance Least Connection

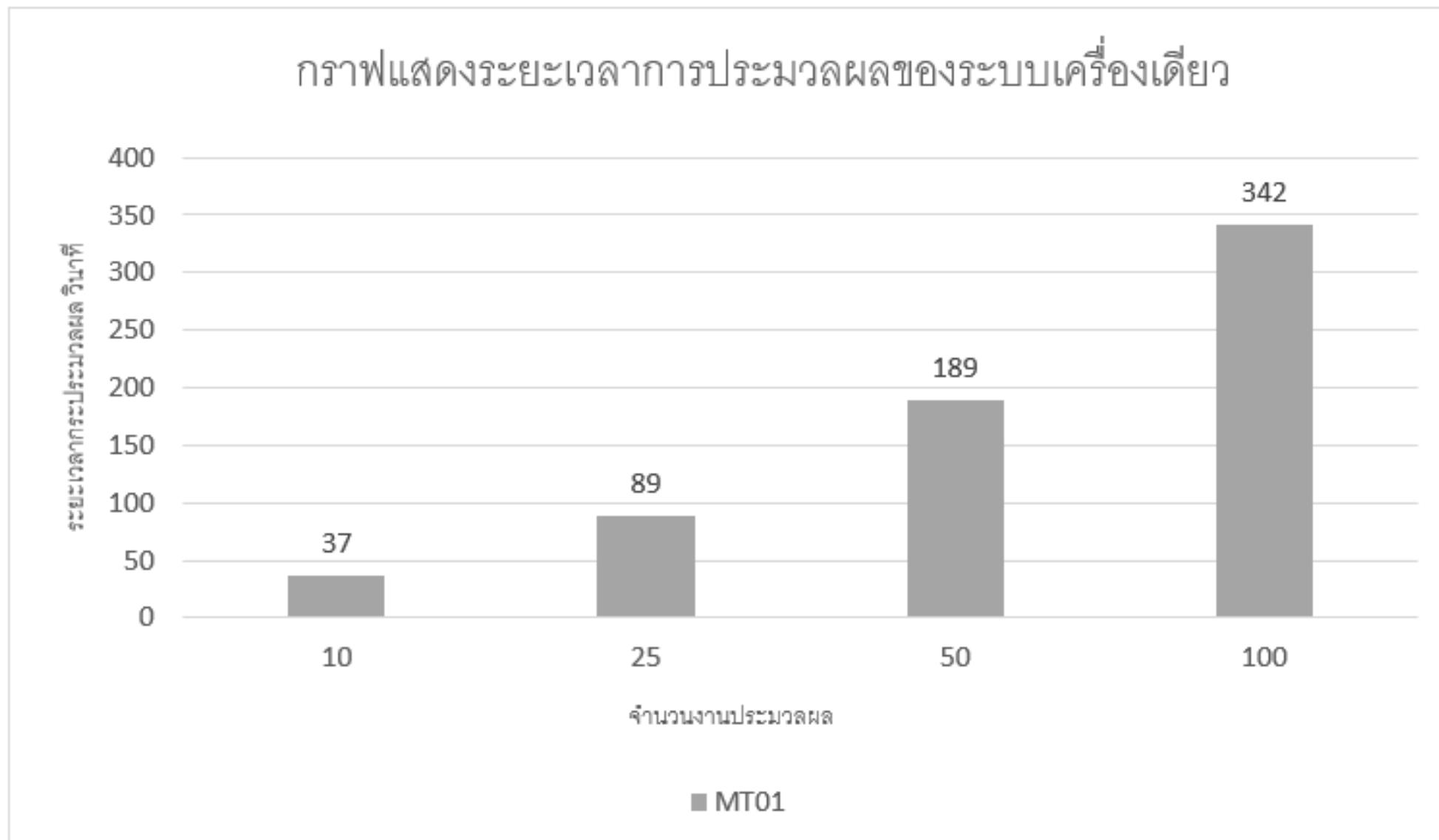


- เป็นวิธีการหนึ่งของ load balancing
- คำขอดีเข้ามาจะถูกกระจายไปยังเครื่อง Server ที่มีการเชื่อมต่อน้อยที่สุด หรือ อีกในคือมีการเข้าແนวนน้อยที่สุด หรือ มีจำนวนงานที่ Active อยู่น้อยที่สุด
- เป็นวิธีที่มีประสิทธิภาพในการกระจายโหลดระหว่างกลุ่มของทรัพยากรที่แตกต่างกันเพื่อกระจายโหลดให้มีจำนวนระยะเวลาใกล้เคียงกัน



Server

กราฟแสดงระยะเวลาการประมวลผลของระบบเครื่องเดียว



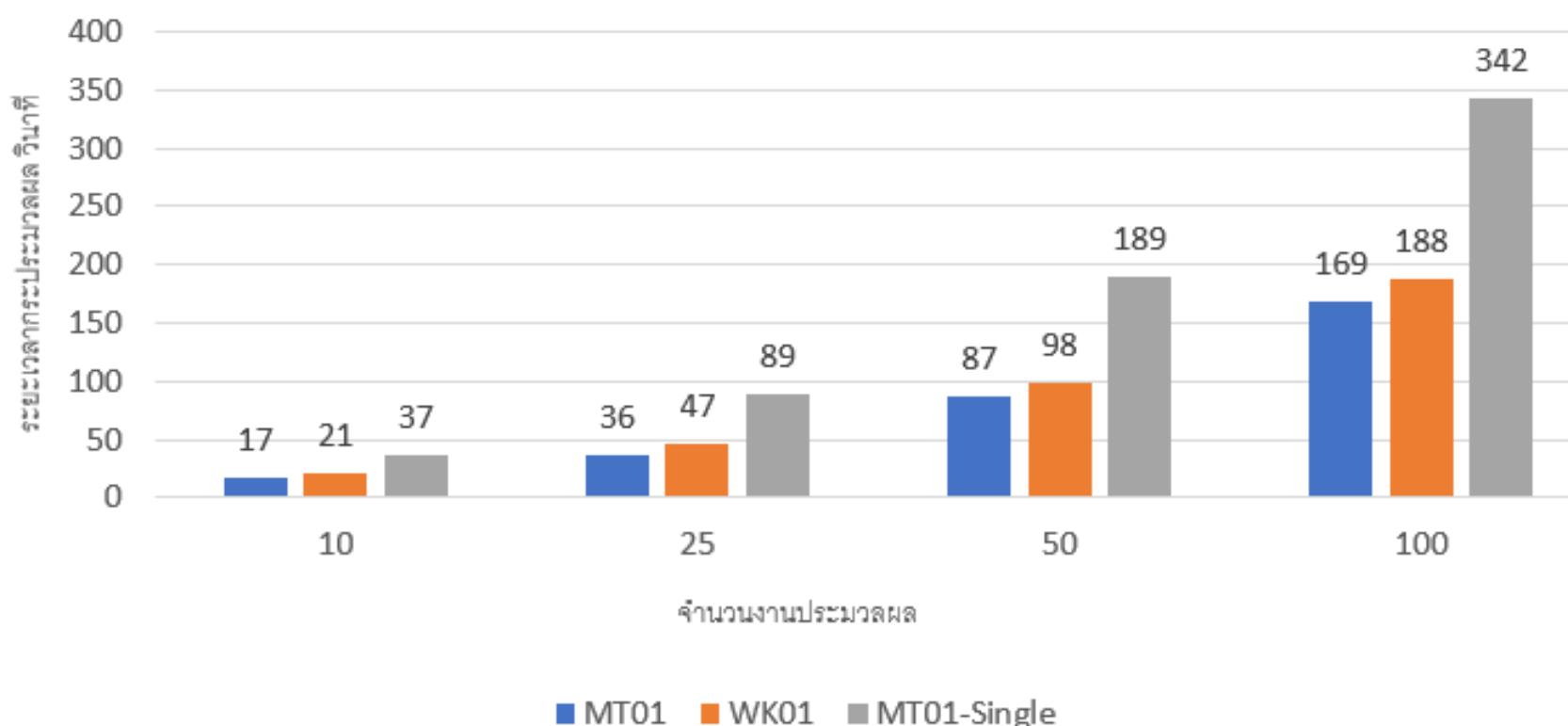
กราฟสรุประยะเวลาการทำงานของ
การกำแบบเครื่องเดียว
โดยมีการรับงานมาทำงานจำนวน



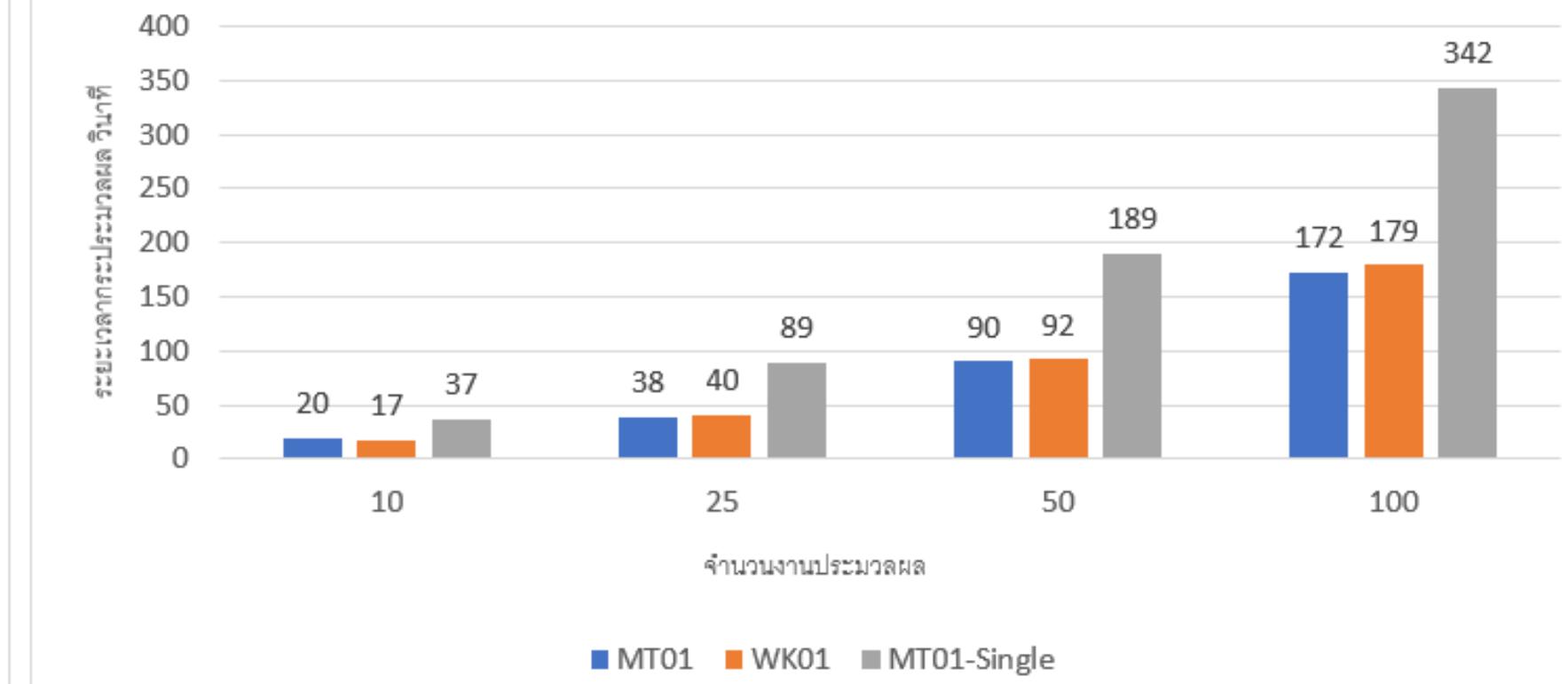
Server

กราฟแสดงระยะเวลาการประมวลผลของระบบคลัสเตอร์ ด้วย Load Balancer กึ่ง 2 Algorithm

กราฟแสดงระยะเวลาการประมวลผลของระบบคลัสเตอร์
Round Robin



กราฟแสดงระยะเวลาการประมวลผลของระบบคลัสเตอร์
Least Connection



Round Robin

Least Connection



Server

สรุป Load Balancer

- ในรูปแบบ Single เมฆะกับงานประมวลผลที่มีจำนวนงานที่น้อย และ เร็วกว่าการประมวลผลด้วยการแบ่งงานในรูปแบบ Cluster ด้วย Load Balancer
- Load Balancer ทั้ง 2 Algorithm ในงานที่ทำอยู่จะยังเห็นว่าระยะเวลาห่างกันไม่มากเนื่องด้วยการทดสอบใช้เครื่อง Server ที่กรวยการต่างกันจึงทำให้รูปแบบ Least Connection ยังเห็นผลไม่ได้มากแต่ในกรณีที่จำนวนงานมากขึ้น ตัว Least Connection ก็ยังมีความเร็วที่ดีกว่า

สรุปความเร็วของระบบ Cluster เร็วกว่า Single

Round Robin

51% - 62%

Least Connection

53% - 66 %



Server

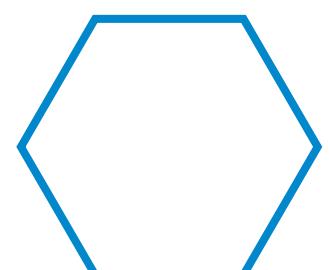
การทำงานของระบบ Cluster LoadBalancer



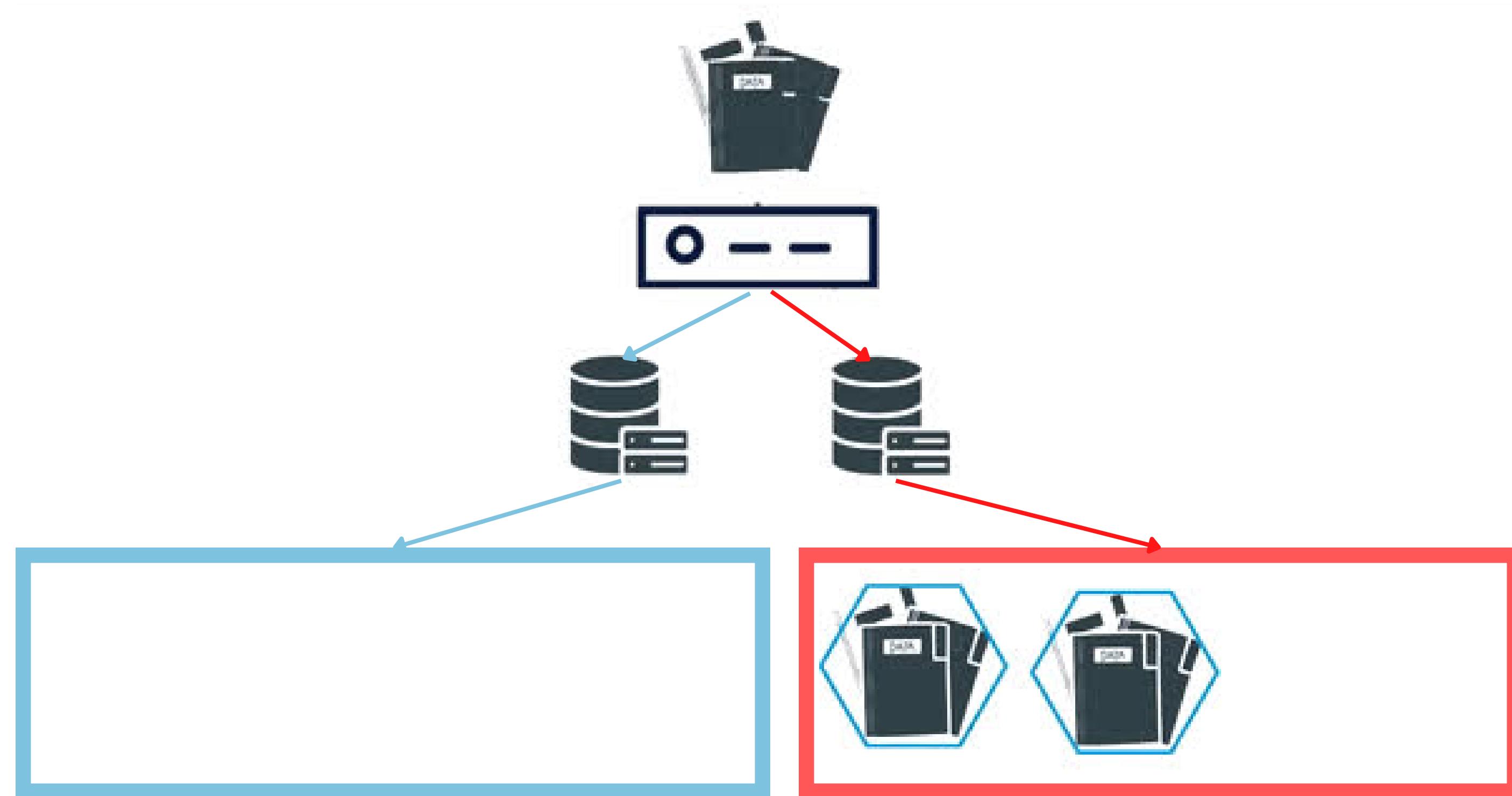
JOB



Server
Node

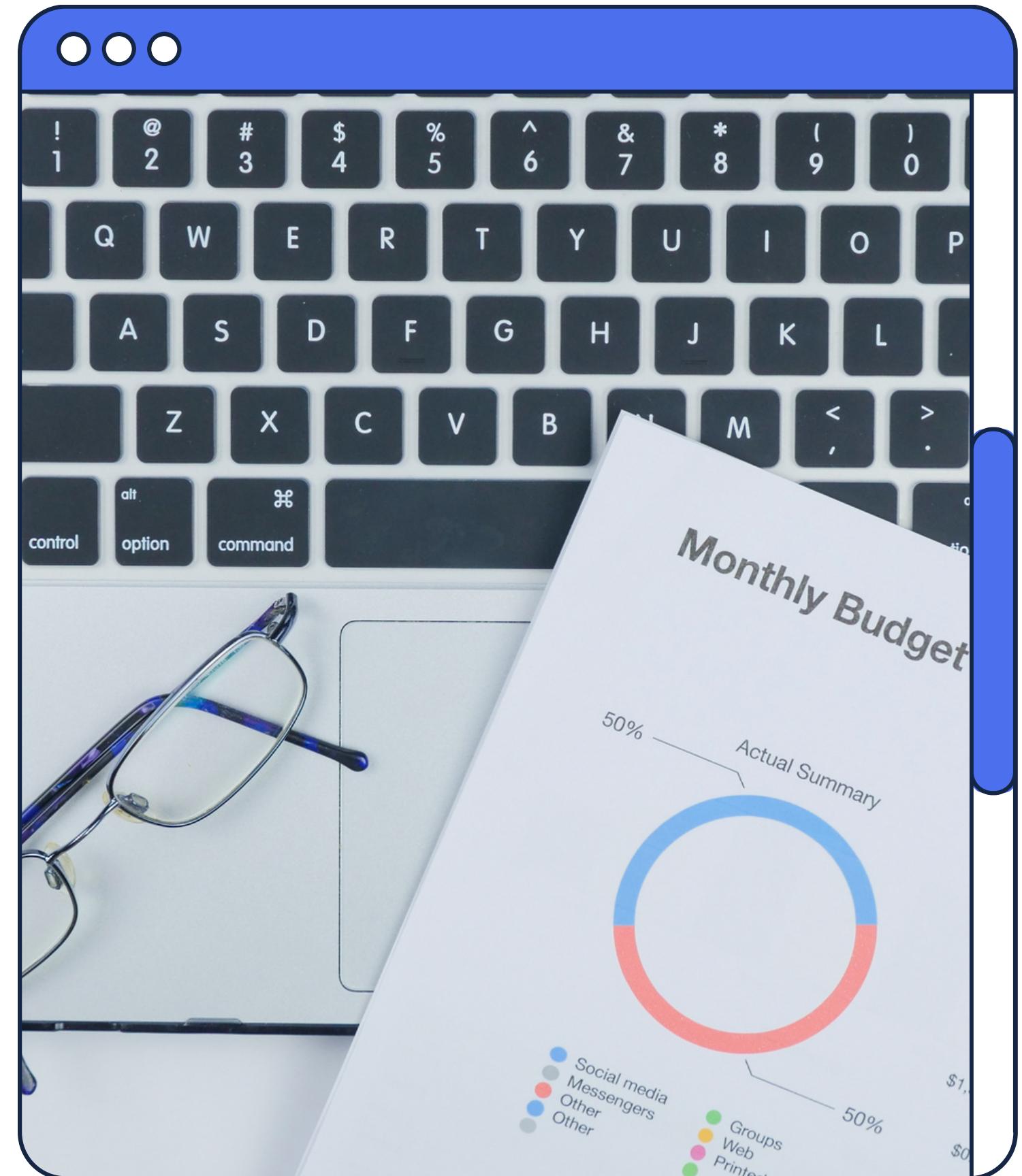


Container
Pod



สรุป

ปัญหาที่เกิดขึ้น
เป้าหมายงานที่ได้ดำเนินการไปแล้ว
ข้อจำกัดของโครงการ
แนวการพัฒนาต่อไปในอนาคต



ปัญหาที่เกิดขึ้น

แนวทางแก้ไข



Web application

- Library เสริมที่ใช้เกิด Conflict กันเองบ่อยครั้ง ก็ต้อง Version และการติดตั้ง
- มีการดึงข้อมูลจากฐานข้อมูลบ่อยครั้ง จึงทำให้จำนวน Traffic ดึงข้อมูลจำกัดและใช้งานต่อไม่ได้



Server & CaaS

- CaaS ที่ใช้จริงมีความต่างกันเรื่องเวลาการทำงาน และ การจัดการ
- อุปกรณ์ Hardware คอมพิวเตอร์มีทรัพยากรที่แตกต่างกันในการทดสอบ Load Balance
- Network ของสถาบัน



- ใช้ K3S และ MicroK8S

- จึงได้ทดลอง Load Balance จาก Round Robin เป็น Least Connection

- ใช้ Private Network ในการทดสอบแทน การใช้ Public Network ของสถาบัน

ເປົ້າມາຍົງນາທີ່ໄດ້ດຳເນີນກາຣໃປແລ້ວ

Web Application

-  Drive ເກີບຮູບພາວ
-  Image processing applications
-  Marketplace
-  ຮະບຸ Payment
-  ຮະບຸ User management

Image Processing App

-  Application ພື້ນຫຼານສໍາຮັບ ປະນວລ
ຜລ
-  API Preview
-  Model GANs ແລະ YOLOv5
-  API ສໍາຮັບ Model ກັ້ງປະນວລພລ ແລະ
Preview

Server

-  ກຳ Cluster ໃນຮະບຸ Local
-  Load balance ສໍາຮັບງານ
ປະນວລພລ
-  ຖດສອບ ແລະ ເກີບສົດຕິກາຣກຳງານ
Load Balancer

ข้อจำกัดของโครงการ

Web Application

- รองรับเฉพาะหน้าจอคอมฯ
- Structures ของ Directory แตกระดับได้ 1 ขั้น
- รองรับไฟล์เฉพาะนามสกุล JPG, PNG, TIFF และ Weight
- ไม่มีการ Share Drive

Image Processing App

- Weight ต้องรองรับกับ Model ที่เตรียมไว้ใน Marketplace เท่านั้น
- กำหนด Parameter ได้ 1 ต่อ 1 งานประมวลผลรูปจำนวนมาก

Server

- Cluster เป็นระบบ Local เท่านั้น
- เป็นการแบ่งงานในแต่ละเครื่อง ต่อ 1 งานประมวลผลเท่านั้น

แนวทางการพัฒนาต่อไปในอนาคต

Web Application

- เพิ่มความรวดเร็ว และ ลดจำนวนครั้งที่ ส่ง-รับข้อมูล
- แก้ไขในส่วน UX และ UI ให้สวยงาม และ เหมาะสมกับการใช้งานมากขึ้น
- นำระบบเติมเงินซื้อเครดิตเข้ามาใช้งาน
- แก้ไขเพิ่มเติมในส่วนของระบบเครดิต เพื่อให้การใช้งานสั่งงานประมวลผลนั้น คุ้มค่ากับ Client และผู้ให้บริการ

Server

- นำระบบ Cluster ให้เป็นระบบ Cloud
- Load Balancer อัลกอริทึมอื่น ๆ และ เก็บผลการทดลอง

Image Processing App

- แก้ไขให้สามารถรองรับ Model อื่นๆได้
- แก้ไขรูปแบบการโหลด Image Processing Application เพื่อลด Overhead time ทำให้การประมวลผลไวขึ้น

จบการนำเสนอ