

ลอนซ์ฮับ: ระบบให้บริการโฮสต์ตั้งสำหรับคณะเทคโนโลยีสารสนเทศ สจล.

LaunchHub: A hosting service system for IT KMITL

โดย

66070121 พงศธร สารีรูป

66070228 อินทิรา ธนัปประภักดิ์

66070286 ปณัสยา บุญประกอบ

โครงการนี้เป็นส่วนหนึ่งของวิชาเทคโนโลยีกลุ่มเมฆ

แขนงวิชาวิศวกรรมซอฟต์แวร์

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ภาคเรียนที่ 1 ปีการศึกษา 2568

COPYRIGHT 2025

FACULTY OF INFORMATION TECHNOLOGY

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

ใบรับรองโครงการ (PROJECT)

เรื่อง

ลอนซ์ฮับ: ระบบให้บริการโฮสต์ตั้งสำหรับคณะเทคโนโลยีสารสนเทศ สจล.

LaunchHub: A hosting service system for IT KMITL

นายพงศธร สารีรูป	รหัสประจำตัว 66070121
นางสาวอินทิรา ธนัพประภักดิ์	รหัสประจำตัว 66070228
นางสาวปณัสยา บุญประกอบ	รหัสประจำตัว 66070286

ขอรับรองว่ารายงานฉบับนี้ ข้าพเจ้าไม่ได้คัดลอกมาจากที่ใด
รายงานฉบับนี้ได้รับการตรวจสอบและอนุมัติให้เป็นส่วนหนึ่งของการ
การศึกษาวิชาเทคโนโลยีกลุ่มเมฆ แขนงวิชาวิศวกรรมซอฟต์แวร์
คณะเทคโนโลยีสารสนเทศ
ภาคเรียนที่ 1 ปีการศึกษา 2568

.....
(นายพงศธร สารีรูป)

.....
(นางสาวอินทิรา ธนัพประภักดิ์)

.....
(นางสาวปณัสยา บุญประกอบ)

หัวข้อโครงการ (ภาษาไทย) ลอนซ์ฮับ
(ภาษาอังกฤษ) LaunchHub
นักศึกษา นาย พงศธร สารีรูป
นางสาว อินทิรา ธนัพประภัสร์
นางสาว ปณัสยา บุญประกอบ
แขนงวิชา วิศวกรรมซอฟต์แวร์
ปีการศึกษา 1/2568

บทคัดย่อ

ในปัจจุบันนักศึกษาคณะเทคโนโลยีสารสนเทศ สจล. มักประสบปัญหาในการเผยแพร่โครงการ อันเนื่องมาจากขั้นตอนที่ยุ่งยากและค่าใช้จ่ายสูง ด้วยเหตุนี้ คณะผู้จัดทำจึงได้พัฒนา LunchHub ซึ่งเป็น PaaS (Platform as a Service) ที่ให้บริการด้านการเผยแพร่โครงการเว็บแอปพลิเคชันโดยเฉพาะ โดยมีวัตถุประสงค์หลักคือเพื่อแก้ปัญหาการเผยแพร่โครงการเว็บแอปพลิเคชันของนักศึกษา อีกทั้งยังอำนวยความสะดวกแก่บุคลากรที่เกี่ยวข้อง อาทิ อาจารย์ ให้สามารถเผยแพร่เว็บแอปพลิเคชันได้ง่ายยิ่งขึ้น

คณะผู้จัดทำหวังเป็นอย่างยิ่งว่าโครงการนี้จะให้ประโยชน์แก่ผู้ใช้บริการไม่มากก็น้อย และช่วยสนับสนุนการพัฒนานวัตกรรมภายในคณะเทคโนโลยีสารสนเทศ สจล.

กิตติกรรมประกาศ

โครงการนี้สามารถสำเร็จลุล่วงได้ด้วยความกรุณาจากอาจารย์ที่ปรึกษารายวิชาเทคโนโลยีกลุ่มเมฆ ดร. พัฒนพงษ์ ฉันทมิตรโอภาส และ ผศ. ดร. ลภัส ประดิษฐ์ศุภณีย์ ที่คอยให้คำปรึกษา ให้คำแนะนำ แนวคิด และการสนับสนุนโครงการตั้งแต่เริ่มต้น นอกจากนี้ คณะผู้จัดทำขอขอบพระคุณ คณาจารย์ทุกท่านจาก คณะเทคโนโลยีสารสนเทศ ที่ได้ประสิทธิ์ประสาทวิชาความรู้ และเป็นแรงบันดาลใจให้แก่พวกเราตลอดมา ขอขอบพระคุณ เพื่อน ๆ และครอบครัว ที่ให้กำลังใจและสนับสนุนมาโดยตลอด

สุดท้ายนี้ คุณค่าและประโยชน์อันพึงมีจากโครงการฉบับนี้ คณะผู้จัดทำขอมอบแด่ผู้มีส่วนเกี่ยวข้องทุกท่าน และหวังเป็นอย่างยิ่งว่าผลงานชิ้นนี้จะเป็นประโยชน์ต่อผู้ที่สนใจต่อไป

สารบัญ

	หน้า
บทคัดย่อ.....	ก
กิตติกรรมประกาศ	ข
สารบัญ	ค
สารบัญรูปภาพ	จ

บทที่

1. บทนำ	1
1.1 ความเป็นมา	1
1.2 วัตถุประสงค์และเป้าหมาย	1
1.3 ขอบเขตของระบบ	1
1.4 ประโยชน์ที่คาดว่าจะได้รับ	1
1.5 โครงสร้างของรายงาน	2
2. ทฤษฎีที่เกี่ยวข้อง.....	3
2.1 สถาปัตยกรรม อเมซอน คลาวด์เทคโนโลยี.....	3
2.1.1 Amazon Web Services (AWS)	3
2.1.2 Amazon S3 (Simple Storage Service)	3
2.1.3 Amazon CloudFront	4
2.1.4 Amazon Route 53	5
2.1.5 AWS Lambda	6
2.1.6 Amazon Relational Database Service (RDS).....	7
2.1.7 Amazon Elastic Container Registry (ECR).....	8
2.1.8 AWS Fargate.....	9
2.1.9 Amazon Cognito.....	10
2.1.10 AWS Application Load Balancer (ALB).....	11

2.2 PaaS, SaaS และ IaaS.....	12
2.3 แนวคิด Serverless Architecture	12
2.4 Continuous Integration / Continuous Deployment (CI/CD)	12
2.5 อื่น ๆ.....	12
2.5.1 ภาษาที่ใช้.....	12
3. ขั้นตอนการดำเนินงานของโครงการ	13
3.1 ความต้องการของระบบ	13
3.1.1 Functional Requirements	13
3.1.2 Non-Functional Requirements.....	14
3.2 การวิเคราะห์และออกแบบระบบคลาวด์แอปพลิเคชัน.....	15
3.2.1 Use Case Diagram.....	15
3.3 หลักการทำงานของระบบ.....	16
3.4 Mockup UI.....	18

สารบัญรูปภาพ

รูปที่	หน้า
รูปภาพที่ 1 AMAZON S3	3
รูปภาพที่ 2 AMAZON CLOUDFRONT	4
รูปภาพที่ 3 AMAZON ROUTE 53	5
รูปภาพที่ 4 AWS LAMBDA	6
รูปภาพที่ 5 AMAZON RDS	7
รูปภาพที่ 6 AMAZON ECR	8
รูปภาพที่ 7 AWS FARGATE	9
รูปภาพที่ 8 AMAZON COGNITO	10
รูปภาพที่ 9 AWS APPLICATION LOAD BALANCER (ALB)	11
รูปภาพที่ 10 USE CASE DIAGRAM	15
รูปภาพที่ 11 แผนภาพแสดงขั้นตอนการเข้าถึงหน้าแพลตฟอร์มหลัก	16
รูปภาพที่ 12 แผนภาพแสดงขั้นตอนการทำงานเมื่อผู้ใช้เข้ามา DEPLOY	16
รูปภาพที่ 13 ระบบ CI/CD ของแพลตฟอร์ม	17

บทที่ 1

บทนำ

1.1 ความเป็นมา

เนื่องด้วยนักศึกษาคณะเทคโนโลยีสารสนเทศ สจล. จำเป็นต้องใช้บริการแพลตฟอร์มประเภท PaaS (Platform as a Service) เพื่อเผยแพร่และทดลองใช้งานโครงงานของตนเองบนระบบจริงเพื่อวัตถุประสงค์ทางการศึกษา แต่กระบวนการในการนำระบบขึ้นเผยแพร่ผ่านช่องทางของคณะฯ มักประสบข้อจำกัด ทั้งในด้านขั้นตอนที่ยุ่งยากและค่าใช้จ่ายในการเช่าใช้บริการเซิร์ฟเวอร์หรือโดเมน ซึ่งนักศึกษาส่วนใหญ่ไม่สามารถแบกรับภาระดังกล่าวได้ ด้วยเหตุนี้จึงได้มีการพัฒนาแพลตฟอร์มนี้ขึ้น เพื่อเป็นทางเลือกสำหรับนักศึกษาในการนำเสนอโครงงานและทดลองใช้งานจริงในสภาพแวดล้อมที่ใกล้เคียงกับระบบการใช้งานจริง ทั้งยังช่วยลดอุปสรรคด้านเทคนิคและค่าใช้จ่ายอีกด้วย

1.2 วัตถุประสงค์และเป้าหมาย

- เพื่อพัฒนาแพลตฟอร์มประเภท PaaS (Platform as a Service) ในการเผยแพร่ (Deploy) โครงงานเว็บแอปพลิเคชัน ด้วย AWS (Amazon Web Service)
- เพื่อแก้ปัญหาที่ยุ่งยากและค่าใช้จ่ายในการเผยแพร่โครงงานเว็บแอปพลิเคชัน สำหรับนักศึกษาและบุคลากรคณะเทคโนโลยีสารสนเทศ สจล.

1.3 ขอบเขตของระบบ

โครงงานนี้มุ่งเน้นการ Deploy web-based application เพื่อให้พร้อมใช้งานจริง โดยมีการจำกัดการใช้งานเฉพาะบุคลากรภายในคณะเทคโนโลยีสารสนเทศ สจล.

LaunchHub แบ่งบทบาทการใช้งานหลักเป็น 2 ระดับ คือ

- Admin คือ ผู้ดูแลระบบที่สามารถเข้าถึงระบบหลังบ้านได้
- User คือ นักศึกษาและบุคลากรภายในคณะเทคโนโลยีสารสนเทศ สจล.

1.4 ประโยชน์ที่คาดว่าจะได้รับ

- ผู้ใช้สามารถนำเว็บแอปพลิเคชันของตนเองขึ้นสู่ระบบจริงได้อย่างสะดวกยิ่งขึ้น
- นักศึกษาคณะเทคโนโลยีสารสนเทศสามารถนำโครงงานของตนเองขึ้นเผยแพร่และทดลองใช้งานบนระบบจริงเพื่อวัตถุประสงค์ทางการศึกษาได้
- เป็นเครื่องมือสำหรับอาจารย์ในการพัฒนาและจัดการสื่อการสอนในรูปแบบเว็บไซต์
- ผู้จัดทำได้เรียนรู้เกี่ยวกับ CI/CD และมาประยุกต์ใช้จริงกับแพลตฟอร์ม PaaS (Platform as a Service)

1.5 โครงสร้างของรายงาน

รายงานฉบับนี้ มีโครงสร้าง ดังต่อไปนี้

บทที่ 1 กล่าวถึงความเป็นมาและความสำคัญของปัญหา วัตถุประสงค์ ขอบเขตของระบบ และประโยชน์ที่คาดว่าจะได้รับ

บทที่ 2 ทฤษฎีที่เกี่ยวข้องและเทคโนโลยีที่ได้นำมาใช้ประกอบการศึกษาและพัฒนาแอปพลิเคชัน

บทที่ 3 การออกแบบระบบ

บทที่ 2

ทฤษฎีที่เกี่ยวข้อง

2.1 สถาปัตยกรรม อเมซอน คลาวด์เทคโนโลยี

2.1.1 Amazon Web Services (AWS)

Amazon Web Services (AWS) เป็นบริการของ Amazon.com คือแพลตฟอร์มคลาวด์ที่ครอบคลุม และมีการนำไปใช้งานอย่างกว้างขวางที่สุดในโลก โดยให้บริการมากกว่า 200 รายการจากศูนย์ข้อมูลทั่วโลก เปิดให้บริการตั้งแต่ปีค.ศ. 2006 ด้านโครงสร้างพื้นฐานและเครื่องมือด้านไอทีผ่านอินเทอร์เน็ต โดยคิดค่าใช้จ่ายตามการใช้งานจริง (Pay-as-you-go)

2.1.2 Amazon S3 (Simple Storage Service)

Amazon S3 คือบริการ Object Storage ที่ให้ความสามารถในการปรับขนาด ความพร้อมใช้งาน ความปลอดภัย และประสิทธิภาพในระดับชั้นนำ S3 จัดเก็บข้อมูลในรูปแบบของ Object ซึ่งประกอบด้วยไฟล์ และ Metadata ที่อธิบายไฟล์นั้นๆ โดย Object ทั้งหมดจะถูกเก็บไว้ใน Bucket ผู้ใช้สามารถสร้าง Bucket ได้ตั้งแต่หนึ่ง Bucket ขึ้นไป และสามารถควบคุมการเข้าถึง, ดู Access Log และเลือก Region ที่ต้องการ จัดเก็บ Bucket นั้นได้ S3 สามารถจัดเก็บข้อมูลได้แทบทุกขนาด ตั้งแต่ขนาดเล็กไปจนถึง Exabytes โดยไม่จำเป็นต้อง Provision Storage ล่วงหน้า และผู้ใช้จะจ่ายเฉพาะส่วนที่ใช้งานจริง

จุดเด่นที่สำคัญของ S3 คือความทนทาน (Durability) ที่สูงถึง 99.999999999% (11 nines) และ ความพร้อมใช้งาน (Availability) ที่ 99.99% โดย AWS รับประกันด้วย Service Level Agreement (SLA) ที่แข็งแกร่งที่สุดในคลาวด์ เมื่อมีการอัปโหลดไฟล์ไปยัง S3 AWS จะทำการจำลองเนื้อหาเหล่านั้นไปยัง Data Center หลายแห่งโดยอัตโนมัติ ซึ่งช่วยให้เว็บไซต์ยังคงทำงานได้แม้ว่า Data Center ทั้งหมดจะเกิดความเสียหาย นอกจากนี้ S3 ยังเป็นตัวเลือกที่ยอดเยียมสำหรับการ Hosting เว็บไซต์แบบ Static เนื่องจากสามารถ ทำหน้าที่เป็น Web Server ได้โดยตรงสำหรับไฟล์ HTML, CSS, JavaScript และรูปภาพ โดยไม่ต้องจัดการ เซิร์ฟเวอร์

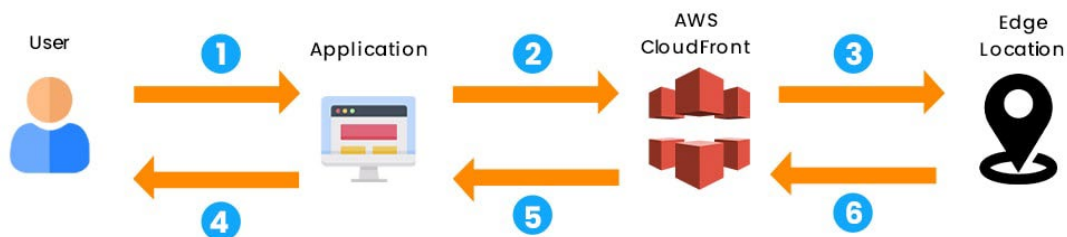


รูปภาพที่ 1 Amazon S3

2.1.3 Amazon CloudFront

Amazon CloudFront เป็นเครือข่ายส่งมอบเนื้อหา (Content Delivery Network หรือ CDN) ที่จัดการโดย AWS โดยใช้เครือข่ายศูนย์ข้อมูลทั่วโลกที่เรียกว่า Edge Locations เพื่อลดเวลาแฝง (Latency) และเพิ่มความเร็วในการถ่ายโอนข้อมูลสำหรับผู้ใช้งานปลายทาง

กลไกการทำงานของ CloudFront คือเมื่อผู้ใช้งานร้องขอเนื้อหา เว็บเซิร์ฟเวอร์จะถูกนำไปยัง Edge Location ที่ใกล้ที่สุด หากเนื้อหาที่ร้องขอไม่มีการ Cache อยู่ที่ Edge Location นั้น CloudFront จะไปดึงเนื้อหาจาก Origin (แหล่งที่มาของเนื้อหา) เช่น S3 Bucket, EC2 Instance หรือ Web Server ของลูกค้า หลังจากนั้น CloudFront จะเก็บสำเนาของเนื้อหาดังกล่าวไว้ที่ Edge Location นั้นเพื่อให้พร้อมใช้งานสำหรับการร้องขอในอนาคต การทำงานแบบนี้ช่วยให้เนื้อหาถูกส่งมอบอย่างรวดเร็วและมีประสิทธิภาพสูง CloudFront ยังสามารถใช้ในการส่งมอบเว็บไซต์หรือแอปพลิเคชันทั้งหมดได้ รวมถึงเนื้อหาแบบ Static, Dynamic, Streaming และ Interactive



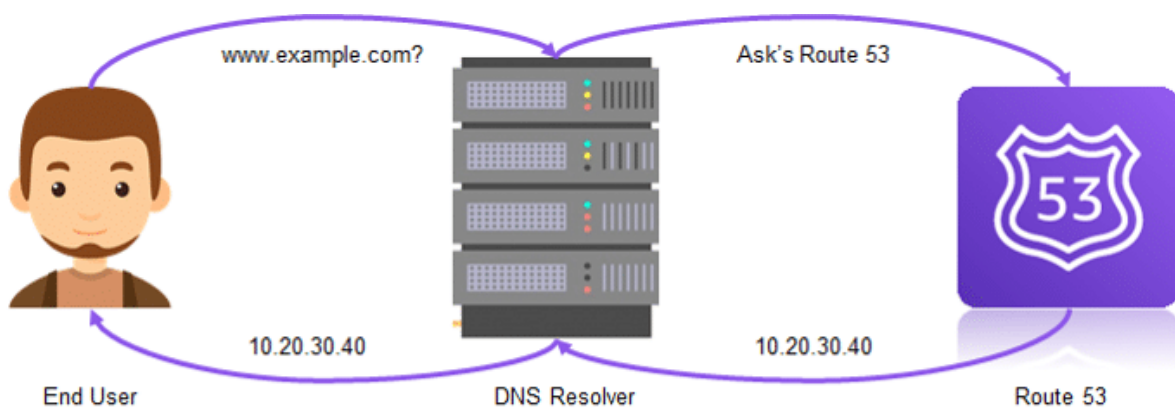
รูปภาพที่ 2 Amazon CloudFront

2.1.4 Amazon Route 53

Amazon Route 53 คือบริการ Domain Name System (DNS) แบบ Managed Service ที่มีความพร้อมใช้งานสูงและปรับขนาดได้ Route 53 ทำงานคล้ายกับสมุดโทรศัพท์ของอินเทอร์เน็ต โดยจะทำการแปลงชื่อโดเมนที่มนุษย์อ่านได้ เช่น example.com ให้เป็น IP Address ที่เครื่องจักรสามารถใช้งานได้ บริการนี้ช่วยให้ผู้ใช้สามารถเข้าถึงแอปพลิเคชันที่รันอยู่บน AWS หรือโครงสร้างพื้นฐานภายนอก

ผู้ใช้สามารถจัดการ DNS Record ใน Hosted Zone ที่ทำหน้าที่เป็น Container สำหรับ Name Servers ได้ หนึ่งในคุณสมบัติหลักของ Route 53 คือการกำหนดเส้นทาง (Routing Policies) ที่หลากหลาย ซึ่งช่วยให้สามารถควบคุมการกระจาย Traffic ตามเกณฑ์ต่าง ๆ ได้ นโยบายเหล่านี้รวมถึง

- **Simple Routing** เป็นนโยบายเริ่มต้นที่ใช้ในการส่งทราฟฟิกไปยังทรัพยากรเดียว
- **Latency Routing** ส่งทราฟฟิกไปยัง AWS Region ที่ให้เวลาแฝง (Latency) ต่ำที่สุดสำหรับผู้ใช้
- **Failover Routing** กำหนดให้ทราฟฟิกถูกส่งไปยังทรัพยากรสำรองเมื่อทรัพยากรหลักเกิดความเสียหาย
- **Geolocation Routing** ส่งทราฟฟิกตามตำแหน่งทางภูมิศาสตร์ของผู้ใช้ (เช่น ทวีป, ประเทศ)



รูปภาพที่ 3 Amazon Route 53

2.1.5 AWS Lambda

AWS Lambda คือบริการ Function-as-a-Service (FaaS) ที่ช่วยให้ผู้พัฒนาสามารถรันโค้ดได้โดยไม่ต้องทำการ Provision หรือจัดการเซิร์ฟเวอร์ โค้ดจะถูกรันในรูปแบบของฟังก์ชัน เพื่อตอบสนองต่อเหตุการณ์ (Event) ต่าง ๆ เช่น การอัปโหลดไฟล์ใน S3 หรือการเรียกใช้ API Lambda จะจัดการงานด้านการดูแลระบบทั้งหมด รวมถึงการบำรุงรักษาเซิร์ฟเวอร์และระบบปฏิบัติการ การจัดการ Capacity และการปรับขนาดอัตโนมัติ

คุณสมบัติหลักของ Lambda คือการปรับขนาดอัตโนมัติ (Auto-scaling) ซึ่งทำให้สามารถรองรับ Workload ได้ทุกระดับโดยไม่มีค่าใช้จ่ายสำหรับเวลาที่ไม่มีการใช้งาน อีกทั้งยังรองรับภาษาและ Runtime ที่หลากหลาย เช่น Node.js, Python, Java, Go และ C# การเรียกใช้ฟังก์ชันสามารถทำได้สองวิธีหลัก คือ แบบ Synchronous (รอการตอบกลับ) และ Asynchronous (ไม่ต้องรอการตอบกลับ) ในกรณีที่เกิดข้อผิดพลาด การเรียกใช้แบบ Asynchronous จะมีการจัดการการ Retry โดยอัตโนมัติ



AWS Lambda

รูปภาพที่ 4 AWS Lambda

2.1.6 Amazon Relational Database Service (RDS)

Amazon RDS เป็นบริการฐานข้อมูลเชิงสัมพันธ์แบบ Managed Service ที่ช่วยลดความยุ่งยากในการตั้งค่า ดำเนินการ และปรับขนาดฐานข้อมูล บริการนี้จัดการงานด้านการดูแลระบบที่ซับซ้อนและใช้เวลามาก เช่น การติดตั้งซอฟต์แวร์ การสำรองข้อมูล การทำ Replication เพื่อ High Availability และการอัปเดต

สำหรับ PostgreSQL, Amazon RDS มีคุณสมบัติเด่นที่ช่วยเพิ่มความน่าเชื่อถือและประสิทธิภาพ

- **High Availability:** Multi-AZ Deployments จะทำการสร้าง Instance สำรองใน Availability Zone อื่นโดยอัตโนมัติ ซึ่งช่วยเพิ่มความทนทานและความพร้อมใช้งานให้กับ Workload ที่ใช้ในการผลิต
- **การสำรองข้อมูลและการกู้คืน:** มีคุณสมบัติ Automated Backup ที่ช่วยให้สามารถกู้คืนฐานข้อมูลไปยังช่วงเวลาใดก็ได้ภายในระยะเวลาที่กำหนด และยังรองรับ User-initiated Backup ที่ถูกจัดเก็บไว้ใน S3
- **การปรับขนาด (Scalability):** Read Replicas ช่วยในการขยายขีดความสามารถเพื่อรองรับ Workload ที่เน้นการอ่านข้อมูลเป็นจำนวนมาก
- **ความปลอดภัย:** RDS มีการรักษาความปลอดภัยในระดับสูง รวมถึงการแยกเครือข่ายด้วย Virtual Private Cloud (VPC), การเข้ารหัสข้อมูลที่จัดเก็บ (Encryption at Rest) ด้วย AWS KMS และการเข้ารหัสข้อมูลระหว่างการส่ง (Encryption in Transit) ด้วย SSL



amazon
RDS

รูปภาพที่ 5 Amazon RDS

2.1.7 Amazon Elastic Container Registry (ECR)

Amazon ECR คือบริการ Container Registry แบบ Fully Managed Service ที่ใช้สำหรับจัดเก็บ จัดการ และปรับใช้ Docker และ Open Container Initiative (OCI) Images บริการนี้ถูกออกแบบมาให้ปรับ ขนาดได้และมีความปลอดภัยสูง โดยผสมรวมเข้ากับบริการอื่นๆ ของ AWS เช่น ECS, EKS และ AWS Lambda

องค์ประกอบหลักของ ECR ประกอบด้วย

- **Registry** กล่าวคือ AWS Account แต่ละบัญชีจะได้รับ Private Registry ซึ่งเป็นพื้นที่จัดเก็บที่ ปลอดภัยสำหรับ Repositories หลายรายการ
- **Repository** เป็นหัวใจหลักของ ECR ที่ทำหน้าที่เป็นพื้นที่เฉพาะสำหรับการจัดเก็บ Docker และ OCI Images
- **Image** คือ ไฟล์ที่เป็นองค์ประกอบสำคัญของแอปพลิเคชันแบบ Container ซึ่งสามารถถูก Push และ Pull โดยใช้ Docker CLI มาตรฐานได้



Amazon ECR

รูปภาพที่ 6 Amazon ECR

2.1.8 AWS Fargate

AWS Fargate คือ Compute Engine แบบ Serverless สำหรับ Amazon ECS ที่ช่วยให้ผู้พัฒนาสามารถรัน Container ได้โดยไม่ต้อง Provision, กำหนดค่า หรือจัดการ Cluster ของ EC2 Instance Fargate จัดการะในการจัดการเซิร์ฟเวอร์เบื้องหลัง, ระบบปฏิบัติการ, Security Patch และการปรับขนาด Capacity ของ Cluster ลงอย่างสิ้นเชิง ผู้ใช้เพียงแค่ต้องแพ็กแอปพลิเคชันใน Container และระบุ CPU และ Memory ที่ต้องการเท่านั้น

ความแตกต่างที่สำคัญระหว่าง Fargate และ EC2 คือในขณะที่ EC2 ผู้ใช้ต้องจัดการ Instance ด้วยตนเอง Fargate จะทำหน้าที่เป็น Engine ที่จัดการทรัพยากรเบื้องหลังทั้งหมด Fargate จะสร้างสภาพแวดล้อมที่แยกออกมาสำหรับแต่ละ Task (Container) ทำให้มีความปลอดภัยสูงและไม่มีการใช้ทรัพยากรร่วมกัน Fargate ทำงานร่วมกับ ECR โดยการดึง Image ที่ถูกจัดเก็บไว้ใน ECR Repository เพื่อนำมาสร้างเป็น Container Task



AWS Fargate

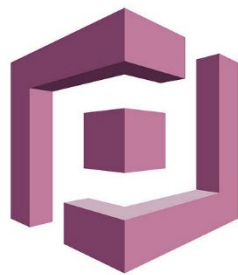
รูปภาพที่ 7 AWS Fargate

2.1.9 Amazon Cognito

Amazon Cognito เป็นบริการจัดการการยืนยันตัวตน (Authentication) และการกำหนดสิทธิ์การเข้าถึง (Authorization) บนคลาวด์ของ Amazon Web Services (AWS) ที่ช่วยให้นักพัฒนาสามารถเพิ่มฟังก์ชันการเข้าสู่ระบบ (Sign-in) และการลงทะเบียน (Sign-up) ให้กับแอปพลิเคชันได้อย่างรวดเร็วและปลอดภัย รองรับทั้งการยืนยันตัวตนด้วยบัญชีผู้ใช้ที่สร้างขึ้นภายในระบบ (User Pools) และการยืนยันตัวตนจากผู้ให้บริการภายนอก เช่น Google, Facebook, Apple, หรือผู้ให้บริการ SAML

Cognito มีองค์ประกอบหลัก 2 ส่วน คือ

- **User Pools** คือ บริการไดรเวทอรีผู้ใช้และระบบจัดการการยืนยันตัวตน ที่สามารถปรับแต่งนโยบายรหัสผ่าน การยืนยันตัวตนหลายขั้นตอน (MFA) และการตรวจสอบอีเมล/หมายเลขโทรศัพท์ได้
- **Identity Pools** คือ บริการสร้างและจัดการสิทธิ์การเข้าถึงทรัพยากร AWS โดยกำหนดสิทธิ์แยกตามผู้ใช้ที่ได้รับการยืนยันตัวตนและผู้ใช้แบบนิรนาม



Amazon Cognito

รูปภาพที่ 8 Amazon Cognito

2.1.10 AWS Application Load Balancer (ALB)

AWS Application Load Balancer (ALB) เป็นบริการภายใต้ Amazon Elastic Load Balancing (ELB) ที่ทำหน้าที่กระจายปริมาณการรับส่งข้อมูล (traffic) ไปยังหลาย ๆ เซิร์ฟเวอร์หรือ instance เพื่อเพิ่มประสิทธิภาพ ความเสถียร และความพร้อมใช้งานของระบบ ALB ถูกออกแบบมาสำหรับการทำงานใน Layer 7 (Application Layer) ของโมเดล OSI ทำให้สามารถตัดสินใจในการกระจายโหลดตามเนื้อหาของคำขอ (content-based routing) ได้ เช่น การกำหนดเส้นทางตาม URL, Hostname หรือ HTTP header



รูปภาพที่ 9 AWS Application Load Balancer (ALB)

2.2 PaaS, SaaS และ IaaS

Cloud Computing แบ่งการให้บริการหลัก ๆ ออกเป็น 3 รูปแบบ ดังนี้

- **Platform as a Service (PaaS)** คือ ผู้ให้บริการมีทั้งโครงสร้างพื้นฐานและเครื่องมือพัฒนาแอปพลิเคชัน เช่น runtime environment, database, และ CI/CD pipeline ผู้ใช้สามารถโฟกัสที่การพัฒนาและดีพลอย เช่น AWS Elastic Beanstalk, Google App Engine
- **Infrastructure as a Service (IaaS)** คือ ผู้ให้บริการจัดเตรียมโครงสร้างพื้นฐาน เช่น เซิร์ฟเวอร์, ระบบเครือข่าย, และ Storage ผู้ใช้รับผิดชอบการติดตั้งและจัดการซอฟต์แวร์เอง เช่น Amazon EC2, Google Compute Engine
- **Software as a Service (SaaS)** คือ ผู้ให้บริการดูแลทั้งโครงสร้างพื้นฐานและแอปพลิเคชันพร้อมใช้งานผ่านอินเทอร์เน็ต เช่น Google Workspace, Microsoft 365

2.3 แนวคิด Serverless Architecture

Serverless Architecture คือการออกแบบระบบที่ผู้พัฒนาไม่ต้องจัดการเซิร์ฟเวอร์โดยตรง ผู้ให้บริการคลาวด์จะบริหารทรัพยากรตามการใช้งานจริง (on-demand) ทำให้สามารถปรับขนาด (scaling) ได้อัตโนมัติและจ่ายเงินตามการเรียกใช้งาน ฟังก์ชันหลักจะถูกรันแบบ event-driven เช่น AWS Lambda, Azure Functions เหมาะสำหรับงานที่ต้องการความยืดหยุ่นและลดภาระการดูแลโครงสร้างพื้นฐาน

2.4 Continuous Integration / Continuous Deployment (CI/CD)

CI/CD คือแนวทางการพัฒนาและส่งมอบซอฟต์แวร์แบบอัตโนมัติ โดยมีรายละเอียด ดังนี้

Continuous Integration (CI) คือ การรวมโค้ดจากนักพัฒนาหลายคนเข้ากับ repository กลางบ่อย ๆ พร้อมรันการทดสอบอัตโนมัติเพื่อลดข้อผิดพลาด

Continuous Deployment (CD) คือ การดีพลอยซอฟต์แวร์ไปยังสภาพแวดล้อมจริง (production) แบบอัตโนมัติหลังผ่านการทดสอบ ทำให้การส่งมอบฟีเจอร์ใหม่เป็นไปอย่างรวดเร็วและต่อเนื่อง

2.5 อื่น ๆ

2.5.1 ภาษาที่ใช้

- HTML
- CSS
- Node.js (Express)

บทที่ 3

ขั้นตอนการดำเนินงานของโครงการ

3.1 ความต้องการของระบบ

3.1.1 Functional Requirements

ผู้ใช้ทั่วไป

- ผู้ใช้สามารถสร้างบัญชีใหม่ได้
- ผู้ใช้สามารถเข้าสู่ระบบด้วยบัญชีที่ลงทะเบียนไว้
- ผู้ใช้สามารถสร้าง แก้ไข และลบ Project ได้
- ผู้ใช้สามารถตรวจสอบสถานะปัจจุบันและข้อมูลสำคัญของ Project
- ผู้ใช้สามารถปรับแต่งการตั้งค่าของ Project ได้ตามสิทธิ์
- ผู้ใช้สามารถเข้าถึง log ของเว็บไซต์ที่ตนเอง deploy
- ผู้ใช้สามารถ deploy เว็บจาก Project โดยไม่ต้องจด domain name ด้วยตนเอง
- ระบบต้องประมวลผลการ deploy และแจ้งผลลัพธ์สำเร็จ/ล้มเหลว
- ระบบมีบริการดูแลเว็บไซต์ของผู้ใช้โดยอัตโนมัติ

ผู้ดูแลระบบ (Admin)

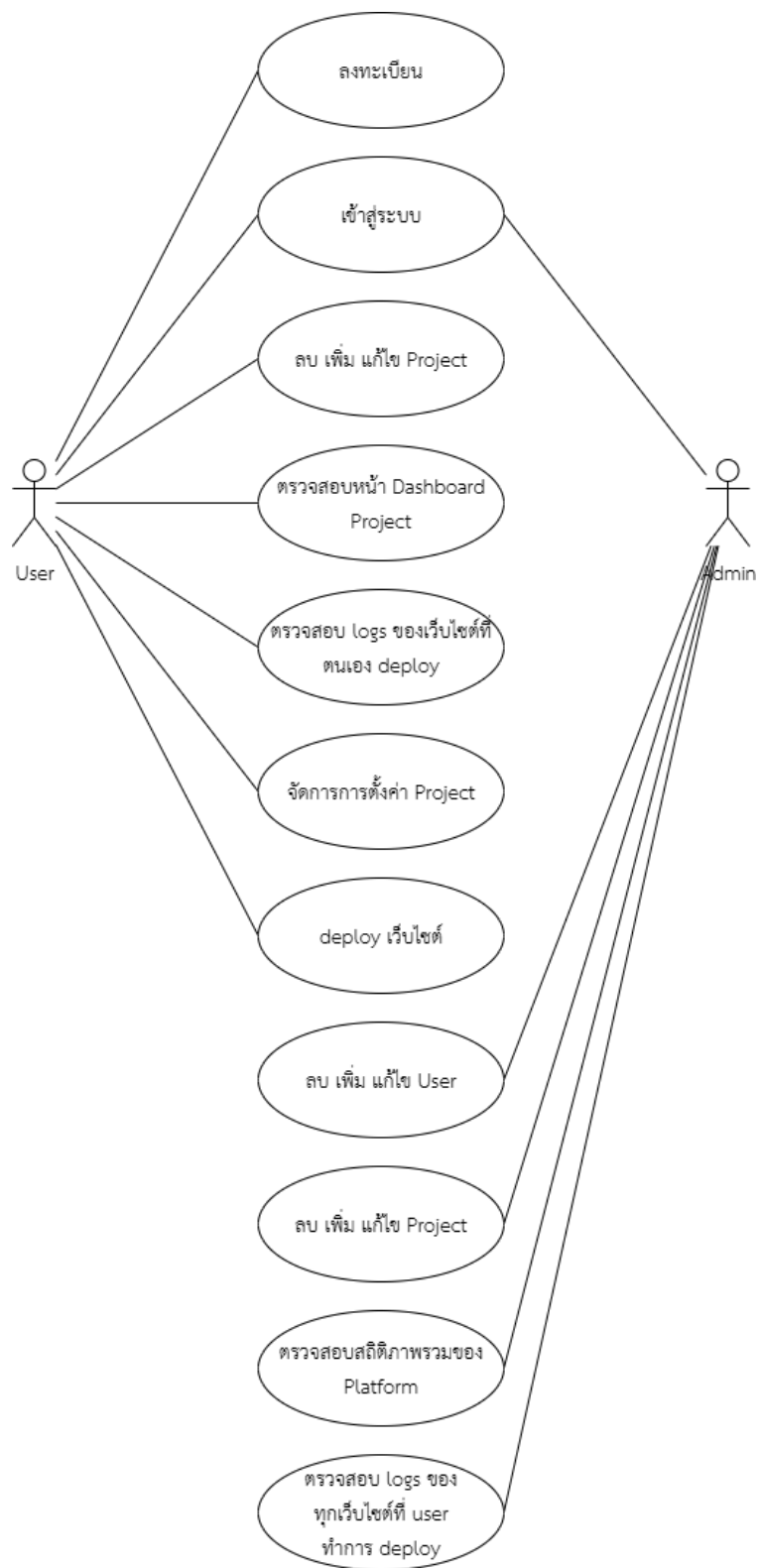
- ผู้ดูแลระบบสามารถเพิ่ม แก้ไขข้อมูล และลบผู้ใช้ได้
- ผู้ดูแลระบบสามารถดูสถิติการใช้งาน Platform ทั้งหมด
- ระบบต้องแสดงสถิติอย่างชัดเจนและเป็นปัจจุบัน
- ผู้ดูแลระบบสามารถเข้าถึง log ของทุกเว็บไซต์ที่ถูก deploy
- ระบบต้องแสดงข้อมูล log สำหรับวิเคราะห์ปัญหาและประเมินสถานะ

3.1.2 Non-Functional Requirements

- ระบบมีความเสถียร
- ระบบจะต้องพร้อมใช้งานอย่างต่อเนื่องตลอด 24/7
- ระบบต้องรองรับการเพิ่มขึ้นของผู้ใช้งานและ workload ที่ไม่แน่นอนได้
- ระบบต้องมีความปลอดภัยของข้อมูลผู้ใช้งาน
- ข้อมูลที่จัดเก็บในระบบต้องมีความคงทนสูงและไม่สูญหาย
- ระบบต้องง่ายต่อการบำรุงรักษาและแก้ไข
- ระบบต้องสามารถตรวจสอบการทำงานและ logs ต่าง ๆ ได้ง่าย
- ระบบต้องทำงานได้อย่างถูกต้องและสม่ำเสมอ ซึ่งฐานข้อมูลต้องมีความน่าเชื่อถือและสามารถสลับการทำงานจากเซิร์ฟเวอร์หลักที่เกิดปัญหา ไปยังเซิร์ฟเวอร์สำรองที่เตรียมพร้อมไว้โดยอัตโนมัติได้ในกรณีที่มีปัญหา
- ระบบต้องสามารถทนทานต่อความผิดพลาดของ service บางส่วนได้

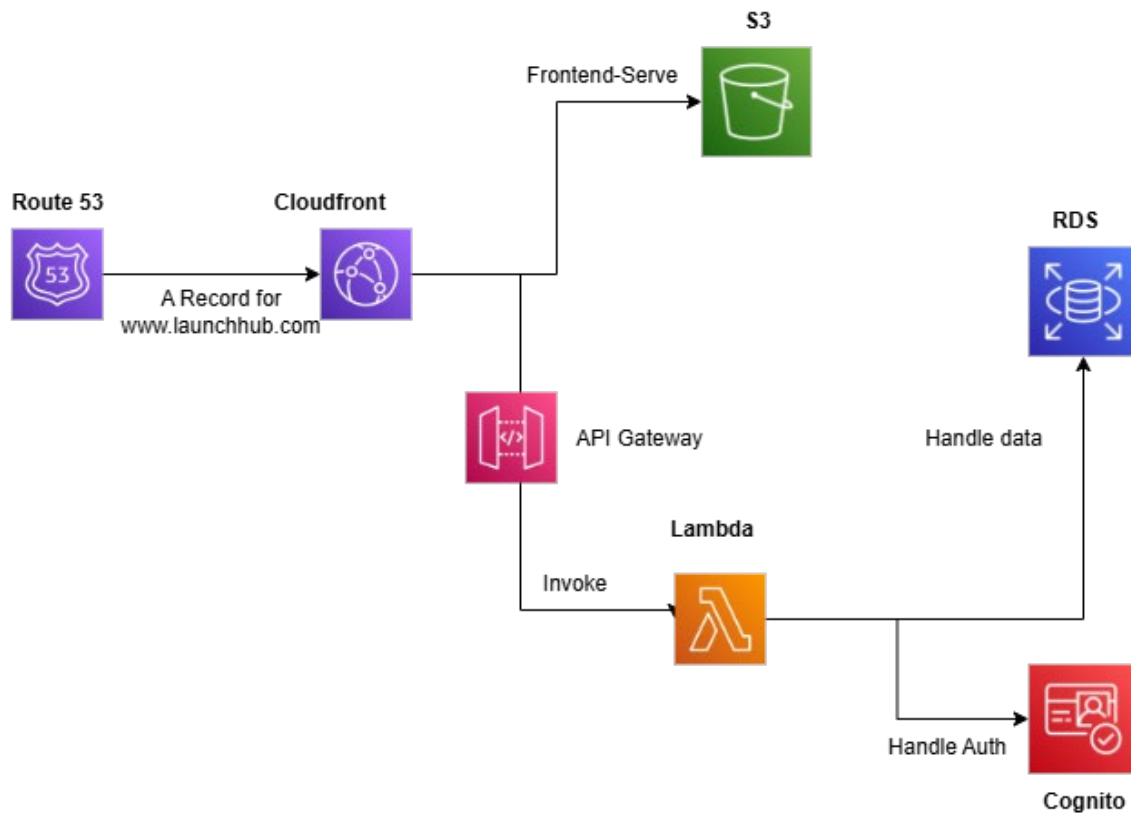
3.2 การวิเคราะห์และออกแบบระบบคลาวด์แอปพลิเคชัน

3.2.1 Use Case Diagram

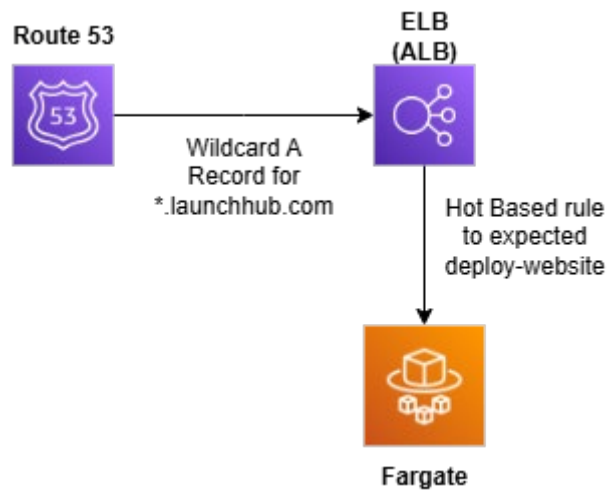


รูปภาพที่ 10 Use Case Diagram

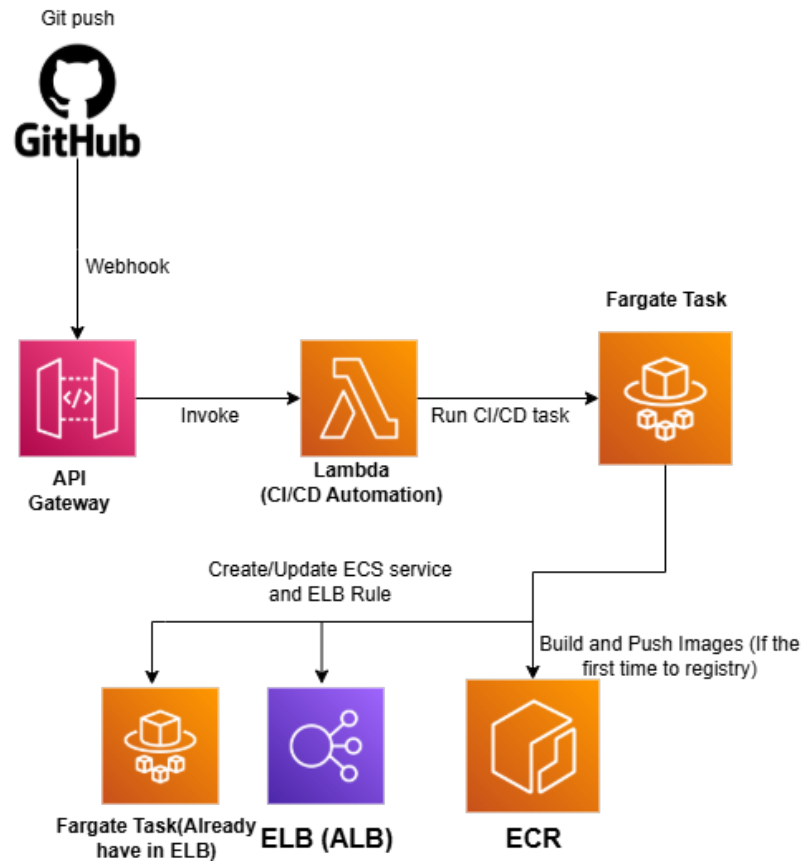
3.3 หลักการทำงานของระบบ



รูปภาพที่ 11 แผนภาพแสดงขั้นตอนการเข้าถึงหน้าแพลตฟอร์มหลัก



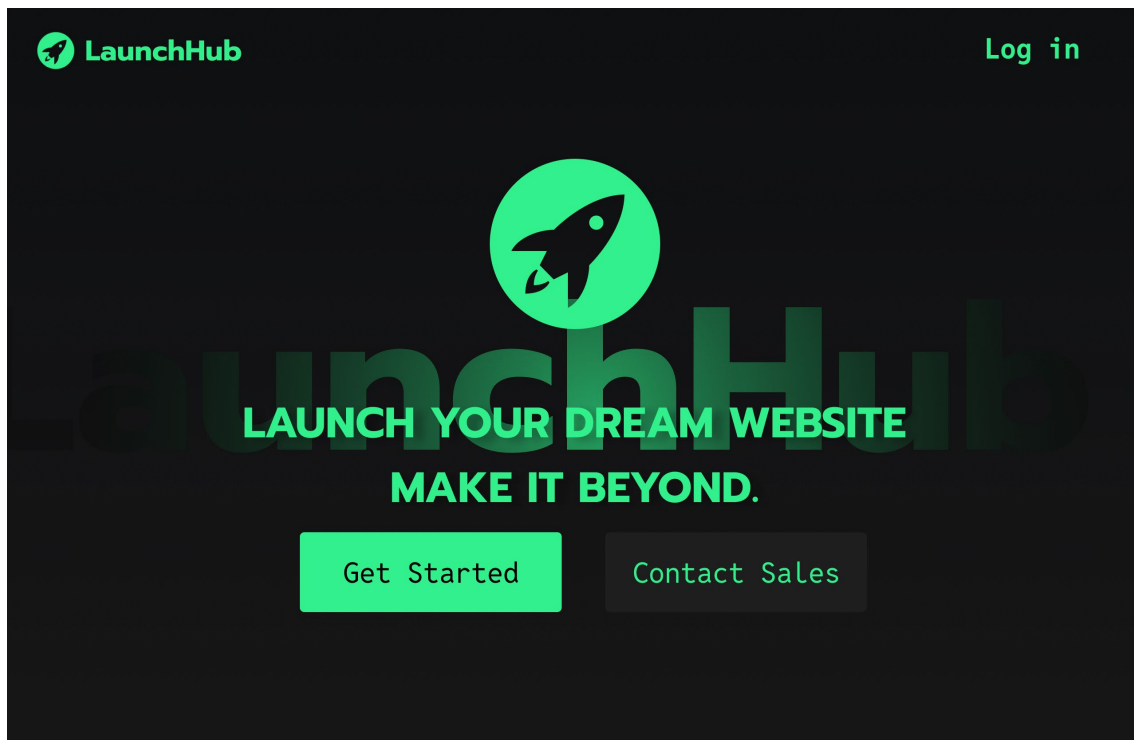
รูปภาพที่ 12 แผนภาพแสดงขั้นตอนการทำงานเมื่อผู้ใช้เข้ามา deploy



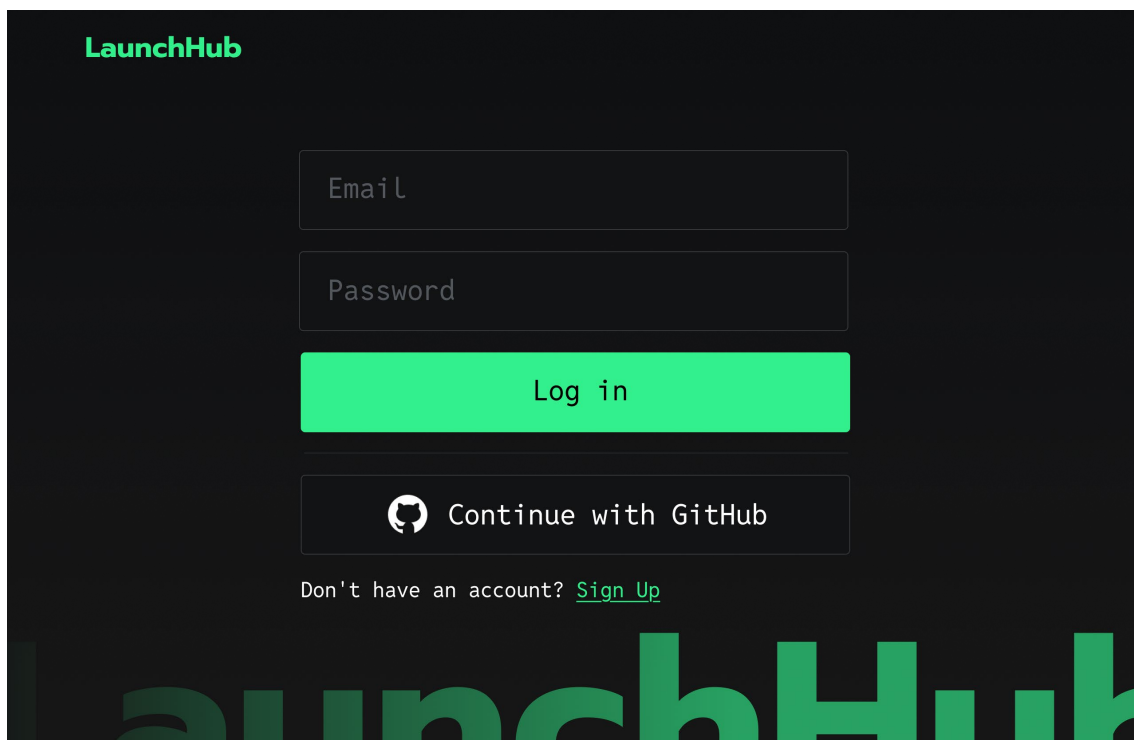
รูปภาพที่ 13 ระบบ CI/CD ของแพลตฟอร์ม

3.4 Mockup UI

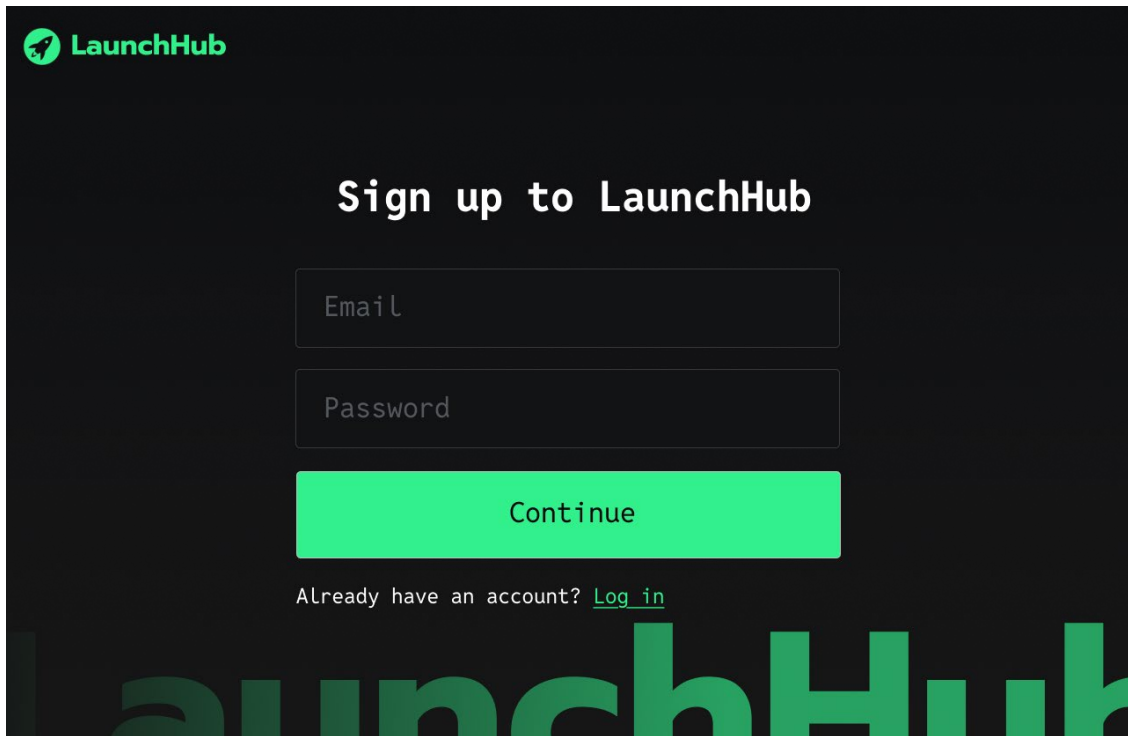
หน้าหลัก



หน้า Login



หน้า Sign up



LaunchHub

Sign up to LaunchHub

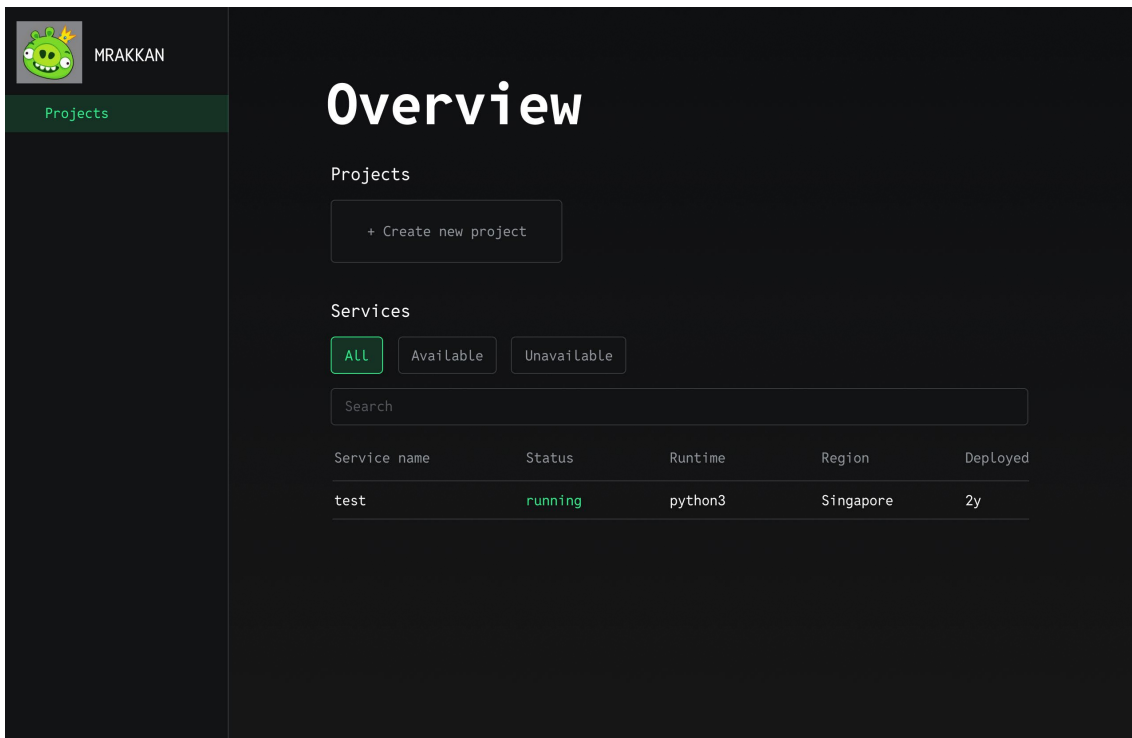
Email

Password

Continue

Already have an account? [Log in](#)

หน้า Dashboard



MRACKAN

Projects

Overview

Projects

+ Create new project

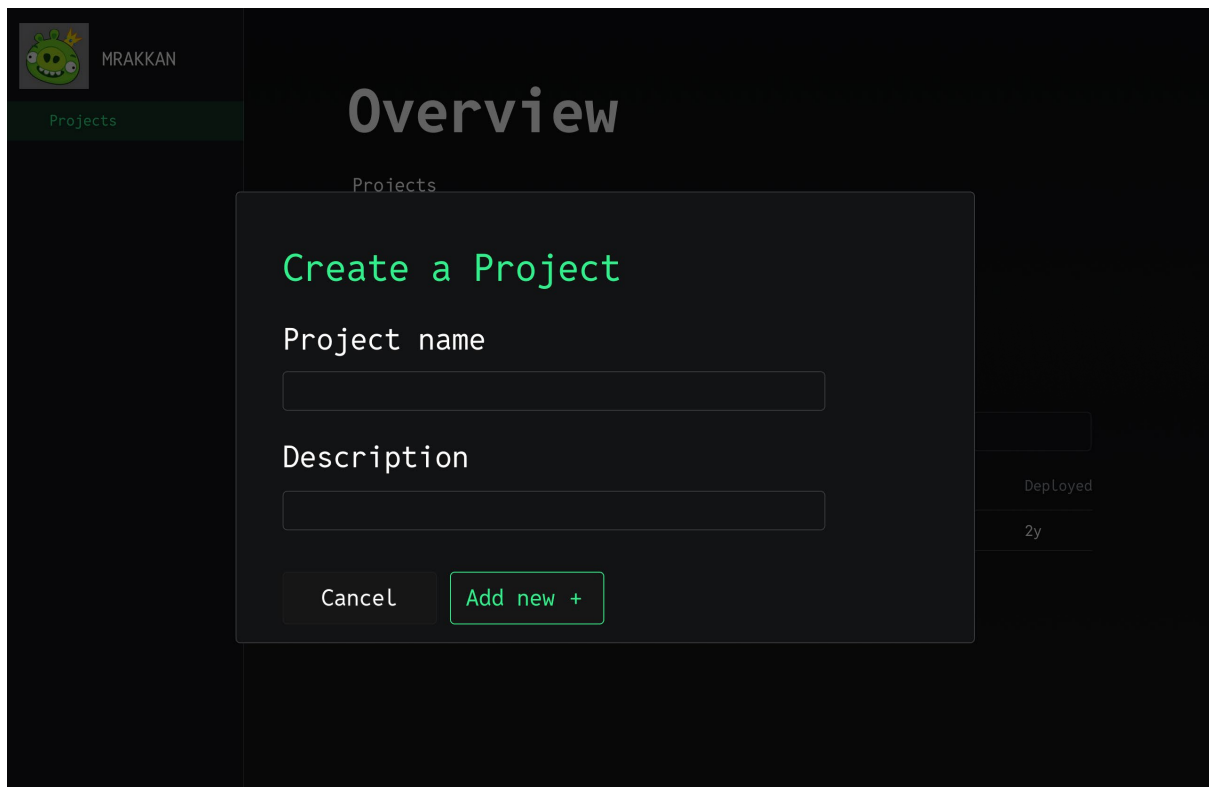
Services

ALL Available Unavailable

Search

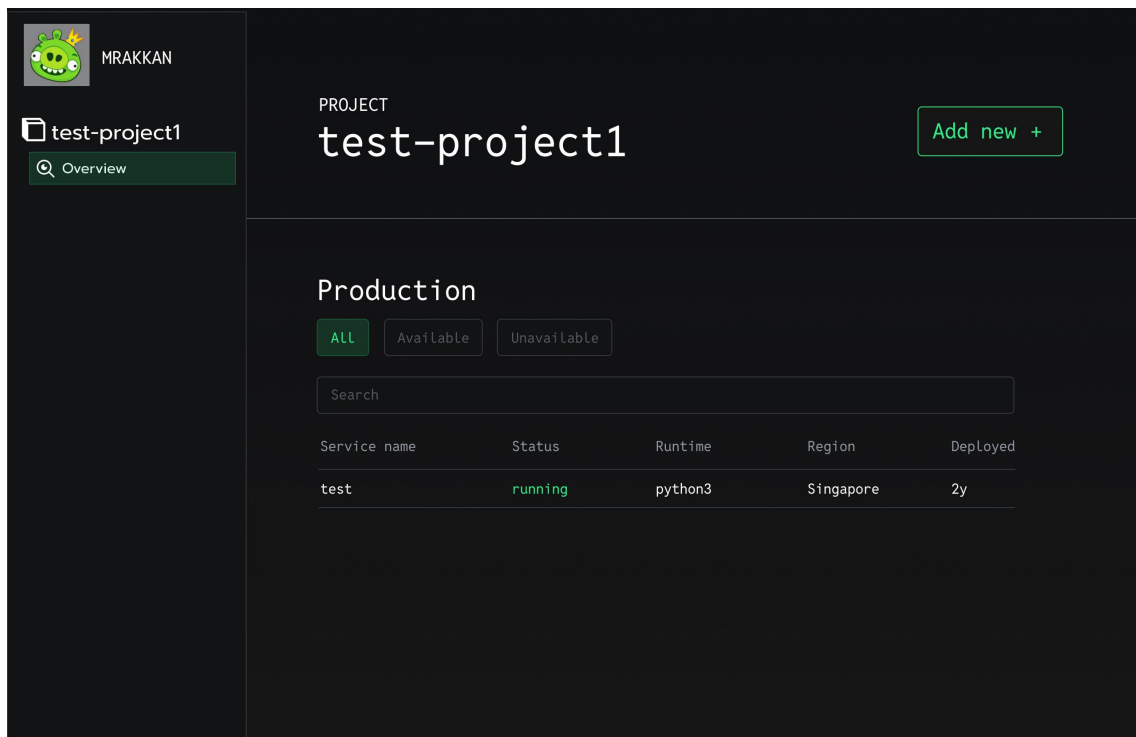
Service name	Status	Runtime	Region	Deployed
test	running	python3	Singapore	2y

หน้าสร้าง Project



The screenshot shows the 'Create a Project' dialog box in the MRAKKAN dashboard. The dialog is centered on a dark background. It has a title 'Create a Project' in green. Below the title are two input fields: 'Project name' and 'Description'. At the bottom of the dialog are two buttons: 'Cancel' and 'Add new +' (highlighted in green). In the background, the 'Overview' page is visible, showing a sidebar with 'Projects' selected and a main area with a table of projects. One project is visible with the status 'Deployed' and a duration of '2y'.


หน้ารายละเอียดโปรเจคที่เคยสร้าง



The screenshot shows the 'test-project1' details page in the MRAKKAN dashboard. The page has a dark background. The sidebar on the left shows 'test-project1' selected, with 'Overview' as a sub-option. The main area has a header 'PROJECT test-project1' with an 'Add new +' button. Below the header is a 'Production' section with three tabs: 'ALL' (selected), 'Available', and 'Unavailable'. There is a search bar below the tabs. A table lists the production services.

Service name	Status	Runtime	Region	Deployed
test	running	python3	Singapore	2y

หน้าสร้าง Web Service ใหม่

 MRAKKAN

test-project1

Overview

New Web Service

Source code


Name

Project:

Language

Branch

root

 MRAKKAN

test-project1

Overview

start-command

build-command

Region


Instance Type

Free Plan

Free Plan

Deploy

หน้า Web Service Detail

 MRAKKAN

test-project1

Overview

WEB-SERVICE

test

Python3

Service ID: asdwdasdwdseqr1234qsd

<https://test.launchhub.com>

Deploy

Filter events