

รายงานการปฏิบัติงานสหกิจศึกษา

เรื่อง

รายงานสหกิจศึกษาตำแหน่ง DevOps Engineer ที่ SCB TechX

ณ บริษัท เอสซีบี เทคโนโลยี จำกัด
19 อาคารไทยพาณิชย์ ปาร์ค พลาซ่า เวสท์ บี ชั้นที่ 21
ถนนรัชดาภิเษก แขวงจตุจักร เขตจตุจักร กรุงเทพมหานคร 10900

โดย

ณัฐพงษ์ เพพพิทักษ์ รหัส 640610634

รายงานนี้เป็นส่วนหนึ่งของกระบวนวิชา 261495 สหกิจศึกษา
ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่
ปีการศึกษา 2567

Cooperative Education Report

Cooperative Education Report for the DevOps Engineer at SCB TechX

**19 SCB Park Plaza West B, 21st Floor,
Ratchadapisek Road, Chatuchak Sub-district, Chatuchak District,
Bangkok 10900**

Natthaphong Thepphithak 640610634

**This report is part of the course 261495 Cooperative Education
Department of Computer Engineering
Faculty of Engineering, Chiang Mai University
Academic Year 2024**

หัวข้อรายงาน : รายงานสหกิจศึกษาตำแหน่ง DevOps Engineer ที่ SCB TechX
โดย : ณัฐพงษ์ เพพพิทักษ์ รหัส 640610634
ภาควิชา : วิศวกรรมคอมพิวเตอร์
อาจารย์ที่ปรึกษา : รศ.ดร. ปฏิเวช วุฒิสารวัฒนา
ปริญญา : วิศวกรรมศาสตรบัณฑิต
สาขา : วิศวกรรมคอมพิวเตอร์
ปีการศึกษา : 2567

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่ ได้อนุมัติให้รายงานนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต (สาขาวิศวกรรมคอมพิวเตอร์)

..... คณบดีคณะวิศวกรรมศาสตร์
(รองศาสตราจารย์ ดร.ธงชัย พองสมุทร)

..... หัวหน้าภาควิชาวิศวกรรมคอมพิวเตอร์
(ศ.ดร. สันติ พิทักษ์กิจนุกร)

คณะกรรมการสอบรายงาน

..... อาจารย์ที่ปรึกษาสหกิจศึกษา
(รศ.ดร. ปฏิเวช วุฒิสารวัฒนา)

..... พนักงานที่ปรึกษา
(พีรวัตร อุยใจเย็น)
(Senior DevOps Engineer)

..... พนักงานที่ปรึกษา
(ภัทรพล หมวดมนี)
(DevOps Engineer)

หัวข้อรายงาน	: รายงานสหกิจศึกษาตำแหน่ง DevOps Engineer ที่ SCB TechX
โดย	: ณัฐพงษ์ เทพพิทักษ์ รหัส 640610634
ภาควิชา	: วิศวกรรมคอมพิวเตอร์
อาจารย์ที่ปรึกษา	: รศ.ดร. ปฏิเวช วุฒิสารวัฒนา
ปริญญา	: วิศวกรรมศาสตรบัณฑิต
สาขา	: วิศวกรรมคอมพิวเตอร์
ปีการศึกษา	: 2567

บทคัดย่อ

รายงานนี้เป็นนำเสนอการสหกิจของวิศวกรรมศาสตรสาขาวิชาคอมพิวเตอร์ในตำแหน่ง DevOps Engineer ที่ SCB TechX ซึ่งเป็นส่วนหนึ่งของธนาคารไทยพาณิชย์ (SCB) ในระหว่างช่วงเวลาของการทำงาน ได้มีส่วนร่วมในการสนับสนุนทีมพัฒนาซอฟต์แวร์ โดยเฉพาะในด้านการสร้างและการลงทุนทรัพยากร รวมถึงการพัฒนาและบำรุงรักษา Terraform Modules สำหรับโมดูลกลางที่ถูกใช้งานทั่วไปใน SCB TechX และ SCB ซึ่งทั้งหมดนี้เป็นส่วนสำคัญในการเพิ่มประสิทธิภาพและความคล่องตัวให้กับกระบวนการ DevOps ของ บริษัท เพื่อตอบสนองความต้องการของลูกค้าอย่างมีประสิทธิภาพและรวดเร็ว

Project Title : Cooperative Education Report for the DevOps Engineer at SCB TechX
Name : Natthaphong Thepphithak 640610634
Department : Computer Engineering
Project Advisor : Assoc. Prof. Patiwet Wuttisarnwattana, Ph.D.
Degree : Bachelor of Engineering
Program : Computer Engineering
Academic Year : 2024

ABSTRACT

This report presents a cooperative education experience in Computer Engineering for the position of DevOps Engineer at SCB TechX, a subsidiary of Siam Commercial Bank (SCB). During the work period, I participated in supporting the software development team, particularly in the creation and deletion of resources, as well as the development and maintenance of Terraform Modules for central modules used in both SCB TechX and SCB. All of these activities were crucial in enhancing the efficiency and agility of the company's DevOps processes to effectively and rapidly meet customer needs.

กิตติกรรมประกาศ

การที่ข้าพเจ้าได้มาปฏิบัติงานสหกิจศึกษา ณ บริษัท เอสซีบี เทคโนโลยี จำกัด ตั้งแต่วันที่ 4 มิถุนายน 2567 ถึง วันที่ 25 ตุลาคม 2567 ส่งผลให้ข้าพเจ้าได้รับความรู้และประสบการณ์ต่างๆ ที่มีค่ามากmany สำหรับรายงาน วิชาสหกิจศึกษาฉบับนี้ สำเร็จลงได้ด้วยดีจากความร่วมมือและสนับสนุนจากหลายฝ่าย ดังนี้

1. พีรวัตร ออยใจเย็น ตำแหน่ง Senior DevOps Engineer
2. กั้ทรพล หมากมณี ตำแหน่ง DevOps Engineer

และบุคคลท่านอื่นๆ ที่ไม่ได้กล่าวนามทุกท่านที่ได้ให้คำแนะนำช่วยเหลือในการจัดทำรายงาน ข้าพเจ้าicr'ขอ ขอบพระคุณผู้ที่มีส่วนเกี่ยวข้องทุกท่าน ที่มีส่วนร่วมในการให้ข้อมูล เป็นที่ปรึกษาในการทำงานฉบับนี้จน เสร็จสมบูรณ์ ตลอดจนให้การดูแลและให้ความเข้าใจเกี่ยวกับชีวิตของการทำงานจริง ข้าพเจ้าขอขอบคุณ ไว้ ณ ที่นี่

ณัฐพงษ์ เทพพิทักษ์

25 ตุลาคม 2567

สารบัญ

บทคัดย่อ	๑
Abstract	๒
กิตติกรรมประกาศ	๓
สารบัญ	๔
สารบัญตาราง	๕
สารบัญรูป	๖
1 บทนำ	1
1.1 วัตถุประสงค์	1
1.2 ประวัติความเป็นมาของบริษัท	1
1.3 บริการและผลิตภัณฑ์ของบริษัท	1
1.4 ผู้บริหารของบริษัท	2
1.5 งบการเงินและงบกำไรขาดทุน	3
1.5.1 งบการเงิน	3
1.5.2 งบกำไรขาดทุน	4
1.6 หน้าที่ของหน่วยงานที่ได้มาสหกิจ	4
2 รายละเอียดเกี่ยวกับการทำงาน	5
2.1 ปรับความรู้พื้นฐานของการเป็น DevOps	5
2.1.1 Jenkins	6
2.1.2 Terraform	7
2.2 TOR	8
2.3 งานที่ได้รับมอบหมาย	8
2.3.1 งาน Support	8
2.3.2 งาน Develop	10
2.3.3 งาน Prove Of Concept	12
2.4 สวัสดิการที่ได้รับ	13
2.5 วัฒนธรรมองค์กร	13
2.6 กิจกรรมที่นอกเหนือจากการทำงาน	14
3 สรุปผลการปฏิบัติงาน	15
3.1 สรุปผลการศึกษา/การฝึกปฏิบัติงาน	15
3.2 ประโยชน์ที่ได้รับจากการศึกษา/การฝึกปฏิบัติงาน	15
3.3 ข้อเสนอจากบริษัท	15
บรรณานุกรม	16
ก เอกสารสำคัญที่เกี่ยวข้อง	18

สารบัญตาราง

1.1 ตารางแสดงตำแหน่งผู้บริหารของบริษัท 2

สารบัญรูป

1.1	ผู้บริหารและตำแหน่งของบริษัท	2
1.2	งบการเงินย้อนหลังตั้งแต่ก่อตั้งบริษัท	3
1.3	งบกำไรขาดทุนย้อนหลังตั้งแต่ก่อตั้งบริษัท	4

บทที่ 1

บทนำ

1.1 วัตถุประสงค์

- เพื่อเรียนรู้แนวคิดและหลักการทำงานของ DevOps ในสถานการณ์การทำงานจริงซึ่งในองค์กรขนาดใหญ่
- เพื่อศึกษาเรียนรู้และปรับใช้เครื่องมือและเทคโนโลยีใหม่ ๆ ที่เกี่ยวกับ Iac (Infrastructure as Code) และ CI/CD (Continuous Integration / Continuous Deployment)
- เพื่อนำความรู้ที่ได้มาปรับใช้กับโครงการฯ ในปีการศึกษาต่อไปเพื่อเสริมการใช้ Standard ให้กับการทำงานเพื่อให้โครงการมีคุณภาพมากยิ่งขึ้น
- เพื่อฝึกกระบวนการคิดและการออกแบบระบบโครงสร้างพื้นฐานของระบบที่มีความยืดหยุ่นสูงและสามารถใช้ได้กับองค์กรทุกขนาด

1.2 ประวัติความเป็นมาของบริษัท

SCB TechX [1] ก่อตั้งขึ้นจากความร่วมมือระหว่าง SCBX กลุ่มธุรกิจการเงินและเทคโนโลยีชั้นนำของไทย และ Publicis Sapient บริษัทที่ปรึกษาด้านดิจิทัลทรานส์ฟอร์เมชันระดับโลก มีจุดมุ่งหมายเพื่อมอบบริการด้านเทคโนโลยีที่ตอบสนองความต้องการของธุรกิจต่าง ๆ ตั้งแต่การสร้างนวัตกรรมและผลิตภัณฑ์ใหม่ไปจนถึงการนำเทคโนโลยีมาเพิ่มประสิทธิภาพในการดำเนินงาน

บริษัทมีความเชี่ยวชาญในการพัฒนาโซลูชันในระดับองค์กร (Enterprise-grade solutions) ที่ปลอดภัยและรองรับการใช้งานของฐานลูกค้าจำนวนมาก นอกจากนี้ SCB TechX ยังจัดองค์กรในรูปแบบ Startup ถึงแม้จะเป็นองค์กรขนาดใหญ่ เพื่อลดความซ้ำซ้อนในการทำงานและส่งเสริมความคิดริเริ่มใหม่ ๆ ทำให้สามารถพัฒนาโซลูชันให้กับลูกค้าได้อย่างรวดเร็วและมีประสิทธิภาพ

1.3 บริการและผลิตภัณฑ์ของบริษัท

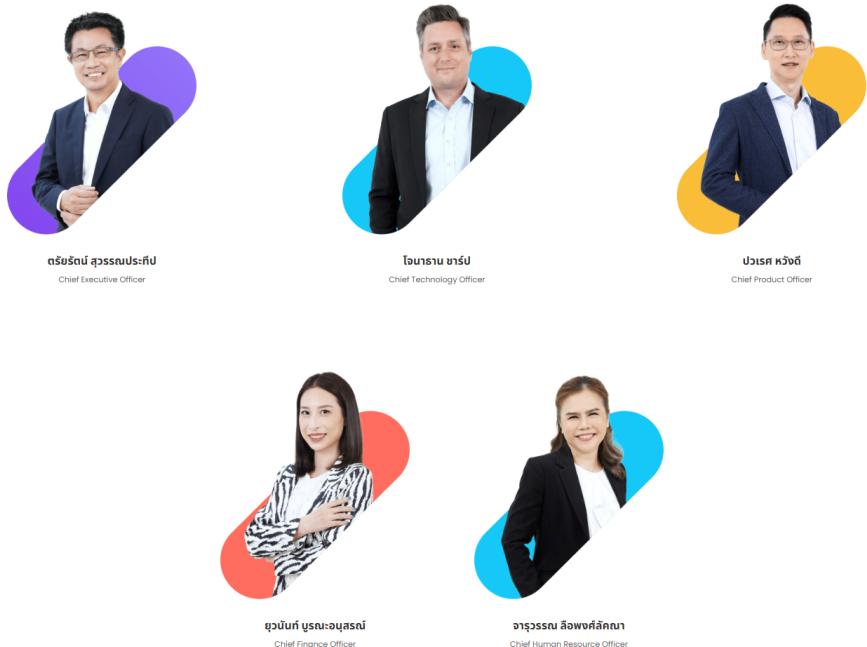
SCB TechX นำเสนอวัตกรรมที่พร้อมใช้งานหลากหลายด้าน [2] ทั้งระบบบัญชีตัวตนแบบดิจิทัลด้วยระบบ KYC [3] ซึ่งทางบริษัทจะเรียกว่า eKYC และแพลตฟอร์มทางการเงินที่หลากหลาย นวัตกรรมเหล่านี้สามารถเชื่อมต่อกับระบบของลูกค้าได้อย่างรวดเร็วและง่ายดาย พร้อมทั้งปรับแต่งตามความต้องการเฉพาะของธุรกิจ ส่งผลให้ลูกค้าของ SCB TechX สามารถเปิดตัวบริการใหม่หรือยกระดับการให้บริการได้อย่างทันท่วงที

นอกจากนี้ SCB TechX ยังให้บริการที่ครอบคลุมด้านการให้คำปรึกษาทางเทคโนโลยี (Technology Consulting), โซลูชันด้านโครงสร้างพื้นฐานและแพลตฟอร์ม (Infrastructure & Platforms), โซลูชันคลาวด์ (Cloud Solutions), แพลตฟอร์ม DevOps as a Service ที่ครบวงจร (xPlatform), การจัดการข้อมูลและความปลอดภัย (Data & Security), และโซลูชันด้านข้อมูลและ AI (TechX Data & AI Solutions) ที่ออกแบบมาเพื่อรองรับและเสริมสร้างศักยภาพให้กับธุรกิจในยุคดิจิทัล

1.4 ผู้บริหารของบริษัท

Name	Position
Trirat Suwanprateeb	Chief Executive Officer
Jonathan Sharp	Chief Technology Officer
Pavarej Hwangdee	Chief Product Officer
Yuwanan Buranaanusorn	Chief Finance Officer
Jaruwan Luepongluckanna	Chief Human Resource Officer

ตารางที่ 1.1: ตารางแสดงตำแหน่งผู้บริหารของบริษัท



รูปที่ 1.1: ผู้บริหารและตำแหน่งของบริษัท

1.5 งบการเงินและงบกำไรขาดทุน

1.5.1 งบการเงิน

Balance Sheet 2021 - 2023

Unit : Baht	2021	2022	2023
Total asset	1,742,154,154.00	2,350,832,199 +34.93%	1,954,277,028 -16.86%
Total liability	1,241,214,554.00	821,341,868 -33.82%	587,006,753 -28.53%
Equity	500,939,600.00	2,350,832,199 +205.32	1,367,270,275 -10.60%

รูปที่ 1.2: งบการเงินย้อนหลังตั้งแต่ก่อตั้งบริษัท

เนื่องจากบริษัท SCB TechX เป็นบริษัทในเครือของ SCB ดังนั้นจึงจะเห็นว่าทรัพย์สินของบริษัทนั้น มีมูลค่าสูงหลักพันล้านบาท ตั้งแต่เริ่มก่อตั้งบริษัท ทั้งนี้หากมองลึกลงไปในปีแรกก็จะเห็นว่าทรัพย์สินหลักพันล้านนั้นเกิดมาจากการกู้ยืมเป็นส่วนใหญ่ ในส่วนของผู้ถือหุ้นนั้นยังไม่มากในช่วงแรก หากอ้างอิงข้อมูลจากในรูปที่ 1.2 แล้วนั้นจะเห็นได้ว่าในปีที่สองมูลค่าทรัพย์สินรวมเพิ่มสูงขึ้นอย่างก้าวกระโดด ทั้งนี้ด้วยการเติบโตที่อาจจะเริ่วเกินไปทำให้ในปีต่อมาค่อยๆปรับลดลง แต่มูลค่าหนี้สินจะเห็นได้ว่าค่อยๆลดลงเรื่อยๆ แสดงให้เห็นถึงการพัฒนาของบริษัทได้เป็นอย่างดี

1.5.2 งบกำไรขาดทุน

Income Statement 2021 - 2023

Unit : Baht	2021	2022	2023
Total income	1,595,662,931.00	3,169,316,485.00 +98.62%	2,288,582,370.00 -27.78%
Net profit	349,939,600.00	671,775,791.00 +91.96%	245,779,944.00 -63.41%

รูปที่ 1.3: งบกำไรขาดทุนตั้งแต่ก่อตั้งบริษัท

จากข้อมูลใน รูปที่ 1.3 จะเห็นได้ว่าบริษัท SCB TechX มีกำไรตั้งแต่ปีที่สอง และกำไรเพิ่มขึ้นอย่างมีนัยสำคัญ ถึงแม้ในปี 2023 กำไรลดลงจากปี 2022 ถึง ร้อยละ 63 เป็นผลของการพัฒนาของบริษัทที่เร็วขึ้น การขยายขนาดของบริษัท ทำให้ต้องเพิ่มการจ้างงานมากขึ้นซึ่งเป็นเหตุที่ทำให้รายได้ลดลงจากปีก่อนหน้า แต่ถ้าหากเปรียบเทียบกับปีแรกก็ยังถือว่ามีกำไรอยู่ในระดับที่ดีในทุก ๆ ปี

1.6 หน้าที่ของหน่วยงานที่ได้มาสหกิจ

ในตำแหน่ง DevOps Engineer ที่ SCB TechX หน้าที่หลักของผู้มีคือการทำงานเป็นตัวกลางระหว่างทีมพัฒนา (Development Team) และทีมปฏิบัติการ (Operation Team) โดยมุ่งเน้นไปที่การพัฒนาและจัดหาเครื่องมือเพื่อช่วยในการบริหารจัดการระบบ รวมถึงสนับสนุนการทำงานของทีมพัฒนา นอกจากนี้ ยังมีบทบาทสำคัญในการดูแลและปรับปรุงกระบวนการ CI/CD เพื่อให้การส่งมอบระบบไปยังผู้ใช้ (Delivery) เป็นไปอย่างราบรื่นและรวดเร็ว

ในด้านอื่น ๆ ผู้มีหน้าที่ในการเตรียมความพร้อมของ Pipeline สำหรับการ Deploy และสร้างเครื่องมืออัตโนมัติ (Automation Tools) เพื่อช่วยลดเวลาทำงานของทุกทีมในองค์กร อีกทั้งยังทำหน้าที่ประสานงานกับเครือข่าย (Network) และความปลอดภัย (Security) เพื่อสนับสนุนทีมอื่น ๆ ภายใต้บริษัทตามขอบเขตงานที่ได้รับมอบหมาย

บทที่ 2

รายละเอียดเกี่ยวกับการทำงาน

2.1 ปรับความรู้พื้นฐานของการเป็น DevOps

แนวทางในการเริ่มต้นทำงานในสายงาน DevOps จำเป็นต้องมีการศึกษาและปรับพื้นฐานความรู้ที่สำคัญเพื่อให้แน่ใจว่าพร้อมสำหรับการทำงานจริง เนื่องจาก DevOps เป็นสายงานที่มีความใหม่และมีการพัฒนาอย่างต่อเนื่องในวงการซอฟต์แวร์ โดยหัวข้อที่ได้รับมอบหมายให้ศึกษาเพื่อเตรียมความพร้อมจะประกอบด้วย

- **Docker:** เครื่องมือสำหรับการทำ Containerization เพื่อเพิ่มประสิทธิภาพในการพัฒนาและการส่งมอบซอฟต์แวร์
- **Kubernetes:** ระบบสำหรับการทำ Orchestration และจัดการคอนเทนเนอร์ที่ทำงานในสเกลใหญ่
- **Jenkins:** เครื่องมือสำหรับการทำ Continuous Integration/Continuous Deployment (CI/CD) เพื่อเพิ่มความรวดเร็วและลดข้อผิดพลาดในการพัฒนาซอฟต์แวร์
- **Terraform & IaC:** เครื่องมือสำหรับการทำ Infrastructure as Code (IaC) เพื่อจัดการและปรับแต่งโครงสร้างพื้นฐานด้วยโค้ด
- **Monitoring Tools:** เครื่องมือที่ใช้ในการตรวจสอบและติดตามการทำงานของระบบอย่างมีประสิทธิภาพ
- **ELK Stack:** ระบบสำหรับการจัดการและวิเคราะห์ข้อมูล Logging เพื่อช่วยในการตรวจสอบและวิเคราะห์ปัญหาในระบบ

ทั้งนี้การศึกษาหัวข้อเหล่านี้มีระยะเวลาประมาณ 2-4 สัปดาห์ และในท้ายที่สุดจะต้องมีการนำเสนอสิ่งที่ได้เรียนรู้ ให้กับพี่ ๆ ในทีมได้ฟังและประเมินว่าพร้อมที่จะทำงานจริงหรือไม่ อย่างไรก็ตามรายละเอียดในหัวข้ออยู่ต่าง ๆ หลังจากนี้จะเป็นการนำเสนอสิ่งที่ได้เรียนรู้และได้นำมาประยุกต์ใช้ในการทำงานจริง ส่วนหัวข้อนอกเหนือจากที่จะกล่าวถึงก็สำคัญไม่น้อยเช่นกันแต่จะขอนำเสนอ Documentation ที่ได้ทำสรุปการเรียนรู้มาแล้วนั้นในส่วนภาคผนวก

2.1.1 Jenkins

Jenkins คือ Software (Tool) ตัวนึงที่สามารถใช้ทำ CI/CD [4] เพื่อที่จะสามารถทำให้งานของ Dev & Dev ถูกพัฒนาและส่งมอบให้กับลูกค้าได้เร็วขึ้น โดยที่ Jenkins จะช่วยในการทำงานของการ Build, Test, Deploy โดยที่ไม่ต้องอาศัยบุคคลในการทำงานในส่วนนี้ซ้ำ ๆ ทั้งนี้ในบริบทของการใช้งาน Jenkins ของ DevOps ในงานจริงอาจแตกต่างออกไปดังนั้นในหัวข้อนี้จะเป็นการสรุปว่า DevOps ใช้ Jenkins ทำอะไรบ้าง และ Jenkins ช่วยในการทำงานของ DevOps อย่างไร

Jenkins จะมีหนึ่งความสามารถที่เรียกว่า Jenkins Pipeline ซึ่ง DevOps เองก็ทำความสามารถตรงนี้มาใช้ในการทำงาน ไม่ว่าจะเป็นการ Provisioning & Destroying resources, Deploying โดยการเขียน Script ในรูปแบบของ Jenkinsfile ขึ้นมาเพื่อทำงานตามที่ต้องการ เช่น Pipeline สำหรับ Deploy microservice Pipeline นี้ก็จะทำงานสำหรับการ Deploy โดยเฉพาะ โดยที่เมื่อมีการสั่งให้ทำงาน Jenkins จะทำงานตามที่เขียนไว้ใน Jenkinsfile หลังจาก Jenkins ทำงานเสร็จสิ้นก็เป็นอันว่า microservice นั้น Deploy สำเร็จ

จะเห็นได้ว่าเมื่อเรามี Jenkins เข้ามาช่วยทำงานที่เป็น Routine ที่เราต้องอะไรเดิม ๆ ซ้ำ ๆ ทำให้เราสามารถลดเวลาในการทำงานลงได้และยังช่วยให้ Productivity ของทีมทำงานเพิ่มขึ้นอีกด้วย ทั้งจากที่กล่าวมาข้างต้นว่า DevOps ใช้ Jenkinsในการ Provisioning resource ต่าง ๆ ด้วยนั้นหลักการก็จะคล้าย ๆ กับการ Deploy เพียงแต่ Script ที่ใช้สั่งการนั้นจะเปลี่ยนแปลงให้เป็น Script สำหรับ Provisioning resource แทน ทั้งนี้ทำให้เราสามารถทำงานได้เร็วขึ้นและลดความผิดพลาดในการทำงานลงได้

```
pipeline {
    agent any
    stages {
        stage('Build') {
            steps {
                echo 'Building...'
            }
        }
        stage('Deploy') {
            steps {
                echo 'Deploying...'
            }
        }
    }
}
```

2.1.2 Terraform

Terraform คือเครื่องมือหนึ่งที่ใช้ในการสร้างและจัดการโครงสร้างพื้นฐาน (Infrastructure) ด้วยแนวคิด *Infrastructure as Code* (IaC) โดยที่ Terraform นั้นมีความสามารถในการ Provision และ Manage resource ต่าง ๆ ใน cloud provider ได้หลายแห่ง เช่น AWS, Azure, และ GCP รวมถึง on-premises environments อื่น ๆ ซึ่งช่วยให้การจัดการโครงสร้างพื้นฐานเป็นไปอย่างมีประสิทธิภาพและสามารถควบคุมได้ง่ายขึ้น

การใช้งาน Terraform ในบริบทของ DevOps

ในบริบทของการใช้งานจริง Terraform มักจะถูกนำมาใช้โดยทีม DevOps เพื่อจัดการและ Provision resource ต่าง ๆ เช่น เซิร์ฟเวอร์, ฐานข้อมูล, Network configurations และอื่น ๆ รวมถึงสามารถใช้ Terraform ในการจัดการ *infrastructure lifecycle* ทั้งหมดไม่ว่าจะเป็นการสร้าง, เปลี่ยนแปลง, หรือการลบ resources ได้อย่างง่ายดายและเป็นอัตโนมัติ

Terraform มีโครงสร้างที่เรียบง่ายในการใช้งาน ซึ่งประกอบด้วยการเขียน Configuration files (มักเป็นไฟล์ที่มีนามสกุล .tf) เพื่อกำหนด resource ที่ต้องการ ซึ่งไฟล์เหล่านี้สามารถจัดเก็บไว้ในระบบ version control (เช่น Git) เพื่อให้สามารถทำการ versioning และการทำงานร่วมกันในทีมได้

นอกจากนี้ Terraform ยังมีความสามารถในการ *plan* และ *preview* การเปลี่ยนแปลงที่จะเกิดขึ้นกับ infrastructure ก่อนที่จะทำการ *apply* จริง ทำให้สามารถลดความเสี่ยงจากการปรับเปลี่ยนโครงสร้างพื้นฐานที่อาจทำให้เกิดปัญหาได้

ข้อสรุปของการใช้งาน Terraform โดย DevOps

- **Provisioning Resources:** ใช้ในการสร้าง resource ต่าง ๆ ใน cloud หรือ on-premises
- **Infrastructure as Code (IaC):** ทำให้การจัดการ infrastructure มีความคล่องตัวและควบคุมได้ง่ายขึ้น
- **Automation:** ทำให้การจัดการ resource เป็นอัตโนมัติ ลดงาน manual
- **Version Control:** Configuration files สามารถจัดเก็บและ versioning ได้ ทำให้การทำงานร่วมกันในทีมง่ายขึ้น
- **Safety:** สามารถ *plan* และ *preview* การเปลี่ยนแปลงก่อน *apply* จริงเพื่อลดความเสี่ยง

2.2 TOR

เนื่องจากการมาสหกิจศึกษาจำเป็นต้องมีการประเมินที่เข้มงวด ดังนั้นจึงได้จัดทำ TOR ขึ้นเพื่อเป็นมาตรฐาน และข้อตกลงในการทำงานและประเมินผลกับบริษัทและอาจารย์ว่างานควรจะทำได้ดังนี้

- ทำงานในลักษณะของ Task based จำนวน 30 tasks เป็นขั้นต่ำ เนื่องจากทีม DevOps ของ SCB Techx ไม่ได้ใช้การทำงานแบบ Agile จึงทำให้ลักษณะการจะเป็น Kanban กล่าวคือ เมื่อมีงานใหม่เข้ามาจะสามารถทำได้ทันทีโดยไม่ต้องมีการ Sprint planning ก่อน ดังนั้นของการทำงานจึงต้องอาศัยความรอบคอบและความรวดเร็วในการทำงาน เพื่อตอบสนองความต้องการของทีม Dev และ ลูกค้าให้ได้มากที่สุด

ดังนั้นรายหัวข้อต่อไปจะเป็นการนำเสนอ Task งานที่ได้รับมอบหมายให้ทำทั้งหมด เพื่อแสดงให้เห็นว่าสามารถบรรลุเป้าหมายของ TOR ได้

2.3 งานที่ได้รับมอบหมาย

สืบเนื่องจากการนี้ได้รับมอบหมายนั้นจะเป็นในลักษณะ Task งาน ดังนั้นใน Section นี้จะนำเสนอ Task งานที่ได้ทั้งหมดพร้อมทั้งรายละเอียดของแต่ละ Task งานที่ได้รับมอบหมาย โดยจะแบ่งออกได้เป็น 3 กลุ่มใหญ่ ๆ คือ Task งานที่เป็น Support, Task งานที่เป็น Develop และงาน Prove of Concept

2.3.1 งาน Support

- XDO-7387: CBS-SunCBS | Destroy Dev02 resources**
งานนี้จะต้อง Destroy Resource ที่อยู่ใน Dev02 ของ Sun-CBS Project เนื่องจากเกิดข้อผิดพลาดในการวางแผนการใช้งาน Resource ทำให้สินเปลือง Cost โดยไม่จำเป็น
- XDO-7389: CBS-SunCBS Request to add role to managed identity**
งานนี้สืบเนื่องมาจาก Dev Request ให้ DevOps เพิ่ม Role ในการ login เข้าใช้งานให้กับ Managed Identity เนื่องจาก Managed Identity ไม่มีสิทธิการใช้งานในส่วนที่ Dev ต้องการจึงต้องทำการเพิ่มให้ภายหลัง
- XDO-7394: CBS-MCR | Provisioning Storage Account for Dev01 and QA01**
งานนี้จะต้องทำการ Provision Storage Account ให้กับ Dev01 และ QA01 ของ MCR Project ตาม Planing ที่ได้กำหนดไว้
- XDO-7421 | XDO-7490: CBS-SunCBS | Request to set parameter on MySQL to Off on IaC**
งานนี้จะต้องหารือการในการปิด Parameter บางอย่างใน MySQL ซึ่งมีข้อจำกัดคือจะต้องทำผ่าน Terraform เท่านั้น
- XDO-7429: Require provision the existing azure manage identity to version v.1.2.1**
งานนี้เป็นงานที่ทีม Platfrom ได้เปิดการด้วยแบบตัวเองเข้ามาทีม DevOps เพิ่มให้อัพเดท Module Terraform สำหรับการสร้าง Azure Manage Identity เป็น version 1.2.1 (ล่าสุด ณ ขณะนี้) เพื่อให้สอดคล้องกับ Pipeline ใหม่ของ Platform Team ที่จะถูกนำมาใช้งาน

- **XDO-7453:** CBS MCR | Request to grant db_datareader access to the Azure Automation Account user-managed identity

งานนี้เป็นการเพิ่ม Role ให้กับ Azure Automation Account ที่ใช้งานใน MCR Project เพื่อให้สามารถอ่านข้อมูลจาก Database ได้ ซึ่ง Role นั้นก็อาศัยการใช้ Managed Identity ในการ Login นั้นหมายความว่า Managed Identity อันนี้จะต้องผ่านการเพิ่ม Login Role มาแล้วครั้งๆ กับงาน XDO-7389

- **XDO-7604:** CBS-SunCBS | Provisioning Infrastructure resource for Migration-Dev1 (DEV03)

งานที่เป็นการ Provision Infrastructure ให้กับ Environment ใหม่ที่ชื่อว่า Dev03 ซึ่งหมายความว่า Resource ทุกอย่างไม่เคยมีมาก่อนและต้องทำการสร้างขึ้นมาใหม่ทั้งหมดประกอบไปด้วย

- Storage Account
- Managed Identity
- Redis Cache
- Mysql flexible server
- CosmosDB
- Key vault

ซึ่ง Spec ของแต่ละรายการจะถูกกำหนดไว้ใน Jira Card นั้น ๆ และการสร้าง Resource ทุกอย่างนี้จะต้องผ่านการใช้ Terraform และจะมีลำดับการสร้างเพื่อไม่ให้เกิด Conflict ในการสร้าง Resource ดังนั้นเองงานนี้จึงต้องมีความเข้าใจและรอบคอบในการทำงาน

- **XDO-7686 | XDO-7754:** CBS-SunCBS | Asssement of changing Storage account ADLS Gen2

งานนี้คือการเปลี่ยน Version ของ Storage Account จาก General ไปเป็น ADLS Gen2 ซึ่งเป็นการเปลี่ยนแปลงที่สำคัญเพื่อเพิ่มประสิทธิภาพในการใช้งานของ Storage Account โดยเฉพาะในการใช้งานในส่วนของ Data Lake ซึ่งการเปลี่ยนแปลง Version นั้นจะต้องเกิดการลบและสร้างขึ้นมาใหม่ทั้งหมดดังนั้นก่อนเริ่มดำเนินการจะต้องมีการ Approve จาก Dev ต้นทางซึ่งเป็นเจ้าของข้อมูลที่อยู่ใน Storage Account นั้น และงานลักษณะนี้จะต้องทำในหลายๆ Environment เช่นกัน

- **XDO-7844:** CBS-MCR | Provisioning resource to support POC SSIS

สืบเนื่องมาจากทางทีม Dev มีเป้าหมายที่จะเปลี่ยนแปลงการ Sync ของมูลของ Database ซึ่งมีอยู่จำนวนมากทั้ง On Cloud และ On Premise ซึ่งก่อนหน้านี้นั้นใช้ Db Sync service ที่ Azure ให้มาแต่ด้วยที่มี Database หลายตัวจึงทำให้เกิดปัญหาในการใช้งาน จึงจำเป็นต้องหา Solution ใหม่ๆ โดยที่การใช้ SSIS ก็เป็นอีกหนึ่งวิธี ดังนั้นเอง DevOps จึงต้อง Support การสร้าง SSIS ขึ้นมาโดยที่การสร้างนั้นต้องคำนึงถึงเรื่อง Network และ Security ด้วยเนื่องจาก Database แต่ละตัวนั้นอยู่ใน Network ที่แตกต่างกันและเป็น Private Network ด้วย

- **XDO-7861 | XDO-7862:** Destroy unused resource

งานนี้เป็นงานที่ต้องทำการ Destroy Resource ที่ไม่ได้ใช้งานอยู่ใน Environment ของ Project ต่าง ๆ ซึ่งเป็นเรื่องที่สำคัญในการลด Cost ของ Project และเป็นเรื่องที่ต้องทำอย่างสม่ำเสมอเพื่อไม่ให้เกิดปัญหาในการใช้งานของ Project ในอนาคต ซึ่งระหว่างการทำงานก็มีปัญหาเกิดขึ้น เรื่องจาก Terraform Provider ที่เราใช้นั้นมีการอัพเดท version ซึ่งทำให้ Module ที่เคยเขียนไว้ใน version ที่เก่ากว่าใช้งานไม่ได้ทำให้จะต้องมีการทำงานในการ Upgrade Module ที่เก่าให้ใช้งานได้กับ version ใหม่ก่อนทำการ Destroy Resource

- **XDO-7909:** CBS-SunCBS | Request to add more endpoint of DFS to SunCBS storage account

สืบเนื่องจาก card XDO-7686 | XDO-7754 หลักจากที่ได้มีการอัพเกรด Storage Account ให้เป็น ADLS Gen2 แล้ว ดังนั้นจึงทำให้ความสามารถของ Storage Account นั้นเพิ่มขึ้นมาอีก นั้นจึงทำให้ private endpoint ชนิดเดิมที่มีอยู่นั้นไม่เพียงพอในการใช้งาน จึงต้องทำการเพิ่ม Endpoint ชนิด DFS เข้าไปเพื่อทำให้ใช้งาน ADLS Gen2 ได้เต็มประสิทธิภาพมากยิ่งขึ้น

- **XDO-8124 | XDO-8125:** CBS-MCR | Request to provisioning Azure Resource to support SSIS on DEV01 and QA01

สืบเนื่องจาก XDO-7844 ที่ได้มีการขอให้สร้าง SSIS ขึ้นมาเพื่อ POC การทำ Data Sync ระหว่าง Database on Cloud กับ On Premise หลังจากทีม Dev ได้ลองใช้งานแล้วพบว่าสามารถใช้งานได้ดี จึงต้องการที่จะทำมาใช้งานจริง ๆ ดังนั้นจึงได้มีการขอให้ทาง DevOps Provision SSIS ให้กับ Environment ที่จะใช้งานจริง ๆ คือ DEV01 และ QA01 งานนี้มองในรายละเอียดลงไปจะค่อนข้างยากเนื่องจากก่อนหน้าที่เราเคยสร้าง SSIS ขึ้นมาบ้านนี้เป็นการใช้ Username Password ในการ Login แต่ครั้งนี้เนื่องด้วยมาตรฐานความปลอดภัยที่สูงขึ้นจึงต้องใช้ Managed Identity ในการ Login แทน จึงทำให้ต้องมีการทดลองและทดสอบความเป็นไปได้ในการทำ ซึ่งนั้นก็เป็นหน้าที่ของผู้เชื่อมที่คนที่ทำวิธีการใช้งาน Managed Identity ในการ Login และทำให้ SSIS สามารถใช้งานได้

2.3.2 งาน Develop

- **XDO-7483 | XDO-7484:** [ADF] Convert module from IAC next gen to xplatform multicloud

งานเป็นงานที่คล้ายๆการ Restructure IaC ของการสร้าง ADF ในโปรเจค Payment Domain ใหม่ ทั้งหมดเนื่องจาก Version ปัจจุบันนี้ไม่ได้แยก Module ของแต่ละ Component ของ ADF ออกจากกันซึ่งประกอบด้วย Datafactory, Linked services, Trigger และ Pipeline ส่งผลให้มีความต้องการในการแก้ไขบางอย่างจะต้องพับเจอกับ Code ประมาณ 10,000 บรรทัด ซึ่งเป็นเรื่องที่ไม่ควรจะเกิดขึ้น ทางทีม DevOps จึงเห็นว่าเรื่องนี้ควรจะแก้ไข ซึ่งงานนี้ผมได้ทำกับพี่เลี้ยงอีกคนหนึ่ง โดยผมรับหน้าที่ในการ Restructure ส่วนของ Linked Services และ Trigger ซึ่งเป็นส่วนที่มีความซับซ้อนมากที่สุด โดยการ Restructure ครั้งนี้อ้างอิง Standard ของ Module จากฝ่าย SCB ที่ทาง DevOps ของ Techx เป็นผู้ Design ขึ้นมา ทั้งหมดใช้เวลาประมาณ 2 สัปดาห์ในการ Implement และ Test

- **XDO-7561 | XDO-7737 | XDO-7742:** CBS-DAP | Enhance Azure Data factory linked services modules

งานนี้เป็นการเพิ่มความสร้างมาของ SFTP Linked Services ใน ADF Modole ของ SCB ให้สามารถไปดึงรหัสผ่านของ SFTP server จาก Azure keyvault ได้ โดยที่ไม่ต้องใส่รหัสผ่านลงใน Config ของ ADF โดยตรง ซึ่งเป็นเรื่องที่สำคัญในการเพิ่มความปลอดภัยในการใช้งานของ ADF และเป็นการเพิ่มความสะดวกในการใช้งานด้วย

หลังจาก Enhance ฝั่ง Module หลักไปแล้วก็ต้องไปแก้ไข Catalog ที่เรียกใช้ Module ให้รองรับการใช้งาน Feature นี้ได้ เช่นกันโดย Catalog จะเป็นฝั่ง Techx ที่รับหน้าที่ดูแลให้

ทั้งนี้การที่เรา Develop Module เนื่องเพิ่ม Feature ใหม่เข้ามาจะต้องมีการทำ Documentation ของวิธีการใช้ Module นั้นขึ้นมางานนี้ก็เช่นกันผิดต้องทำ Documentation เพื่อสอนการใช้งาน SFTP Linked Services ที่ผุดได้พัฒนาขึ้นมา ซึ่ง Document จะต้องเป็นภาษาอังกฤษ เนื่องจากมีที่มีพัฒนาที่เป็นต่างชาติอยู่ใน Project นี้ด้วย

- **XDO-7633:** [PYMD] Develop check appconfig for Azure Databrick

งานนี้เป็นความต้องการจาก Dev ที่มีปัญหา ก่อนการ Deploy Production ว่าต้องการ Jenkins Pipeline ซักตัวหนึ่งที่ใช้ในการ Check Config ของ Microservcice ใน Release นั้น ๆ ที่กำลังจะ Deploy ว่าได้มีการ Config ถูกต้องหรือไม่ โดยที่ Config ที่ต้องการ Check นั้นจะเป็น Config ที่เกี่ยวข้องกับ Azure Databrick ซึ่งเป็น Service ที่ใช้ในการทำงานกับ Big Data ซึ่ง Config file นั้นมีอยู่หลายรูปแบบไม่ว่าจะเป็น YAML, JSON ซึ่งนั้นก็เป็นความถูกต้องของผู้ที่จะต้องทำให้ Check ได้ทุกรูปแบบไฟล์ โดยที่ Logic การcheck จะเขียนด้วย Python และจะต้องทำงานผ่าน Jenkins ดังนั้นก็จะต้องมีการเขียน Jenkins Pipelien ขึ้นมาด้วยนั้นเอง

- **XDO-7880:** Fix wrong variable and IAC definition to ADF Trigger catalog

เนื่องจากในแต่ละ Terraform Module นั้นจะต้องมีการเขียน Documentation ของ Module นั้นๆ ปัญหาที่เจอก็คือ Module มีการ Update ไปมากแล้วแต่ Documentation ไม่ได้ Update ตามไปด้วย ทำให้ผู้ที่ใช้งาน Module เกิดความสับสนในการใช้งาน ซึ่งงานนี้จึงเป็นการ Update Documentation ของ Module ให้สอดคล้องกับ Module ที่ Update ล่าสุด

- **XDO-7921:** Fix bug ADF catalog issue can't provision with user managed identity

งานนี้เป็นการค้นหา Bug ที่ทำให้การพัฒนา Catalog ที่ทำการเรียกใช้ Module กลางของ CCOE นั้นไม่สามารถทำการ terratest (Unit test ของ Terraform) ได้ สาเหตุก็เนื่องมาจากการปรับเปลี่ยนการใช้งานของ Catalog ที่ไม่สอดคล้องกับ Module หลัก

- **XDO-7930:** Update ADF trigger document align with SCB requirement

อัพเดท Document ของ ADF Trigger ให้สอดคล้องกับ Requirement ของ SCB เนื่องจากหลังการประชุมได้มีการตกลง standard ของการ Deploy trigger ขึ้นมาใหม่

- **XDO-7940:** Update ADF trigger document align with SCB requirement

สืบเนื่องจากการดูเหมือน XDO-7483 และ XDO-7484 หลังจากที่พัฒนา Terraform Module เหล่านั้นเสร็จสิ้น ต่อมา ก็จะเป็นการเริ่มทำ migration เพื่อย้ายมาใช้ของใหม่แทนที่ของเดิมที่มีปัญหาเรื่องความยากในการจัดการ

2.3.3 งาน Prove Of Concept

- **XDO-7423: POC ADF Storage Event Trigger Over SFTP**

งานนี้ทางทีม Dev ได้พยายามทามาทางทีม DevOps มาว่า Trigger ชนิด Blob Event trigger ใน ADF นั้นสามารถ trigger เมื่อมี File อัปโหลดผ่าน SFTP เข้าไปได้หรือไม่ ซึ่งผล橙ได้รับหน้าที่ในการหาคำตอบเรื่องนี้จึงได้ตอบมาว่า สามารถทำงานได้แต่ไม่ใช่ Solution ที่ทาง Microsoft ให้มา แต่จะเป็นการ Custom Header Parameter บางตัวเข้าไปใน Trigger เพื่อทำให้ Trigger เองสามารถตรวจจับ Event ที่มาจาก SFTP ได้

- **XDO-7743: CBS-SunCBS | Research on Entra ID with CosmosDB for PostgreSQL**

งานนี้เป็น Research หาข้อมูลและวิธีการใช้งาน Feature ใหม่ของ Cosmos DB ที่ทาง Microsoft ปล่อยออกมาก่อนให้ได้ใช้งาน โดยที่ Feature หลักนั้นคือการทำให้ Entra ID ภายใน Azure สามารถที่จะ Connect กับ Cosmos DB ที่ใช้เป็น PostgreSQL ได้ ซึ่งเป็น Feature ที่สำคัญในการทำงานกับ Database ที่มีขนาดใหญ่ เพราะหากสามารถปรับใช้กับ PostgreSQL ได้จะทำให้การทำงานง่ายมากยิ่งขึ้นไม่จำเป็นต้องใช้ Username และ Password ในการเชื่อมต่อกับ Database อีกด้อไป

- **XDO-7744 | XDO-7760: [CBS] | POC Dynamic create trigger via ARM Template and Documentation**

งานนี้เป็นการหารือวิธีการแก้ไขปัญหาให้กับทาง DevOps ของ SCB โดยปัญหาคือทางทีมของ SCB ไม่ต้องการที่จะใช้ Terraform ในการ Manage Trigger ของ ADF และเลือกที่จะใช้ ARM Template แทน ด้วยเหตุนี้เองผมซึ่งดูผลงานนี้จึงต้องทำการหารือวิธีการในการสร้าง Trigger ให้กับ ADF ผ่าน ARM Template และจะต้องทำ Documentation ของวิธีการใช้งานด้วย และผลลัพธ์ออกมาเป็นสิ่งที่น่าพึงพอใจทำให้วิธีการนี้ถูกนำไปเป็น Standart ให้กับหลายๆ โครงการสืบต่อไปอย่างเช่น Payment Domain Project ก็จะใช้วิธีการนี้ในการสร้าง Trigger ของ ADF ด้วย

- **XDO-7784: POC about dataset reference to use in ADF pipeline.**

สืบเนื่องจาก XDO-7744 ทางทีมที่ได้ลองใช้งานก็เกิดคำถามขึ้นมาว่าหากใช้งาน ARM Template ในการ Deploy Component อื่น ๆ ของ ADF ได้ไหม โดยที่ดูแลกๆ ที่ต้องการใช้งานนั้นคือ Dataset ซึ่ง Dataset นั้นจะต้องมีการ Reference กับ Linked Services ที่เป็น Config ของการเชื่อมต่อ กับ Data Source ดังนั้นงานนี้จึงเป็นการหารือวิธีการในการสร้าง Dataset ที่สามารถ Reference กับ Linked Services ได้โดยที่ไม่ต้องใส่ Config ของ Linked Services ลงใน Dataset โดยตรง ซึ่ง เป็นเรื่องที่สำคัญในการทำงานของ ADF ที่มีขนาดใหญ่

- **XDO-7949: CBS-MCR | Request to test provisioning SSIS with UMI**

เนื่องจาก Project Manager ของ CBS Project ต้องการวิธีการในการทำงานกับ SSIS ที่ต้องใช้ User Managed Identity ในการ Login แทนการใช้ Username และ Password ซึ่งเป็นเรื่องที่สำคัญในการทำงานกับ Database ที่มีขนาดใหญ่ ดังนั้นงานนี้จึงเป็นการทดสอบการใช้งาน SSIS ที่ใช้ User Managed Identity ในการ Login โดยที่การทดสอบนั้นจะต้องทำการเชื่อมต่อกับ Database ที่อยู่ใน Private Network ด้วย

2.4 สวัสดิการที่ได้รับ

บริษัท SCB TechX ให้ความสำคัญกับเรื่องของสวัสดิการที่ดีให้กับพนักงาน โดยเฉพาะนักศึกษาสาขาวิชาที่เข้ามาทำงานในบริษัท ถึงแม้จะไม่ได้เป็นพนักงานประจำ แต่ก็ได้รับสวัสดิการที่ดีจากบริษัทอย่างเช่น

- ทำงาน 5 วัน / สัปดาห์ เข้าบริษัท 2 วัน ตามแต่ละทีมกำหนด และทำงานจากบ้าน 3 วัน (09:00 - 18:00)
- Bus รับส่ง ไป กลับ ที่สถานีรถไฟฟ้า & หมอชิต
- MacBook Pro ให้ใช้งานตลอดระยะเวลาทำงาน ซึ่งจะเป็นเครื่องประจำของคนนั้นๆ เลย
- อาหารเช้าและน้ำดื่มฟรีทุกวันทำงาน
- บัญชีของแหล่งเรียนรู้ออนไลน์ เช่น Udemy ที่สามารถเรียน Course ใหม่ ๆ ที่ได้โดยไม่มีค่าใช้จ่าย
- กิจกรรมพัฒนานักศึกษาฝึกงานและสาขาวิชาทั้งด้าน Soft Skill และ Hard Skill ตลอดระยะเวลาทำงาน
- เปี้ยนเลี้ยงในการทำงาน 500 บาท / วันทำงาน (ไม่นับวันลา หรือวันหยุด)
- วันหยุดประจำปีตามประเทศไทย และวันหยุดพิเศษตามประเทศไทย
- ประกันชีวิต และประกันอุบัติเหตุ
 - ประกันชีวิต 150,000 บาท
 - ค่าประกันอุบัติเหตุ 5,000 บาท

ทั้งนี้ทั้งหมดที่กล่าวมาเป็นเพียงส่วนหนึ่งของสวัสดิการที่ได้รับจากบริษัท SCB TechX และยังมีสวัสดิการอื่น ๆ ที่ยังไม่ได้กล่าวถึง

2.5 วัฒนธรรมองค์กร

บริษัท SCB Techx ยึดถือ 3 Core Value หลักนี้เป็นที่ตั้งสำหรับการทำธุรกิจการของบริษัทประกอบด้วย

- **Respect Individual** ให้ความเคารพและให้ความสำคัญกับผู้อื่น
- **Create value To Customer** สร้างคุณค่าให้กับลูกค้า
- **Deliver Excellence** ส่งมอบสิ่งล้ำค่าและสิ่งที่ดีที่สุดให้กับลูกค้า
- **Innovate And Make It Happen** คิดอย่างสร้างสรรค์และทำให้มันเป็นจริง

2.6 กิจกรรมที่นักศึกษาฝึกงานได้รับ

กิจกรรมที่นักศึกษาฝึกงานได้รับ ที่มี HR จัดทำให้จะเป็นกิจกรรมเกี่ยวกับการพัฒนา Soft Skill และ Hard Skill ของนักศึกษาฝึกงานและสหกิจศึกษาประกอบด้วย

- **Orientation & Ice Breaking:** เป็นกิจกรรมในวันแรกของการเริ่มงานที่บริษัท โดยมีการแนะนำตัวของนักศึกษาฝึกงานและทีมงาน รวมถึงมีกิจกรรม Ice Breaking เพื่อสร้างความคุ้นเคยและทำให้นักศึกษาได้ทำความรู้จักกันมากขึ้น
- **DISC & Workshop & Buddy:** กิจกรรม DISC ช่วยให้นักศึกษาได้รู้จักลักษณะนิสัยและรูปแบบการทำงานของตนเองและเพื่อนร่วมงาน ผ่านการประเมิน DISC และ Workshop เสริมสร้างการทำงานร่วมกัน โดยมีระบบ Buddy ให้คำแนะนำตลอดการฝึกงาน
- **Tips on Building a Strong Profile:** การอบรมเกี่ยวกับวิธีการสร้างโปรไฟล์ที่แข็งแกร่งสำหรับการสมัครงานและพัฒนาตัวเอง ทั้งในด้านการจัดทำเรซูเม่ การเขียนโปรไฟล์ LinkedIn และเคล็ดลับในการแสดงออกอย่างมืออาชีพ
- **Design Thinking Workshop:** กิจกรรมเวิร์กช้อปที่มุ่งเน้นการพัฒนาความคิดเชิงสร้างสรรค์และแก้ปัญหา โดยใช้หลักการ Design Thinking เพื่อช่วยให้นักศึกษามีทักษะในการคิดนอกกรอบและพัฒนาวิธีแก้ปัญหาได้อย่างมีประสิทธิภาพ
- **Agile & Cloud Skills:** การอบรมเกี่ยวกับหลักการทำงานแบบ Agile และทักษะการใช้งาน Azure AI
- **Communication Skills:** กิจกรรมที่มุ่งเน้นการพัฒนาทักษะการสื่อสาร ทั้งการสื่อสารภายในทีม การสื่อสารระหว่างบุคคล และการนำเสนอ เพื่อให้สามารถถ่ายทอดความคิดและข้อมูลได้อย่างมีประสิทธิภาพ
- กิจกรรม Knowledge Sharing ที่จะมีผู้เชี่ยวชาญในด้านต่าง ๆ มาแชร์ความรู้ให้กับพนักงานทุกคน
- Town Hall กิจกรรมที่จะจัด 1 - 2 เดือนครั้งเพื่อให้พนักงานได้พูดคุยกับผู้บริหาร และได้รับข่าวสารจากบริษัท

บทที่ 3

สรุปผลการปฏิบัติงาน

3.1 สรุปผลการศึกษา/การฝึกปฏิบัติงาน

ตามข้อตกลงในเอกสารเงื่อนไขการประเมินผลการปฏิบัติงาน (TOR) ซึ่งได้กำหนดให้ต้องดำเนินงานให้แล้วเสร็จไม่น้อยกว่า 30 งาน ในช่วงระยะเวลาของการฝึกปฏิบัติงาน ข้าพเจ้าได้ดำเนินการแล้วเสร็จจำนวน 32 งาน ทั้งนี้ รายละเอียดของแต่ละงานได้ถูกสรุปไว้ในส่วนของรายงานก่อนหน้านี้อย่างครบถ้วน

จากการดำเนินงานที่กล่าวมาข้างต้น ข้าพเจ้าสามารถบรรลุเป้าหมายของการฝึกปฏิบัติงานตามที่ได้กำหนดไว้ ซึ่งถือเป็นการบรรลุผลสำเร็จตามเกณฑ์ที่วางไว้

3.2 ประโยชน์ที่ได้รับจากการศึกษา/การฝึกปฏิบัติงาน

การฝึกปฏิบัติสหกิจศึกษาที่บริษัท SCB TechX ในตำแหน่ง *Trainee DevOps Engineer* ตลอดระยะเวลา 5 เดือนนั้น ทำให้ผมได้รับประสบการณ์ที่มีค่ามากมาย โดยเฉพาะการทำงานร่วมกับทีมและการเรียนรู้ทักษะใหม่ ๆ ที่ไม่เคยได้ศึกษามาก่อน การได้รับโอกาสจากเพื่อนทีมและหัวหน้างานที่คอยช่วยเหลือและให้คำแนะนำเป็นสิ่งสำคัญมาก โดยเฉพาะพี่เลี้ยงของผมที่ให้ความไว้วางใจและเปิดโอกาสให้ผมได้ลองผิดลองถูกในการทำงานจริง ซึ่งช่วยพัฒนาทักษะและความมั่นใจในการทำงานอย่างมาก ทำให้ประสบการณ์สหกิจศึกษาครั้งนี้ประสบความสำเร็จ

ในส่วนของ *Hard Skills* ที่ได้เรียนรู้ หนึ่งในสิ่งที่สำคัญคือการได้ลองใช้งาน *Azure Cloud Provider* ซึ่งเป็นความรู้ที่ไม่ได้เรียนในห้องเรียน การได้เห็นและทดลองใช้ *Azure* กับระบบจริงของธนาคาร ซึ่งเป็นระบบที่มีความซับซ้อนและสำคัญมาก ทำให้ผมเข้าใจถึงการออกแบบและการจัดการระบบที่ซับซ้อนมากขึ้น พี่เลี้ยงของผมได้มอบหมายงานเกี่ยวกับ *Azure* ให้ผมรับผิดชอบส่วนใหญ่ ทำให้ผมได้เรียนรู้วิธีการใช้งาน *Azure* อย่างลึกซึ้ง และได้นำความรู้นี้ไปปรับใช้ใน โปรเจกต์ ของผม ทั้งในด้านการออกแบบสถาปัตยกรรมระบบ (*System Architecture*) และการประยุกต์ใช้ *Azure* ให้เหมาะสมกับ-project

นอกจากนี้ *Soft Skills* ก็เป็นสิ่งที่ผมได้พัฒนาระหว่างการทำงานในสหกิจรั้งนี้ บริษัท SCB TechX มีกิจกรรมให้เข้าร่วมเป็นประจำเพื่อส่งเสริม *Soft Skills* เช่น การทำงานร่วมกับคนต่าง *Generation*, ทักษะการสื่อสาร, การเข้าสังคม และการสร้างโปรดไฟล์สำหรับการสมัครงานในอนาคต กิจกรรมเหล่านี้ช่วยให้ผมพัฒนาความสามารถในการทำงานร่วมกับผู้อื่นและสร้างเครือข่ายที่ดีสำหรับอนาคต

3.3 ข้อเสนอจากบริษัท

จากการพูดคุยกันอย่างไม่เป็นทางการกับหัวหน้างาน ทราบว่าทางทีม DevOps ของบริษัท SCB TechX ไม่มีตำแหน่งว่างในขณะนี้ จึงไม่มีข้อเสนอรับเข้าทำงานในตำแหน่งนี้ อย่างไรก็ตาม หัวหน้าทีม DevOps Manager (Platform Service Manager) ได้แนะนำให้สมัครในตำแหน่งอื่นที่ไม่ใช่ DevOps โดยแนะนำตำแหน่ง Platform Engineer ซึ่งมีลักษณะงานที่ไม่แตกต่างจาก DevOps มากนัก ทั้งนี้ในระหว่างการฝึกงาน ผมมีโอกาสได้ช่วยงานทีม Platform Engineer อยู่บ้าง จึงเชื่อว่าสามารถทำงานในตำแหน่งนี้ได้โดยไม่ต้องมีการสอนงานเพิ่มเติมมากนัก นอกจากนี้ ยังมีโอกาสเปลี่ยนแปลงตำแหน่งรายหลังได้ตามความเหมาะสม

ในส่วนของฐานเงินเดือนเริ่มนั้นอยู่ที่ 30,000 บาทต่อเดือน โดยอาจปรับเพิ่มได้ตามประสบการณ์และความสามารถของผู้สมัคร

បរទេសាន្តកម្ម

- [1] About scb techx. SCB TechX. [Online]. Available: <https://scbtechx.io/th/about-us/>
- [2] Innovative products. SCB TechX. [Online]. Available: <https://scbtechx.io/th/services-products/>
- [3] What is kyc. TMBThanachart Bank. [Online]. Available: <https://www.ttbbank.com/th/corporate/corp-digital-banking-and-other-services/other-service-crop/kyc-cdd>
- [4] Ci/cd. Red Hat. [Online]. Available: <https://www.redhat.com/en/topics/devops/what-is-ci-cd>

ภาคผนวก

ภาคผนวก ก
เอกสารสำคัญที่เกี่ยวข้อง

• เอกสารสำคัญของคณะ

- หนังสือยินยอมให้เผยแพร่รายงานปฏิบัติงานสหกิจศึกษา (หน้า 19)
- วศ.สก.-03/04 รายงานตัวเข้าสหกิจศึกษา (หน้า 20)
- วศ.ฝ.ค-03 แบบฟอร์มยืนยันการรับนักศึกษา (หน้า 21)
- วศ.สก.-11 แบบสรุปผลการปฏิบัติงานสหกิจศึกษา (หน้า 22)

• เอกสารสำคัญที่เกี่ยวข้องกับงาน

- เอกสารสรุปสิ่งที่ได้เรียนรู้และลองทำ Docker (หน้า 29)
- เอกสารสรุปสิ่งที่ได้เรียนรู้และลองทำ Kubernetes (หน้า 36)
- เอกสารสรุปสิ่งที่ได้เรียนรู้และลองทำ Jenkins (หน้า 42)
- เอกสารสรุปสิ่งที่ได้เรียนรู้และลองทำ Terraform (หน้า 52)
- เอกสารสรุปสิ่งที่ได้เรียนรู้เกี่ยวกับ Monitoring Tools (หน้า 56)
- เอกสารสรุปสิ่งที่ได้เรียนรู้เกี่ยวกับ ELK Stack (หน้า 62)



ศูนย์ Entaneer Academy

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่

หนังสือยินยอมให้เผยแพร่รายงานปฏิบัติงานสหกิจศึกษา

เพื่อเป็นการส่งเสริมการพัฒนาการศึกษาของประเทศไทย ข้าพเจ้าในฐานะตัวแทนหน่วยงานหรือบริษัท...ผู้สัมภาระนี้ ลงชื่อ.....(SCB ๒๕๖๘ ๑๐.๗.๔๙) มีความยินดีให้คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่ เผยแพร่เนื้อหาในรายงานสรุปผลโครงการของนักศึกษา ภาคการศึกษาที่ ๑ ประจำปีการศึกษา ๒๕๖๗ ณ สถานประกอบการของข้าพเจ้าในส่วนของ “กิจกรรมที่นักศึกษาสหกิจศึกษาทำรายงาน/โครงการ” “บทคัดย่อ” และ “ข้อเสนอแนะในรายงาน/โครงการ” โดย

- อนุญาตให้ระบุชื่อหน่วยงานหรือบริษัท
- ไม่อนุญาต
- อื่นๆ โปรดระบุ.....

ลงชื่อ.....
.....ผู้มีอำนาจกระทำการแทน
(.....)
นิติบุคคล/ผู้ประกอบการ
วันที่ ๒๕/๑๐/๒๔

ประทับตรา บริษัท / โรงงาน (ถ้ามี)

ศูนย์ Entaneer Academy (สาขาวิชานโยบายและภารกิจ)

239 ต.สุเทพ อ.เมือง จ.เชียงใหม่ 50200 โทรศัพท์ (053)944179 ต่อ 108 โทรสาร (053)944113 E-mail:interncoop@eng.cmu.ac.th

ใบรายงานตัวเข้ารับการฝึกศหกิจ

[2086]

ชื่อนักศึกษา ณัฐพงษ์ เทพพิทักษ์
 รหัสนักศึกษา 640610634
 นักศึกษาสาขาวิชาศุภกรรม คอมพิวเตอร์
 คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่
 เบอร์โทรศัพท์นักศึกษา 0955301640
 เบอร์โทรศัพท์ผู้ปกครอง (กรณีฉุกเฉิน) 0928205909
 ให้รายงานตัวเพื่อเข้ารับการฝึกศหกิจ ณ บริษัท เอสซีปี เทคโนโลยี จำกัด
 ที่อยู่ 19 อาคารไทยพาณิชย์ปาร์ค พลาซ่า เวสท์ปี ชั้นที่ 21 ช. 18 ถ. รัชดาภิเษก แขวงจตุจักร เขตจตุจักร กรุงเทพมหานคร 10900

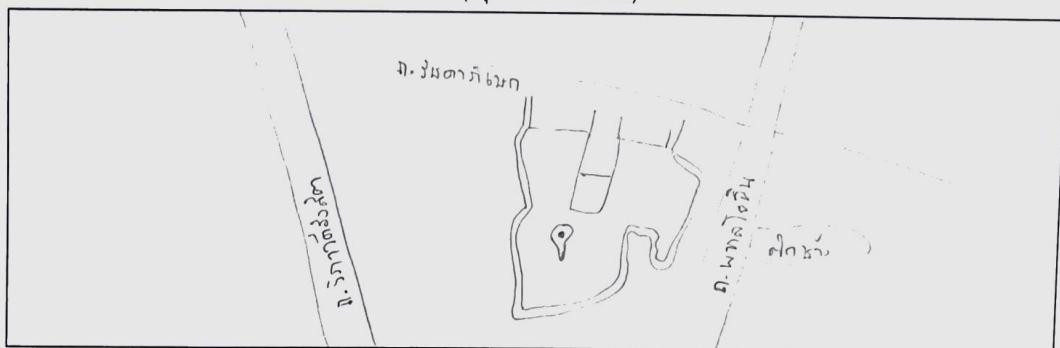
ผู้ควบคุมการฝึกศหกิจ Phataraphon Muakmanee	ตำแหน่ง DevOps Engineer
โทรศัพท์ 0859903803	โทรศัพท์ 028539600
พี่เลี้ยง Phataraphon Muakmanee	โทรศัพท์ 0859903803 อีเมล phataraphon.muakmanee@scbtechx.io
ผู้ประสานงานการนิเทศ Phataraphon Muakmanee	โทรศัพท์ 0859903803 อีเมล phataraphon.muakmanee@scbtechx.io

ลักษณะงานที่นักศึกษารับผิดชอบ Supporting and working on the deployment and development of the development environment.

ระยะเวลาการฝึกศหกิจ 4 มี.ย. 2567 ถึง 31 ต.ค. 2567 เวลาเข้าฝึกศหกิจ 09:00 น. เวลาออกฝึกศหกิจ 18:00 น.

วันหยุดบวชท วันหยุดนันเสาร์ วันหยุดอาทิตย์
 วันหยุดอื่น ๆ 22-Jul-24, 29-Jul-24, 12-Aug-24, 14-Oct-24
 รวมเป็นเวลา 103 วัน (นับเฉพาะวันที่เข้ารับการฝึกศหกิจ)
 ที่พักนักศึกษาจะเดินทางเข้ารับการฝึกศหกิจ JS Residence (Phahon Yothin 23)

แผนที่แสดงตำแหน่งของสถานที่ฝึกศหกิจโดยสังเขป (กรุณาวัดด้วยตนเอง)



ในการฝึกศหกิจครั้งนี้ ข้าพเจ้าจะปฏิบัติตามกฎระเบียบและข้อบังคับ ตลอดจนรับผิดชอบงานที่ได้รับมอบหมายจากทางสถานที่ รับฝึกศหกิจดังกล่าวข้างต้นอย่างเต็มความสามารถ และในการฝึกศหกิจจะไม่นำความลับ ข้อมูล รวมทั้งรูปถ่ายและคลิปวิดีโอ ภายในบริษัทออกมายกเว้นไว้ ณ วันนี้ อนุญาตถูกกฎหมาย

ลงชื่อ..... ก.พ.ร.พ. นามสกุล.....

(ณัฐพงษ์ เทพพิทักษ์)

(Phataraphon Muakmanee)

นักศึกษาผู้เข้ารับการฝึกศหกิจ¹
 วันที่ ..18.. / ..กันยายน.. / ..2567..

ผู้ควบคุมการฝึกศหกิจ

วันที่ ..16.. / ..กันยายน.. / ..2567..

ประทับตรา บริษัท/โรงงาน (ถ้ามี)



ศูนย์ Entaneer Academy คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่

แบบฟอร์มยืนยันการรับนักศึกษา

ผู้ดูแล สาขาวิชา

ชื่อบริษัท/หน่วยงาน บริษัท เอสซีบี เทคโนโลยี จำกัด (SCB TechX Co., Ltd.)

ที่ตั้ง 19 อาคารไทยพาณิชย์ ปาร์ค พลาซ่า เวสท์ ชั้นที่ 21 จังหวัด กรุงเทพมหานคร รหัสไปรษณีย์ 10900
โทรศัพท์ 02-853-9600 โทรสาร

E-mail

ประเภทกิจการ IT Services and IT Consulting

โดยมีลักษณะงานสำหรับนักศึกษาดังนี้ (ระบุพื้นที่) Devops

สถานที่จัดส่งนักศึกษาเข้ารับการฝึกงาน ตามที่อยู่ข้างต้น สำนักงานสาขา
ที่ตั้ง จังหวัด รหัสไปรษณีย์

โทรศัพท์ โทรสาร

ฝึกงาน ตามเวลาที่คณฯ กำหนด อื่น ๆ (โปรดระบุ)

วัน เวลาฝึกงาน วันจันทร์ - ศุกร์ วันเสาร์ วันอาทิตย์ เวลา 9.00 น. ถึง 18.00 น.
 อื่น ๆ

สวัสดิการอื่น ๆ ที่จัดให้ (ถ้ามี)

ที่พักของนักศึกษา จัดที่พักให้ ไม่ได้จัดที่พักให้ จ่ายค่าที่พักให้ในอัตรา บาท
ค่าเบี้ยเลี้ยงของนักศึกษา มี ในอัตรา 500 บาท/วัน ไม่มี

ทางบริษัท/หน่วยงาน มีความประสงค์ (โปรดทำเครื่องหมาย ✓)

รับนักศึกษาฝึกงาน จำนวน 1 คน ดังนี้

๑. (นาย/น.ส.) ณัฐพงษ์ เพทพิทักษ์ รหัสประจำตัว 640610634 สาขาวิชา computer engineering

๒. (นาย/น.ส.) รหัสประจำตัว สาขาวิชา

๓. (นาย/น.ส.) รหัสประจำตัว สาขาวิชา

มีความต้องการอื่น ๆ เพิ่มเติม (โปรดระบุ)

ข้อคิดเห็นอื่น ๆ (ถ้ามี)

ลงนาม.....ภัทรพล มนูกุล

(ภัทรพล มนูกุล)

ตำแหน่ง DevOps Engineer

[๒๐๘๖]

กรุณาส่งแบบตอบรับ ภายใน ๑ สัปดาห์ด้วย จักขอบพระคุณยิ่ง

หมายเหตุ : โปรดกรอกเอกสารและส่งคืน ศูนย์ Entaneer Academy คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่
๒๓๙ ต.สุเทพ อ.เมืองเชียงใหม่ จ.เชียงใหม่ ๕๐๒๐๐ โทรศัพท์ (๐๕๓)๔๔๔๐๗๘ โทรสาร (๐๕๓)๔๔๔๐๑๓ E-mail: interncoop@eng.cmu.ac.th
ผู้ประสานงาน คุณอรกัญญา พุทธวงศ์



งานบริการการศึกษาและพัฒนาคุณภาพนักศึกษา
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่

แบบสรุปผลการปฏิบัติงานสหกิจศึกษา

ชื่อ-สกุล (นาย/นางสาว) นิติพงษ์ เทพพันธุ์กุญชร รหัสนักศึกษา 640610634
ภาควิชา/วิศวกรรม คอมพิวเตอร์ ชื่อ-สกุล พนักงานที่ปรึกษา ภัทรรุจิ นุ่มใจคง
ตำแหน่ง DevOps Engineer โทรศัพท์ 085-9903803 E-Mail: phataraphon.w@outlook.co.th
ชื่อสถานที่ฝึกสหกิจศึกษา SCB TechX CO., LTD.
ระยะเวลาการฝึกสหกิจศึกษา ระหว่างวันที่ 4 มีนาคม ถึง 25 มกราคม

ขอสรุปผลการปฏิบัติงานตลอดระยะเวลาการฝึกสหกิจศึกษาของข้าพเจ้า ดังนี้

- | | | | |
|--------------|-------|----------|-------------------------------|
| 1. วันลาป่วย | จำนวน | <u>0</u> | วัน (แนบใบลาที่ได้รับอนุมัติ) |
| 2. วันลาภิจ | จำนวน | <u>0</u> | วัน (แนบใบลาที่ได้รับอนุมัติ) |
| 3. วันขาดงาน | จำนวน | <u>0</u> | วัน |

คงเหลือจำนวนวันเข้าฝึกสหกิจศึกษา 79 วัน

ลงชื่อ ภัทรรุจิ นุ่มใจคง
(ภัทรรุจิ นุ่มใจคง)
พนักงานที่ปรึกษาสหกิจศึกษา
25 / 10 / 67
ประทับตรา บริษัท/โรงงาน (ถ้ามี)

ลงชื่อ นิติพงษ์ เทพพันธุ์กุญชร
(นิติพงษ์ เทพพันธุ์กุญชร)
นักศึกษาผู้เข้ารับการฝึกสหกิจศึกษา
25 / 10 / 67

Docker Tutorial

What is container :

A process running on host that isolated from host process

What is image :

When we need to run container we need to use an image. Image contain everything needed to run an application - all dependencies, configurations, scripts, binaries. So that means it portable and can be run on any where and any OS.

Docker installation

- Mac
 - Enter this site : <https://docs.docker.com/desktop/install/mac-install/>
 - Then following the description.
- Ubuntu (full docs on this site :
<https://docs.docker.com/engine/install/ubuntu/>)
 - Set up Docker's `apt` repository.
 - install docker package

```
sudo apt-get install \
docker-ce docker-ce-cli \
containerd.io \
docker-buildx-plugin \
docker-compose-plugin \
```

- Add a user group docker to your user via

```
sudo usermod -aG docker $USER
```

-a mean append user group from exist usergroup
-G mean group

-aG mean append new group from existing group

- Now we are getting and initial setup docker on our device already, next we will try to run some container to show simple page.

```
docker run -dp 80:80 yeasy/simple-web:latest
```

If you are work perfectly you will see the result like this via enter IP:
127.0.0.1 or localhost



Create docker image

when you implement your application you need to build an docker image for that project to be run it on container, in this chapter we will show some simple way to create docker image via dockerFile and try to run.

- Firstly, I will implement some simple nodes application to be an example app for building image.
 - Create empty folder to store our source code.
 - execute this command

```
npm init -y #init node project
```

```
npm i express # install express library
```

- c. go to package.json then add `"type" : "module"` into your package.json file. the result should be like this.

```
{
  "name": "simple-node-app",
  "version": "1.0.0",
  "main": "index.js", //added line
  "type": "module",
  "scripts": {
    "test": "echo \\"$Error: no test specified\\" && exit 0"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "express": "^4.19.2"
  }
}
```

- d. create file index.js then write some simple code like this.

```
import express from "express";
import os from "os";

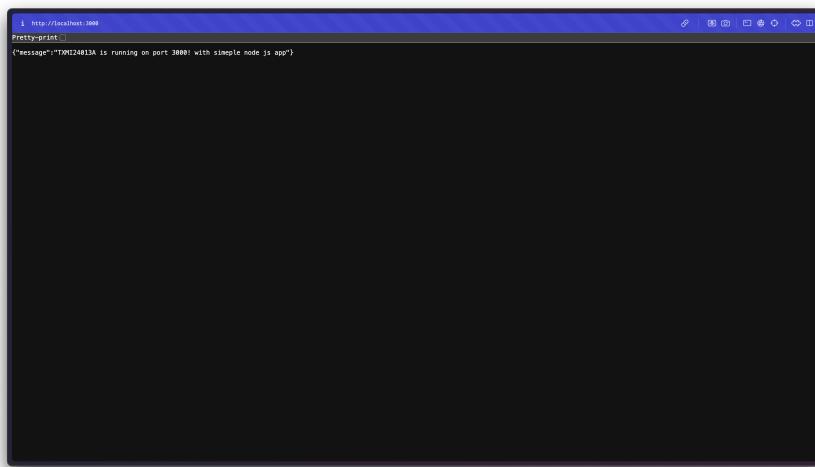
const app = express();
const port = 3000;

app.get("/", (req, res) => {
  res.send({
    message: `${os.hostname()} is running on port ${port}`,
    simpleNodeJsApp,
  });
});

app.listen(port, () => {
```

```
        console.log(`Server is running on port ${port}`);
    });
}
```

- e. try to run this app via command `node index.js` , then enter website with localhost:3000 the result should be



- f. Next, we will write DockerFile to create a docker image for this simple app.

- i. Create DockerFile then write some simple script like this

```
// DockerFile

FROM node:lts-alpine
WORKDIR /usr/src/app
COPY ["package.json", "package-lock.json*", "./"]
RUN npm install && mv node_modules ../
COPY .
EXPOSE 3000
RUN chown -R node /usr/src/app
USER node
CMD ["node", "index.js"]
```

- ii. try to build image via `docker build . -t simple-app`

- ,
- iii. if everything work perfectly when you type `docker image` you need to see an simple-app image
- iv. running an app via `docker run -dp 80:3000 simple-app` , then enter website `localhost:80` you will able to see the same result as before running without docker but device name will be change to docker container id.

```
{"message":"5320be4c6af is running on port 3000! with simple node js app"}
```

Push image to registry

- Public registry

Form the previous chapter we had created an docker image call `simple-app` in this part we will push it on DockerHub public registry

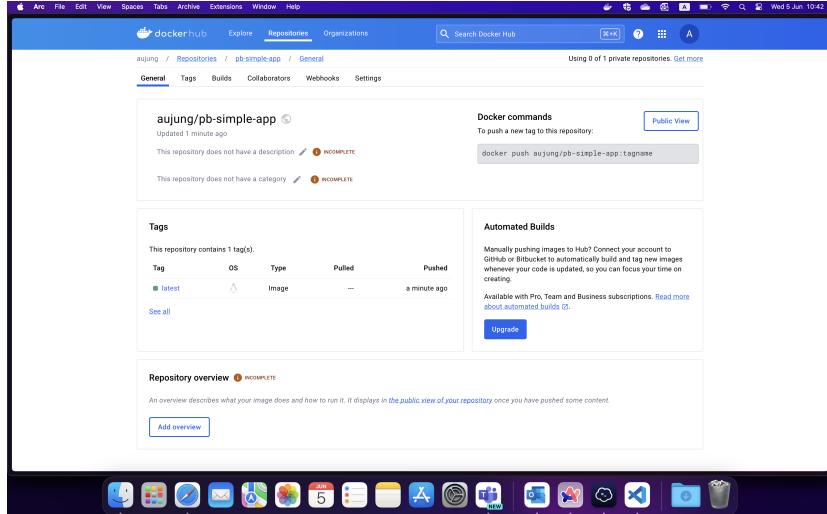
before we push we need to create an repository on docker hub.

- Firstly, we need to give an tag to simple-app before pushing to registry, then push it.

```
docker tag simple-app YOUR-USER-NAME/REPO_NAME
```

```
docker push YOUR-USER-NAME/REPO_NAME
```

- If everything work fine you will see the result look like this



- Private registry

In this part everything is the same as public repository but there are some several thing need to change in execution command.

```
docker tag [OPTIONS] IMAGE[:TAG] [REGISTRYHOST/]USERNA  
 docker push NAME[:TAG]
```

Running container

There are several to run a docker container a way we use it already is run with `docker run` and another version is writing `docker-compose` file, So in this chapter we will create simple docker-compose file to run our simple-app

1. create `docker-compose.yml` then write some script like this.

```
version: "3"  
services:  
  simple-app:  
    image: IMAGETAG //IMAGE TAG FROM previous section  
    ports:  
      - "80:3000"
```

2. run docker `compose up -d` to run container with simple-app image'

Container file system volumes and bind mounts

Each container also gets its own "scratch space" to create/update/remove files. Any changes won't be seen in another container, even if they're using the same image.

volume

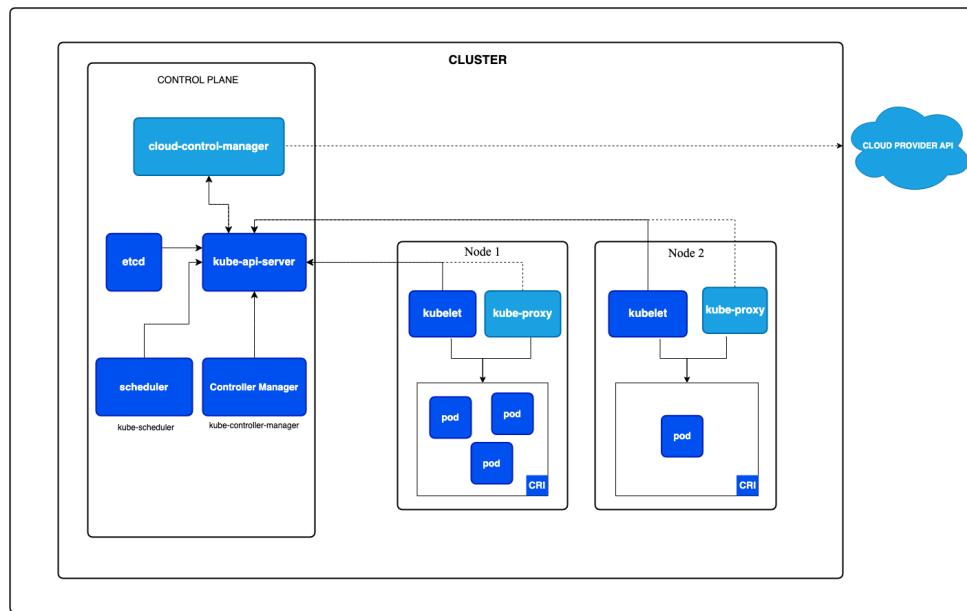
Volumes provide the ability to connect specific filesystem paths of the container back to the host machine. If you mount a directory in the container, changes in that directory are also seen on the host machine. If you mount that same directory across container restarts, you'd see the same files.

Networking

Container networking refers to the ability for containers to connect to and communicate with each other, or to non-Docker workloads.

Kubernetes - K8s

Kubernetes Architecture



In each cluster there are three main components that we need to know before working with K8s

- Master Node (Control plane)
- Worker Node
- etcd

Master Node

- Controlled and managed all of operation in K8s included
 - **kube-api-server** : act as an intermediary for cluster communication via RESTful API
 - **kube-scheduler** : act as an pod manager when new Pos comes in it will find the appropriate worker node for that coming pod which requires information from many sources to make decisions
 - **kube-contoller-manager** :
 - Node Controller
 - Replication Controller : manage pod replication
 - Endpoints Controller

- Service Account & Token controllers : manage service account and API access token
- cloud-controller-manager : this controller is available in only Cloud Provider that support K8s. It act like a kube-controller-manager but working with Cloud Provider instead.

Worker Node

- Container runtime included two services
 - kubelet : Receive commands from kube-api-server and manage container runtime, such as checking if Pods (Containers) are still running on Worker nodes. If there is a problem, it will report to kube-api-server that there is a problem with the Pods and make them manage the Initial Pods again. (manager of Pods)
 - kube-proxy : this service help to make Pods can be connected from outside the Cluster it act like a proxy server to receive request from outside then resend again to appropriate pods inside the node.

etcd

Use to store all of Persistance cluster state this components also in control plane which directly connects by kube-api-server

Networking in Kubernetes

- **Container-to-container** communication within a Pod
- **Pod-to-Pod** communication across nodes
- **Pod-to-Service** communication within the cluster
- **External-to-Service** communication with the outside world

Type of Objects in Kubernetes

- Pods : A Pod can host a single container or a group of containers. It's important to note that, when a Pod contains multiple containers, all of the containers always run on a single worker node. **Pods is that they are ephemeral in nature**
- Deployment : **allows you to declaratively manage the desired state of your application**, such as the number of replicas, the image to use for the Pods, and the resources required.
- ReplicaSets : **Deployments don't manage Pods directly**. That's the job of the ReplicaSet object.
 - ReplicaSet is created automatically

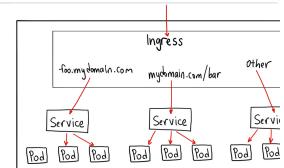
- The ReplicaSet ensures that the desired number of replicas (copies) are running at all times by creating or deleting Pods as needed.
- **StatefulSet**
 - **ensures that each Pod is uniquely identified by a number, starting at zero.**
 - When a Pod in a StatefulSet must be replaced, for example, due to node failure, the new Pod is assigned the same numeric identifier as the previous Pod.
- **DaemonSets**
 - **ensures that a copy of a Pod is running across all, or a subset of nodes in a Kubernetes cluster.**
- **PersistentVolume**
 - **represents a piece of storage that you can attach to your Pod(s).**
 - if your Pod gets deleted, the PersistentVolume will survive.
 - can attach using a PersistentVolume, like local disks, network storage, and cloud storage.
- Service : **a way to access a group of Pods that provide the same functionality.** the Service also provides some simple load balancing.
- Namespaces
 - **a way to divide a single Kubernetes cluster into multiple virtual clusters.**

ClusterIP vs NodePort vs Loadbalancer

Kubernetes NodePort vs LoadBalancer vs Ingress? When should I use what?

Recently, someone asked me what the difference between NodePorts, LoadBalancers, and Ingress were. They are all different ways to get...

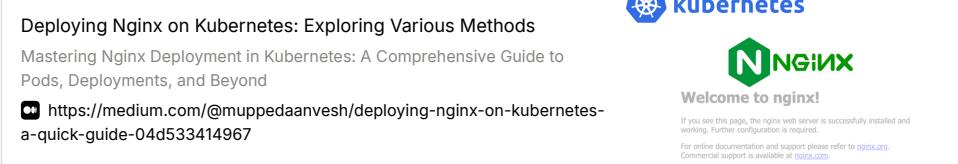
 <https://medium.com/google-cloud/kubernetes-nodeport-vs-loadbalancer-vs-ingress-when-should-i-use-what-922f010849e0>



- ClusterIP : the default Kubernetes service. It gives you a service inside your cluster that other apps inside your cluster can access. There is no external access.
- NodePort : **is a way to expose your application to external clients but doesn't contain LoadBalancer**
- LoadBalancer : is NodePort bit including LoadBalancer
 - key different NodePort and LoadBalancer
 - NodePort : Client → node
 - LoadBalancer : client → load balancer → node

Deploying Nginx on Kuberetes Various way.

,

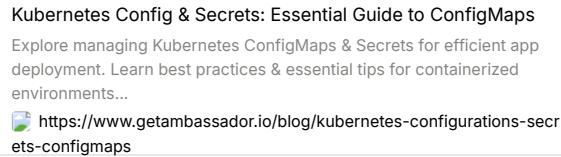


Deploying Nginx on Kubernetes: Exploring Various Methods
Mastering Nginx Deployment in Kubernetes: A Comprehensive Guide to Pods, Deployments, and Beyond
<https://medium.com/@muppedaanvesh/deploying-nginx-on-kubernetes-a-quick-guide-04d533414967>

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

ConfigMap & Secret

- ConfigMap
 - non-confidential
 - key-value format
 - can be injected into container
- Secret



Kubernetes Config & Secrets: Essential Guide to ConfigMaps
Explore managing Kubernetes ConfigMaps & Secrets for efficient app deployment. Learn best practices & essential tips for containerized environments...
<https://www.getambassador.io/blog/kubernetes-configurations-secrets-configmaps>



Blog
Managing Configurations and Secrets in Kubernetes: ConfigMaps and Secrets

- The same as ConfigMap
- sensitive information

Workshop

Initially, I use docker desktop on Mac and enable to use an Kubernetes already,

- first I will check if Kubernetes is working by
 - run `kubectl cluster-info` to check if there is any response to me
 - the response is

```
kubectl cluster-info
Kubernetes control plane is running at https://127.0.0.1:6443
CoreDNS is running at https://127.0.0.1:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
```

- Next I will try to deploy Nginx with many method and strategy. Before we start we need to create new namespace for this learning section via `kubectl create namespace learning`. After we created let's check the result if everything work find let's get started.

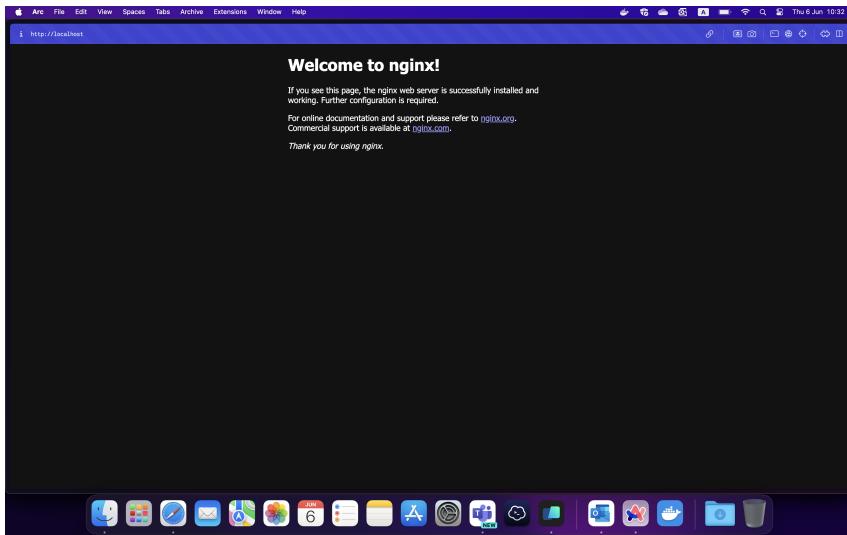
```
kubectl get namespace
NAME          STATUS  AGE
default       Active  20h
kube-node-lease  Active  20h
kube-public    Active  20h
kube-system    Active  20h
learning      Active  18s
```

- Imperative Way

- RUN `kubectl run nginx-pod --image=nginx --port=80 -n learning`
- Now check the pod that created

```
kubectl get pod -n learning
NAME        READY  STATUS   RESTARTS  AGE
nginx-pod  1/1    Running  0         2m11s
```

- Now if you need to enter this site this deployed, Sorry you can't because you need to setup for some service that would be the door that external user need to connect to. let's type `kubectl expose pod nginx-pod --type=LoadBalancer --port=80 --name=nginx-service --namespace=learning`
- then check the result in website on localhost:80 should be look like this.



- Use a manifest file

- Create a file call nginx.yaml and copy this script to create pod and service like above.

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  labels:
    app: nginx
    namespace: learning
spec:
  containers:
  - name: nginx-container
    image: nginx:latest
    ports:
    - containerPort: 80
  resources:
    limits:
      memory: "128Mi"
      cpu: "500m"
    requests:
      memory: "64Mi"
      cpu: "250m"

  ---

apiVersion: v1
kind: Service
```

```
,  
  
  metadata:  
    name: nginx-service  
    namespace: learning  
  spec:  
    selector:  
      app: nginx  
    ports:  
      - protocol: TCP  
        port: 80  
        targetPort: 80  
    type: LoadBalancer
```

- after that save the file then run `kubectl apply -f nginx.yaml`

```
~/personal/learnKube * docker-desktop (0.228s)  
kubectl apply -f nginx.yaml  
pod/nginx-pod unchanged  
service/nginx-service unchanged
```

- now check the result.

Jenkins

Installation

- Terraform install script
 - Vm setup is the same as Terraform
 - install Jenkins script

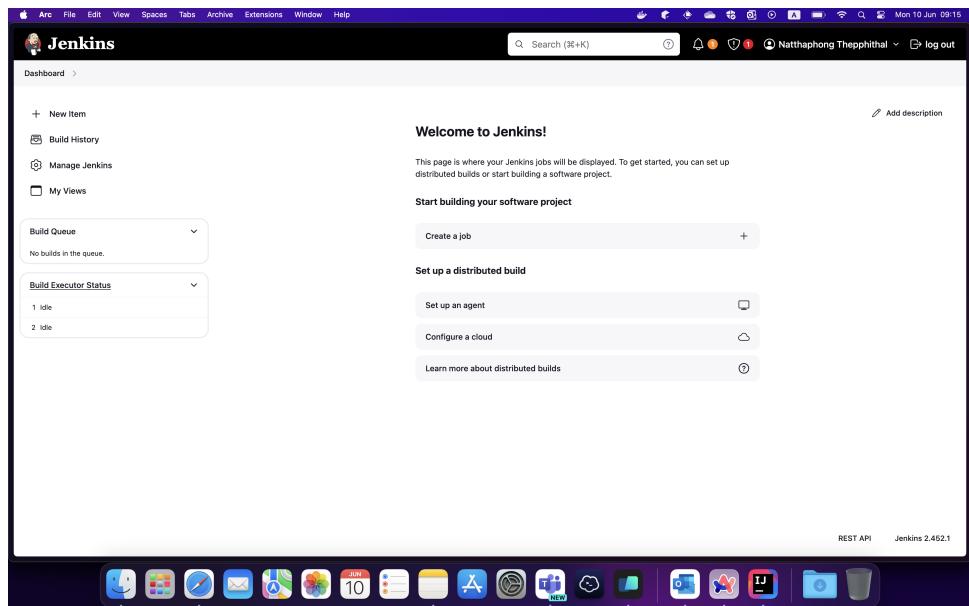
```
resource "null_resource" "install_jenkins" {  
    triggers = {  
        vm_id = azurerm_linux_virtual_machine.vm.id  
    }  
  
    connection {  
        type      = "ssh"  
        host      = azurerm_linux_virtual_machine.vm.p  
        user      = azurerm_linux_virtual_machine.vm.a  
        private_key = file("~/ssh/id_rsa")  
    }  
  
    provisioner "remote-exec" {  
        inline = [  
            "sudo apt-get update -y",  
            "sudo apt-get install -y openjdk-11-jdk",  
            "sudo wget -O /usr/share/keyrings/jenkins-keyr  
            https://pkg.jenkins.io/debian-stable/jenkins.i  
            "echo \"deb [signed-by=/usr/share/keyrings/jen  
            https://pkg.jenkins.io/debian-stable binary/\\""  
            sudo tee /etc/apt/sources.list.d/jenkins.list :  
            "sudo apt-get update -y",  
            "sudo apt-get install -y jenkins",  
            "sudo systemctl start jenkins",  
            "sudo systemctl enable jenkins"  
        ]  
    }  
}
```

enter ip-address:8080

- After that do following Jenkins instructions.

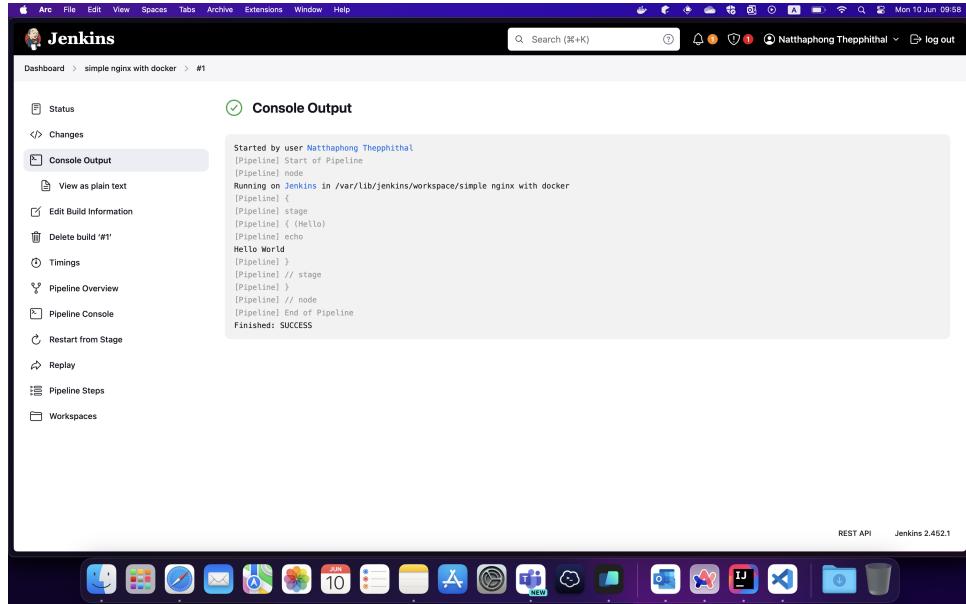


If done with initial setup will see :



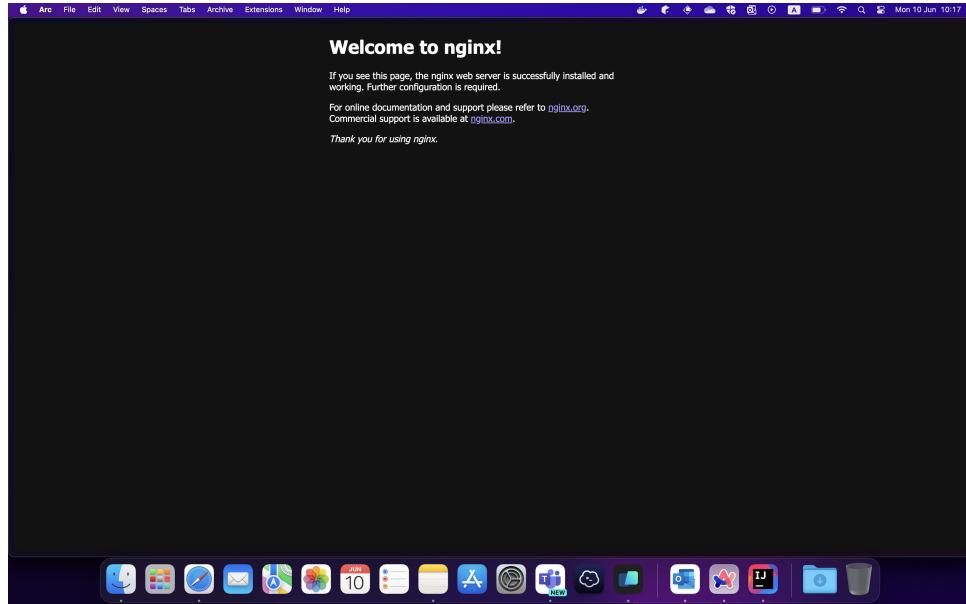
Create first simple groovy script for run Nginx on docker.

- Firstly, let's test more simple script to check if jenkins is working.



```
pipeline {
    agent any

    stages {
        stage('Hello') {
            steps {
                echo 'Hello World'
            }
        }
    }
}
```



- now create let pull Nginx docker image and run it with Jenkins.

```
pipeline {  
    agent any  
  
    stages {  
        stage('Pull Docker Image') {  
            steps {  
                script {  
                    sh 'docker pull nginx:alpine'  
                }  
            }  
        }  
  
        stage('Remove Docker Container') {  
            steps {  
                script {  
                    sh 'docker rm -f mynginx'  
                }  
            }  
        }  
    }  
}
```

```

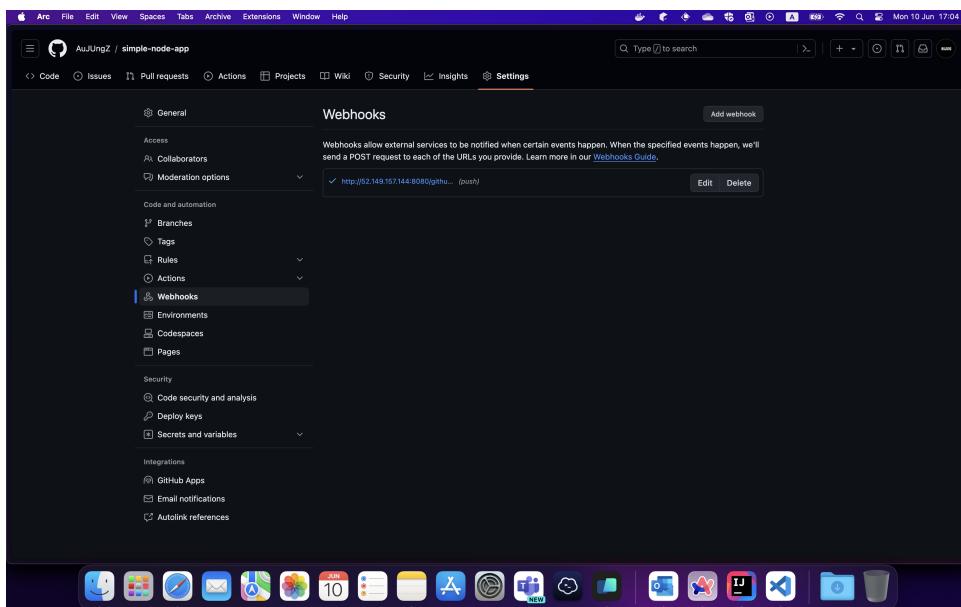
}

stage('Run Docker Container') {
    steps {
        script {
            sh 'docker run -d -p 80:80 --name mynginx'
        }
    }
}
}

```

Github web hook with Jenkins

- add Jenkins web hook for GitHub web hook add `/github-webhook` at the end of Jenkins url.



- Now when on git repository has new commit Jenkins pipeline will automatically redo again.

Set Slave agent

- master Node
 - install java jdk and install jenkins sercie.
 - generate ssh key for that master node via `ssh-keygen`
- Slave Node
 - install only **java jdk**.
 - place public ssh key of master agent on working user on slave node.

Master Node UI settings.

- Create global credential that store ssh private key of Master Node.
- Connect to slave node
 - add new node with ssh method with created credential

Terraform

Installation

- Use Homebrew package manager.

```
brew tap hashicorp/tap
brew install hashicorp/tap/terraform
```

- Try simple to use terraform on local machine.

- Create file .tf

```
#To create a file that contain some text

resource "null_resource" "file-create" {
    provisioner "local-exec" {
        command = "echo 'Hello, World!' > hello.txt"
    }
}

terraform init
terraform plan
terraform apply
```

- Now check inside project folder you will see `hello.txt` file, inside contain "Hello, world"

Terraform with azure

- After installed already if we need to use terraform with any provider we need to do **Authenticating to Azure**
 - **Authenticating to Azure using the Azure CLI**
 - `az login` (login via website method)
 - `az account set --subscription="SUBSCRIPTION_ID"`
 - Let's create simple script for create

- resource group
- storage account including
 - web hosting file
- new file call main.tf

```

provider "azurerm" {
  features {}
}

#create a resource group
resource "azurerm_resource_group" "rg" {
  name      = "rg-learnTerraform"
  location  = "East Asia"
}

#create a Storage Account
resource "azurerm_storage_account" "storage" {
  name                  = "aujungterraformstorage"
  resource_group_name   = azurerm_resource_group.rg.name
  location              = azurerm_resource_group.rg.location
  account_tier          = "Standard"
  account_replication_type = "LRS"
  account_kind = "StorageV2" // StorageV2 is required for static website

  static_website {
    index_document = "index.html"
  }
}

#add index.html file
resource "azurerm_storage_blob" "blob" {
  name            = "index.html"
  storage_account_name = azurerm_storage_account.storage.name
  storage_container_name = "$web"
  type            = "Block"
  content_type    = "text/html"
}

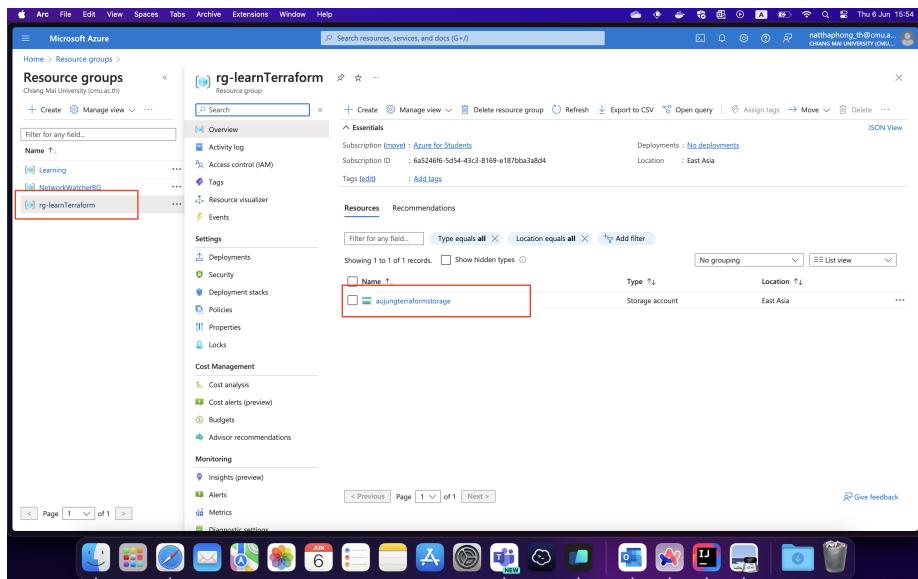
```

```
source_content = "<html><body><h1>Hello, Terraform! from
}
```

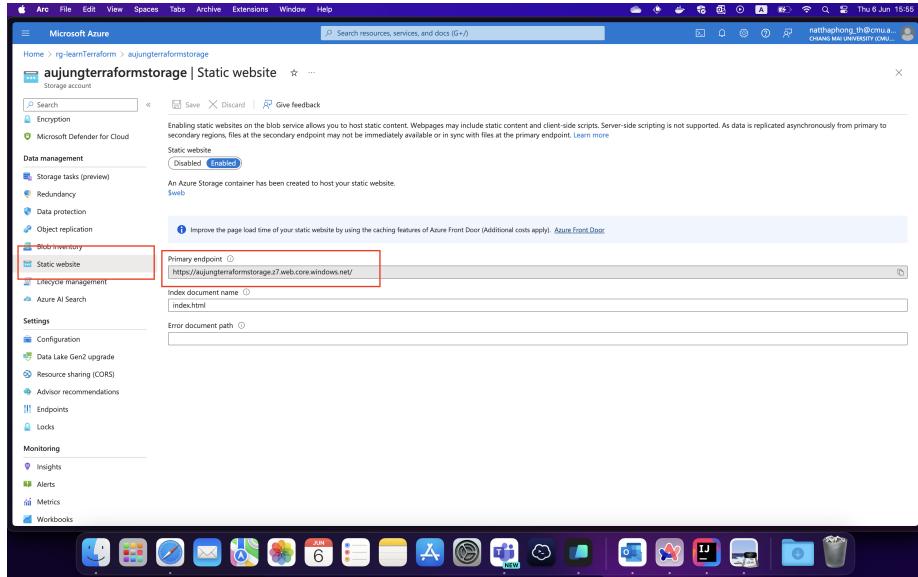
- after that execute this respectively

```
$ terraform init
$ terraform plan
$ terraform apply
```

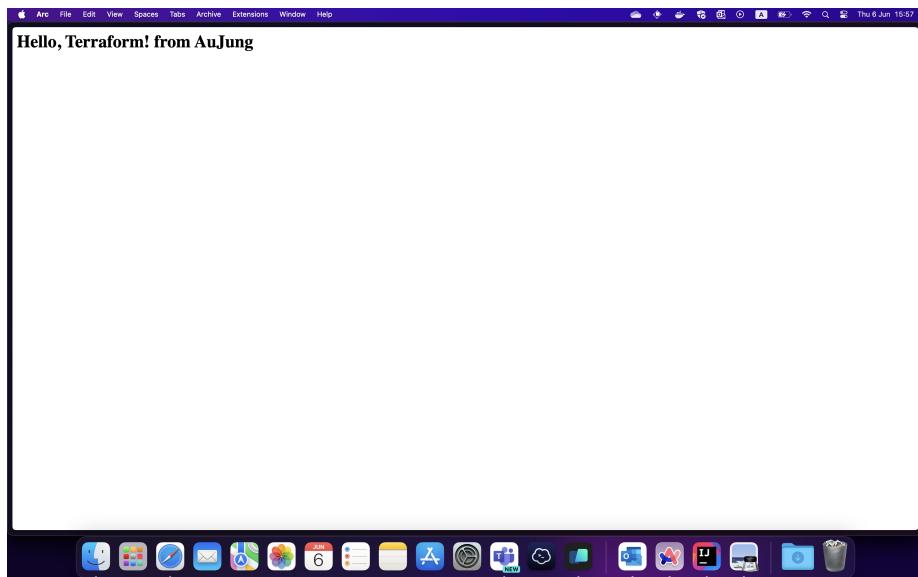
- Everything work fine we will see an resource that was created in azure portal like this.



- Let's see an static web that we created via



- the result on that web should be this.



Create Vm and setup docker with terraform

```
#main.tf
```

```

provider "azurerm" {
    features {}
}

# Create a resource group
resource "azurerm_resource_group" "rg" {
    location = "East Asia"
    name     = "aujung-rg"
}

# Create a virtual network
resource "azurerm_virtual_network" "vnet" {
    name       = "aujung-vnet"
    address_space = ["10.0.0.0/16"]
    location   = azurerm_resource_group.rg.location
    resource_group_name = azurerm_resource_group.rg.name
}

# Create a subnet
resource "azurerm_subnet" "subnet" {
    name           = "aujung-subnet"
    resource_group_name = azurerm_resource_group.rg.name
    virtual_network_name = azurerm_virtual_network.vnet.name
    address_prefixes      = ["10.0.0.0/24"]

}

# Create a public IP
resource "azurerm_public_ip" "ip" {
    location          = azurerm_resource_group.rg.location
    name              = "aujung-ip"
    resource_group_name = azurerm_resource_group.rg.name
    allocation_method = "Dynamic"
}

# Create a network interface

```

```

,
resource "azurerm_network_interface" "nic" {
    location          = azurerm_resource_group.rg.location
    name              = "aujung-nic"
    resource_group_name = azurerm_resource_group.rg.name

    ip_configuration {
        name                = "aujung-ipconfig"
        subnet_id           = azurerm_subnet.subnet.id
        private_ip_address_allocation = "Dynamic"
        public_ip_address_id = azurerm_public_ip.ip.id
    }
}

# Setup inbound security rules
resource "azurerm_network_security_group" "nsg" {
    location          = azurerm_resource_group.rg.location
    name              = "aujung-nsg"
    resource_group_name = azurerm_resource_group.rg.name

    security_rule {
        name          = "SSH"
        priority      = 1001
        direction     = "Inbound"
        access        = "Allow"
        protocol      = "Tcp"
        source_port_range = "*"
        destination_port_range = "22"
        source_address_prefix = "*"
        destination_address_prefix = "*"
    }

    security_rule {
        name          = "HTTP"
        priority      = 1002
        direction     = "Inbound"
        access        = "Allow"
        protocol      = "Tcp"
        source_port_range = "*"
    }
}

```

```

        ,
        destination_port_range      = "80"
        source_address_prefix       = "*"
        destination_address_prefix = "*"
    }
}

# Apply the network security group to vm's network interface
resource "azurerm_network_interface_security_group_association" "vm_nic" {
    network_interface_id      = azurerm_network_interface.nic.id
    network_security_group_id = azurerm_network_security_group.id
}

# Create a vm
resource "azurerm_linux_virtual_machine" "vm" {
    admin_username      = "aujung"
    location           = azurerm_resource_group.rg.location
    name               = "aujung-vm"
    network_interface_ids = [azurerm_network_interface.nic.id]
    resource_group_name = azurerm_resource_group.rg.name
    size               = "Standard_B1s"

    admin_ssh_key {
        public_key = file("~/ssh/id_rsa.pub")
        username   = "aujung"
    }

    os_disk {
        caching          = "ReadWrite"
        storage_account_type = "Standard_LRS"
        disk_size_gb = 30
    }

    source_image_reference {
        publisher = "Canonical"
        offer     = "0001-com-ubuntu-server-jammy"
        sku       = "22_04-lts"
        version   = "latest"
    }
}

```

```

        ,
    }

}

# setup-docker.tf

# Install Docker on the virtual machine
resource "null_resource" "install_docker" {
    triggers = {
        vm_id = azurerm_linux_virtual_machine.vm.id
    }

    connection {
        type          = "ssh"
        host          = azurerm_linux_virtual_machine.vm.public_ip_address
        user          = azurerm_linux_virtual_machine.vm.admin_username
        private_key   = file("~/ssh/id_rsa")
    }

    # Remove old Docker packages
    provisioner "remote-exec" {
        inline = [
            "sudo apt-get remove -y docker docker-engine docker.io"
        ]
    }

    # Install Docker
    provisioner "remote-exec" {
        inline = [
            "sudo apt-get update -y",
            "sudo apt-get install -y -o=APT::Get::Assume-Yes=true curl",
            "sudo rm -rf /etc/apt/keyrings",
            "sudo mkdir -p /etc/apt/keyrings",
            "curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg",
            "echo \"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable\" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null"
        ]
    }
}

```

```

        ,

    "sudo apt-get update -y",
    "sudo apt-get install -y -o=APT::Get::Assume-
    Yes=true
    docker-ce
    docker-ce-cli
    containerd.io",
    "sudo usermod -aG docker ${azurerm_linux_virtual_machine.username}"
]
}

# Configure Docker daemon
provisioner "remote-exec" {
  inline = [
    "sudo mkdir -p /etc/docker",
    "sudo tee /etc/docker/daemon.json > /dev/null <<EOF",
    "{\"",
    "  \"log-driver\": \"json-file\",
    \"  \"log-opt\": {",
    "    \"max-size\": \"10m\",
    "    \"max-file\": \"3\",
    "  }",
    "}",
    "EOF",
    "sudo systemctl daemon-reload",
    "sudo systemctl restart docker",
  ]
}
}

# Deploy nginx container

resource "null_resource" "nginx" {
  depends_on = [null_resource.install_docker]

  triggers = {
    vm_id = azurerm_linux_virtual_machine.vm.id
  }
}

```

```
,  
  
connection {  
  type = "ssh"  
  host = azurerm_linux_virtual_machine.vm.public_ip_address  
  user = azurerm_linux_virtual_machine.vm.admin_username  
  private_key = file("~/ssh/id_rsa")  
}  
  
provisioner "remote-exec" {  
  inline = [  
    "sudo docker run -d -p 80:80 --name nginx nginx",  
  ]  
}  
}
```

Grafana & Prometheus

Grafana is an open-source analytics and interactive visualization web application used for monitoring application performance.

- Need data source to displayed and visualized at this time we will use an Prometheus as datasource

Installation

- via docker-compose file

```
version: '3'
services:
  prometheus:
    image: prom/prometheus:latest
    volumes:
      - ./prometheus/prometheus.yml:/etc/prometheus/prometheus.yml
      - ./prometheus_data:/prometheus
    command:
      - --config.file=/etc/prometheus/prometheus.yml
      - --storage.tsdb.path=/prometheus
      - --storage.tsdb.retention=${PROMETHEUS_RETENTION:-2d}
    restart: always
    networks:
      - monitor-net
  node-exporter:
    image: prom/node-exporter:latest
    networks:
      - monitor-net
  grafana:
    image: grafana/grafana:latest
    volumes:
      - ./grafana_data:/var/lib/grafana
    depends_on:
      - prometheus
    ports:
```

```
,
```

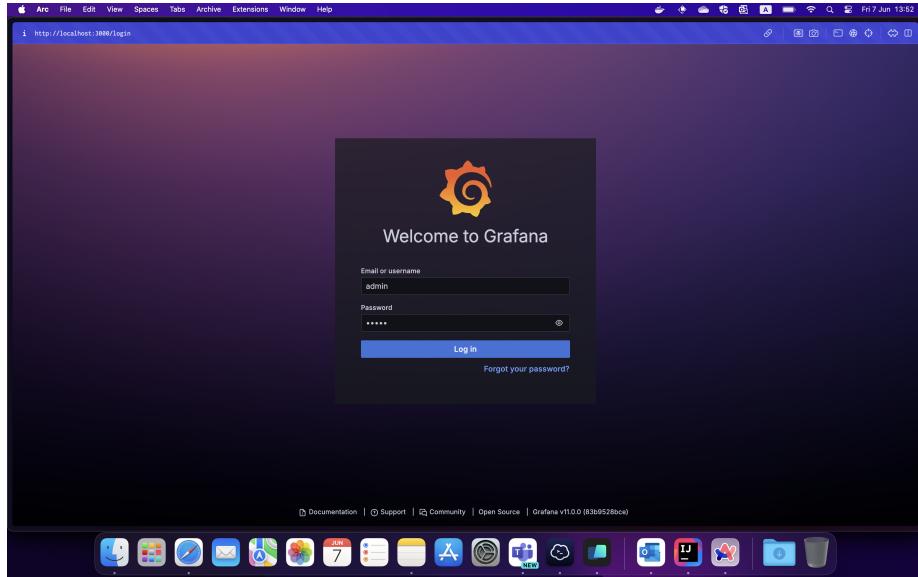
```
    - 3000:3000
networks:
  - monitor-net

networks:
  monitor-net:
    driver: bridge
```

```
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. The default is 15s.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is 15s.

scrape_configs:
  - job_name: 'job-prometheus'
    scrape_interval: 9s
    static_configs:
      - targets: ['prometheus:9090']
  - job_name: "job-node-exporter"
    scrape_interval: 9s
    static_configs:
      - targets: ["node-exporter:9100"]
```

- `docker-compose up -d`



- After, we got grafana webpage let create some dash board to minitor you devices
 - Add data sourcce
 - Click on connection
 - find Prometheus options, then click add new data source
 - connection url : <http://prometheus:9090> (permetheus come from container name in docker compose file)
 - Click Save & Test
 - Go back to dashboard menu
 - Click new → import
 - go to this website to choose some build in dashboard.
<https://grafana.com/grafana/dashboards/1860-node-exporter-full/>
 - Copy dashboard id then paste it on grafana web site then click load.



ELK Stack

Visualize Apache Logs With Elastic Stack on Ubuntu

- Update system package and install java runtime

```
$ sudo apt-get update && sudo apt-get upgrade  
  
$ sudo apt-get install default-jre-headless
```

- Install the Elastic APT Repository → this package repositories contain all of the necessary packages include Elasticsearch, log stash and kibana

```
$ wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch  
sudo apt-key add -  
$ echo "deb https://artifacts.elastic.co/packages/7.x/apt ."  
sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list  
$ sudo apt-get update
```

- Install Elasticsearch

1. `sudo apt-get install elasticsearch`

2. Set up heap size for JVM

```
File: /etc/elasticsearch/jvm.options  
  
-Xmx(one-quarter of your server's available memory)  
  
Ex.  
-Xmx1g (if host have 4g of memory)
```

3. Start elasticsearch

```
$ sudo systemctl enable elasticsearch  
$ sudo systemctl start elasticsearch
```

4. confirm that the Elasticsearch API is available

```
$ curl localhost:9200

response should be :

{

  "name" : "vm-module",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "TZ0bzoEGSaKuUaktiQQLQQ",
  "version" : {
    "number" : "7.17.21",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "d38e4b028f4a9784bb74de339ac1b877e2d",
    "build_date" : "2024-04-26T04:36:26.745220156Z",
    "build_snapshot" : false,
    "lucene_version" : "8.11.3",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

- Install logstash and cabana
 - `sudo apt-get install logstash`
 - `sudo apt-get install kibana`
- Configure the Elastic Stack
 - Overwrite Elasticsearch setup from create multiple shard to use only one shard.
 - Create a temporary JSON

```
{
  "index_patterns": ["*"],
  "template": {
    "settings": {
      "index": {
        "number_of_shards": 1,
```

```
        "number_of_replicas": 0
    }
}
}
}
```

- Use `curl` to create an index template with these settings that is applied to all indices created hereafter:

```
$ curl -XPUT -H'Content-type: application/json' http://localhost:9200/_index_template/defaults -d @template.json
```

- Configure Logstash

collect Apache access logs, Logstash must be configured to watch any necessary files and then process them, eventually sending them to Elasticsearch.

- Set up heap size for logstash at `/etc/logstash/jvm.options`
- Create the following Logstash configuration

```
File: /etc/logstash/conf.d/apache.conf

input {
  file {
    path => '/var/www/*/logs/access.log'
    start_position => 'beginning'
  }
}

filter {
  grok {
    match => { "message" => "%{COMBINEDAPACHELOG}" }
  }
}

output {
```

```
        elasticsearch { }  
    }
```

- Start all service

```
$ sudo systemctl enable logstash  
$ sudo systemctl start logstash  
  
$ sudo systemctl enable kibana  
$ sudo systemctl start kibana
```

- Install apache2 for test log collecting

- `sudo apt-get install apache2`
- change logs dir to appropriate with logstash configuration

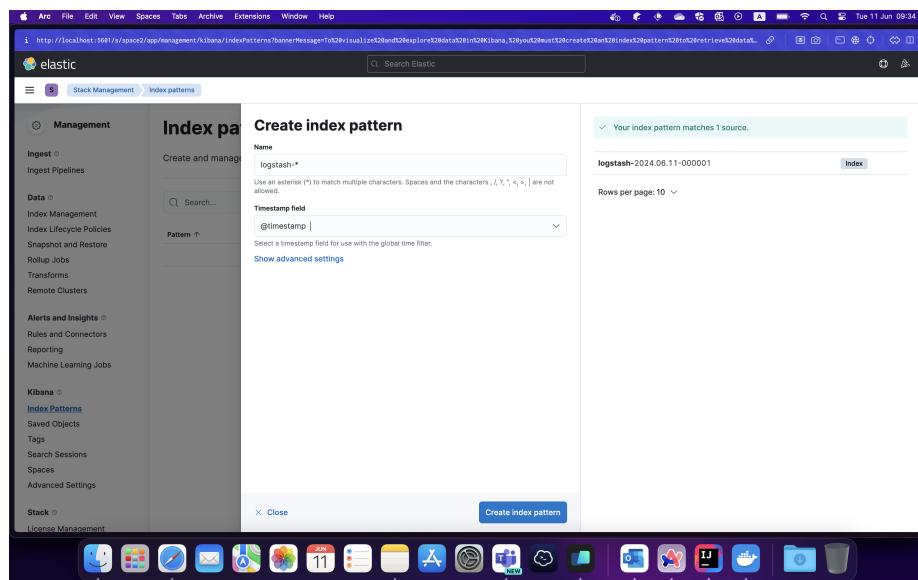
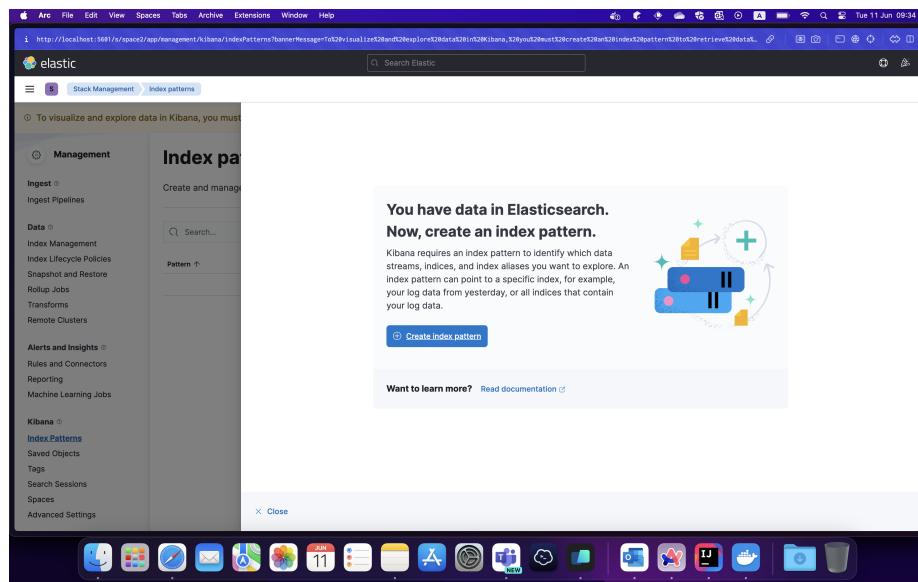
```
$ vim /etc/apache2/sites-available/000-default.conf  
  
ErrorLog /var/www/html/logs/error.log  
CustomLog /var/www/html/logs/access.log combined  
  
$ sudo mkdir /var/www/html/logs  
  
$ sudo systemctl restart apache2
```

- For testing we need some logs data so that we need to enter apache2 landing page to get logs via : `for i in `seq 1 5` ; do curl localhost ; sleep 0.2 ; done`

Watching Logs

- By default, Kibana binds to the local address `127.0.0.1`. This only permits connections that originate from localhost. This is recommended in order to avoid exposing the dashboard to the public internet. use `ssh` command can forward the port to your workstation.
 - `ssh -N -L 5601:localhost:5601 Username@<IP address of kibana>`
- Open Kibana in your browser at <http://localhost:5601>.

- Create an index pattern to collect logs data



- Now when back to Discover page again will see

