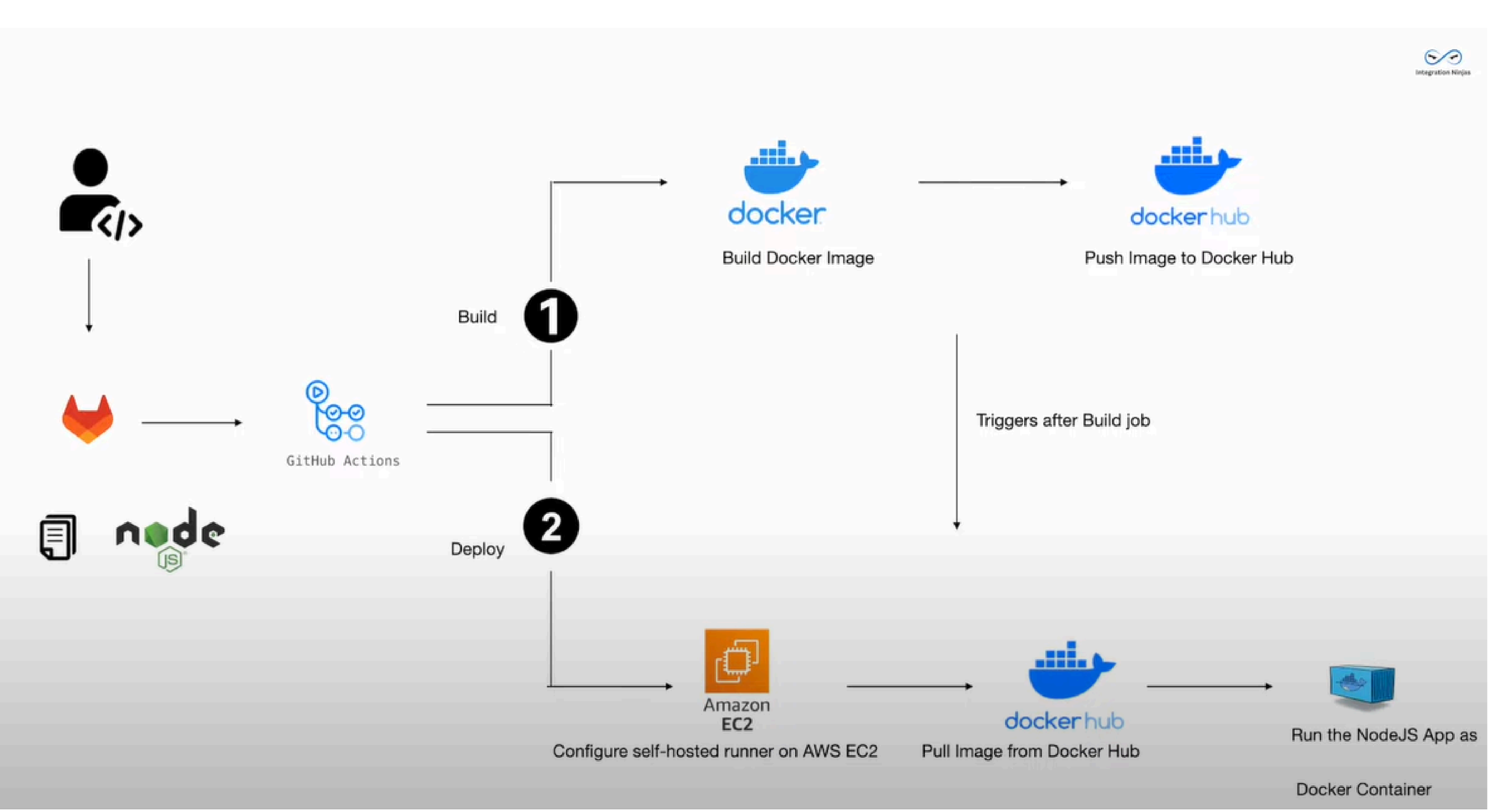
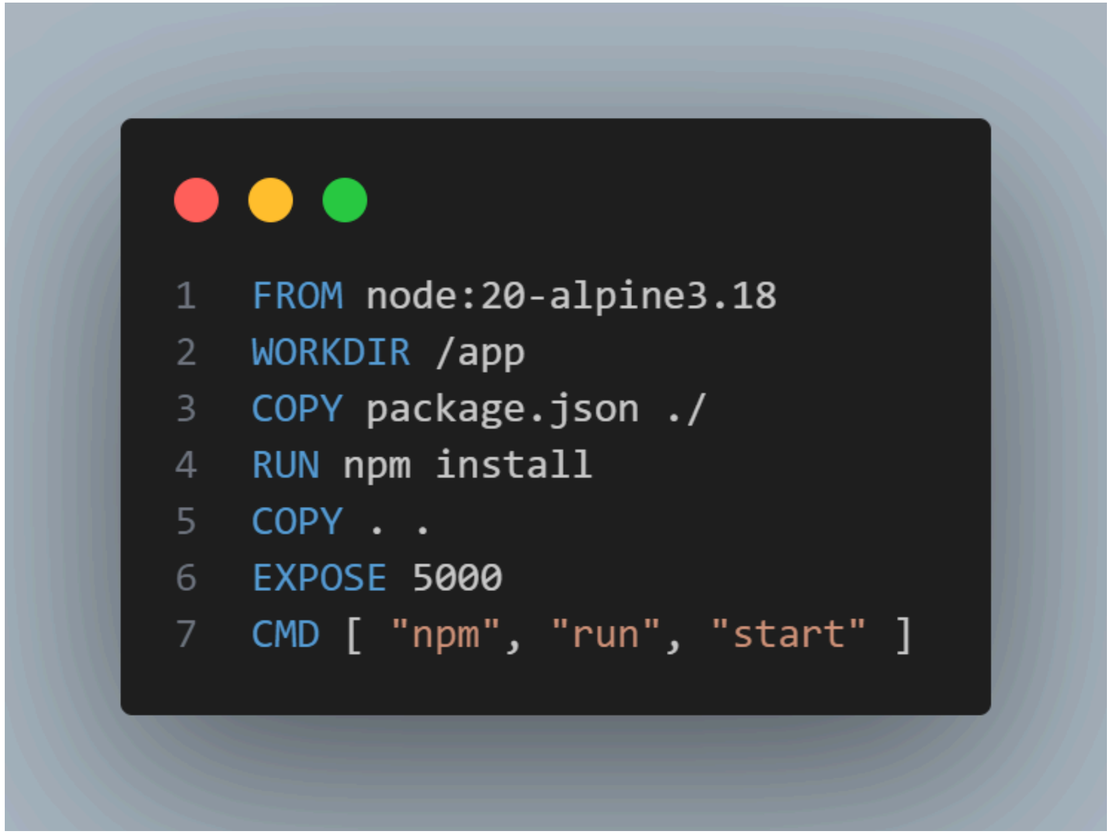
Deploy NodeJS Application to AWS EC2 Using GitLab CI


Integration Ninjas
docker
Build Docker Image
docker hub
Push Image to Docker Hub
Build
1
GitHub Actions
Triggers after Build job
node
Deploy
2
Amazon
EC2
Configure self-hosted runner on AWS EC2
docker hub
Pull Image from Docker Hub
Run the NodeJS App as
Docker Container

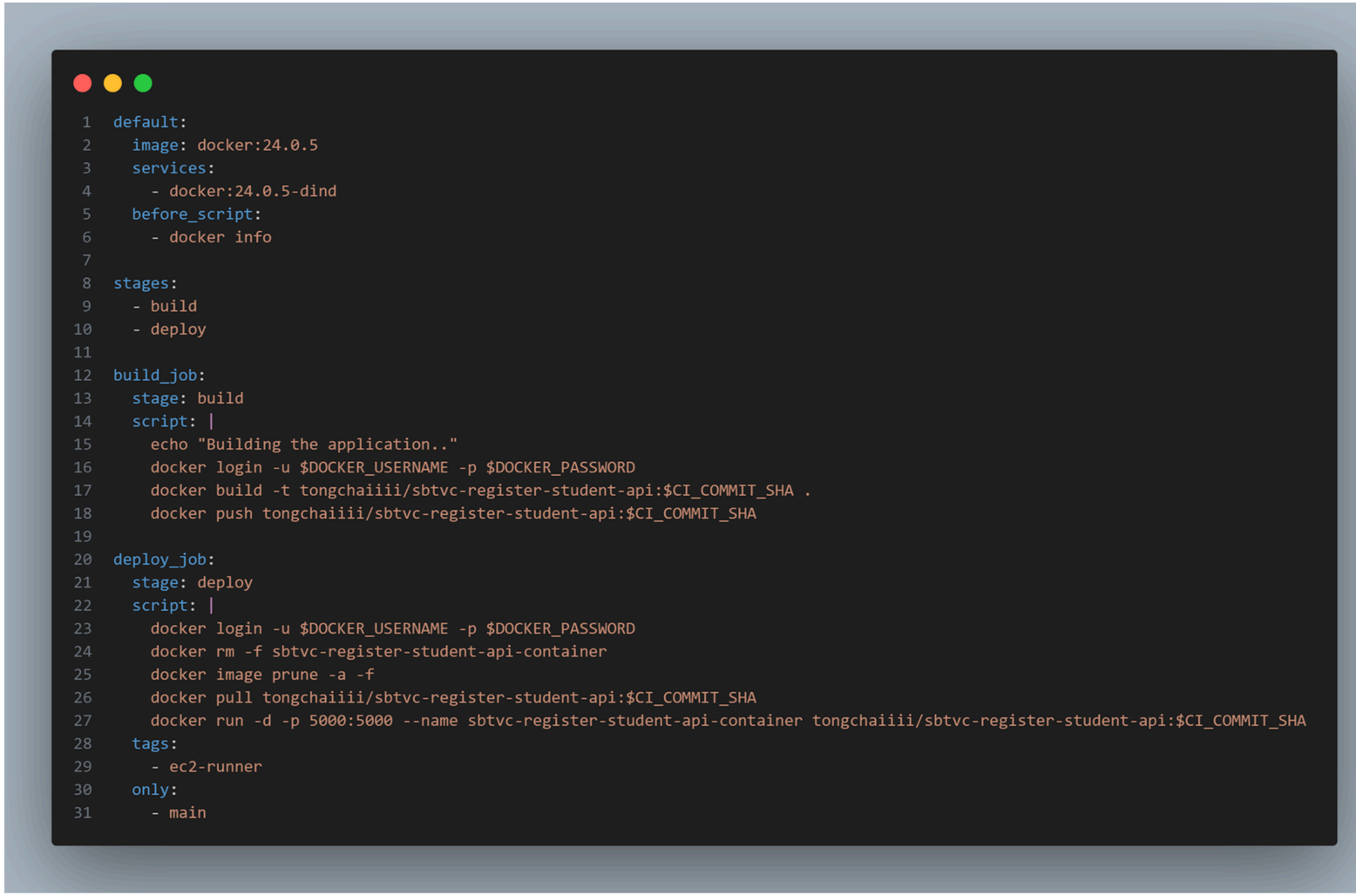
ให้เตรียมไฟล์ Dockerfile ใน Project

```
FROM node:20-alpine3.18
WORKDIR /app
COPY package.json ./
RUN npm install
COPY . .
EXPOSE 5000
CMD [ "npm", "run", "start" ]
```



ให้เตรียมไฟล์ .gitlab-ci.yml ใน Project

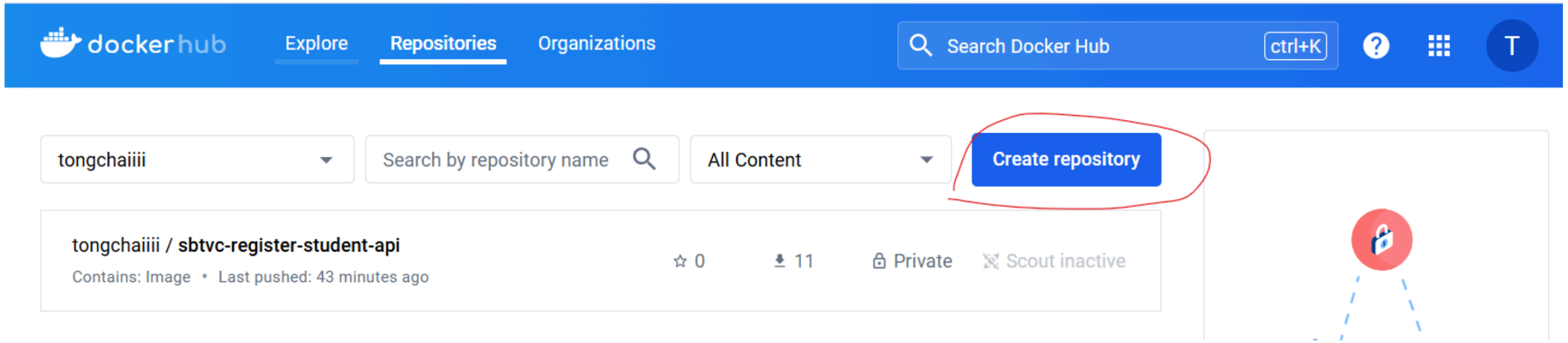
```yaml
default:
  image: docker:24.0.5
  services:
    - docker:24.0.5-dind
  before_script:
    - docker info

stages:
  - build
  - deploy

build_job:
  stage: build
  script: |
    echo "Building the application.."
    docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD
    docker build -t tongchaiiii/sbtvc-register-student-api:$CI_COMMIT_SHA .
    docker push tongchaiiii/sbtvc-register-student-api:$CI_COMMIT_SHA

deploy_job:
  stage: deploy
  script: |
    docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD
    docker rm -f sbtvc-register-student-api-container
    docker image prune -a -f
    docker pull tongchaiiii/sbtvc-register-student-api:$CI_COMMIT_SHA
    docker run -d -p 5000:5000 --name sbtvc-register-student-api-container tongchaiiii/sbtvc-register-student-api:$CI_COMMIT_SHA
  tags:
    - ec2-runner
  only:
    - main #จะ deploy เฉพาะ branch main
```



การตั้งค่า $DOCKER_USERNAME $DOCKER_PASSWORD

docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD

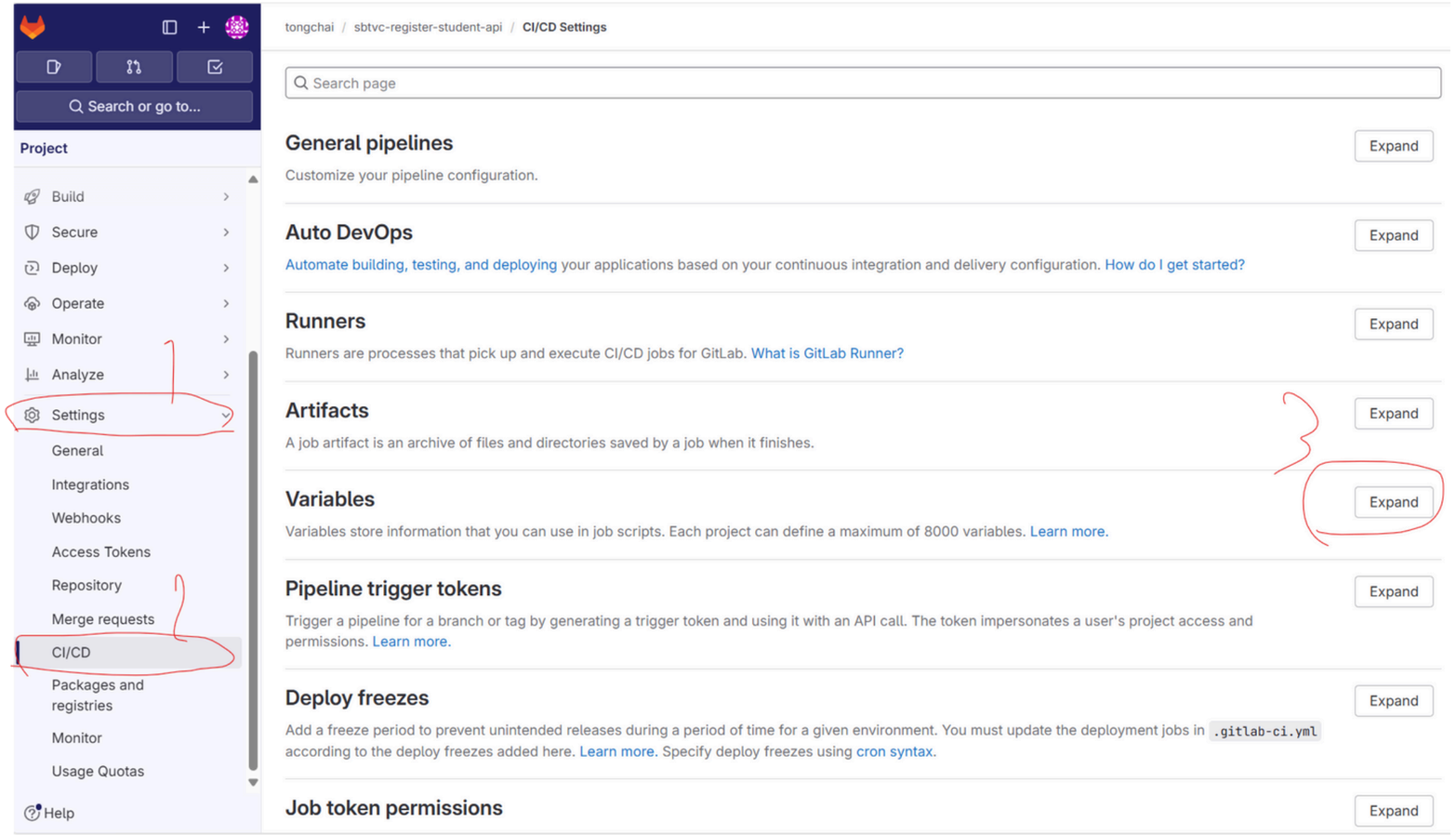
1.เข้าไปที่ docker hub สมัครสมาชิก https://hub.docker.com/
2.สร้าง respository



การตั้งค่า $DOCKER_USERNAME $DOCKER_PASSWORD

docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD

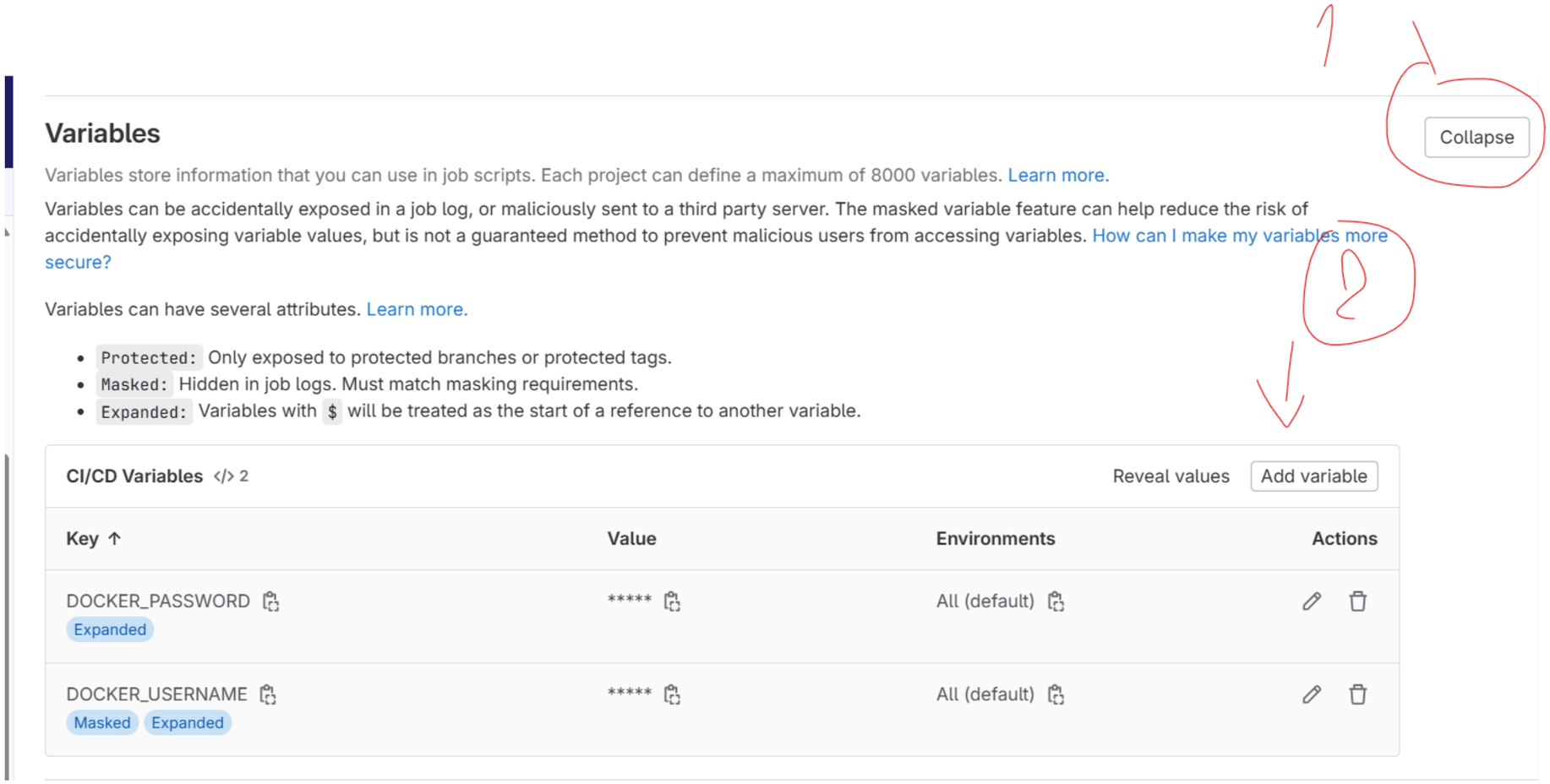
3. เมื่อสร้างสำเร็จก็จะมี username และ password ให้ไปตั้งค่าที่ส่วนของ gitlab



การตั้งค่า $DOCKER_USERNAME $DOCKER_PASSWORD

docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD

4. เมื่อสร้างสำเร็จก็จะมี username และ password ให้ไปตั้งค่าที่ส่วนของ gitlab



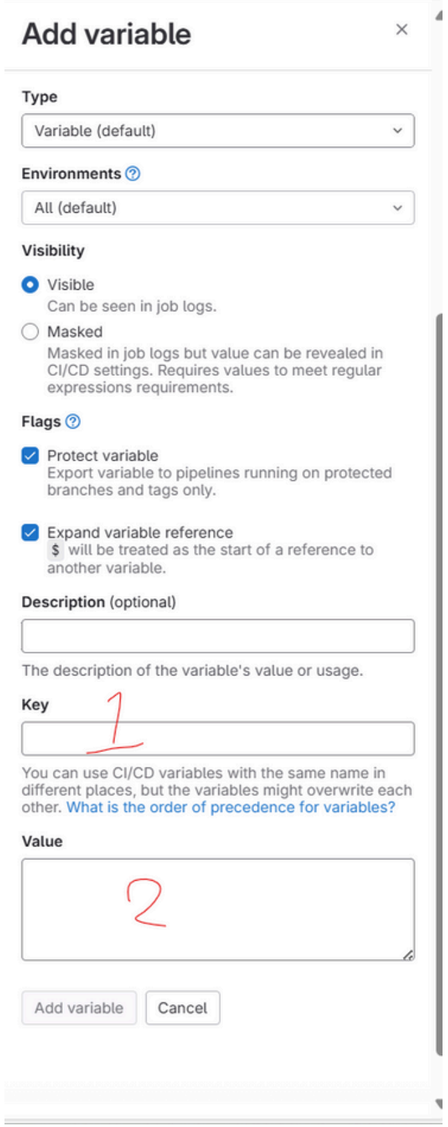
การตั้งค่า $DOCKER_USERNAME $DOCKER_PASSWORD

docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD

5. เมื่อสร้างสำเร็จก็จะมี username และ password ให้ไปตั้งค่าที่ส่วนของ gitlab

5.1 เพิ่มชื่อ key เป็น DOCKER_USERNAME
5.2 เพิ่มชื่อ value เป็น username ของ docker ที่สมัคร

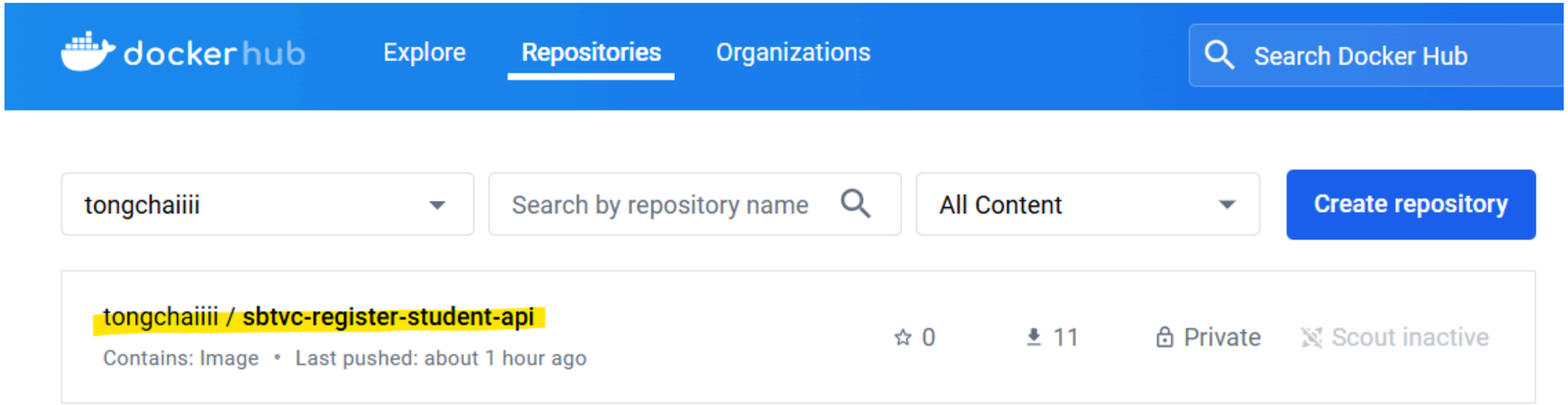
5.3 เพิ่มชื่อ key เป็น DOCKER_PASSWORD
5.4 เพิ่มชื่อ value เป็น password ของ docker ที่สมัคร



การนำข้อมูลจาก Docker hub เข้ามาใช้งานในการตั้งค่าไฟล์

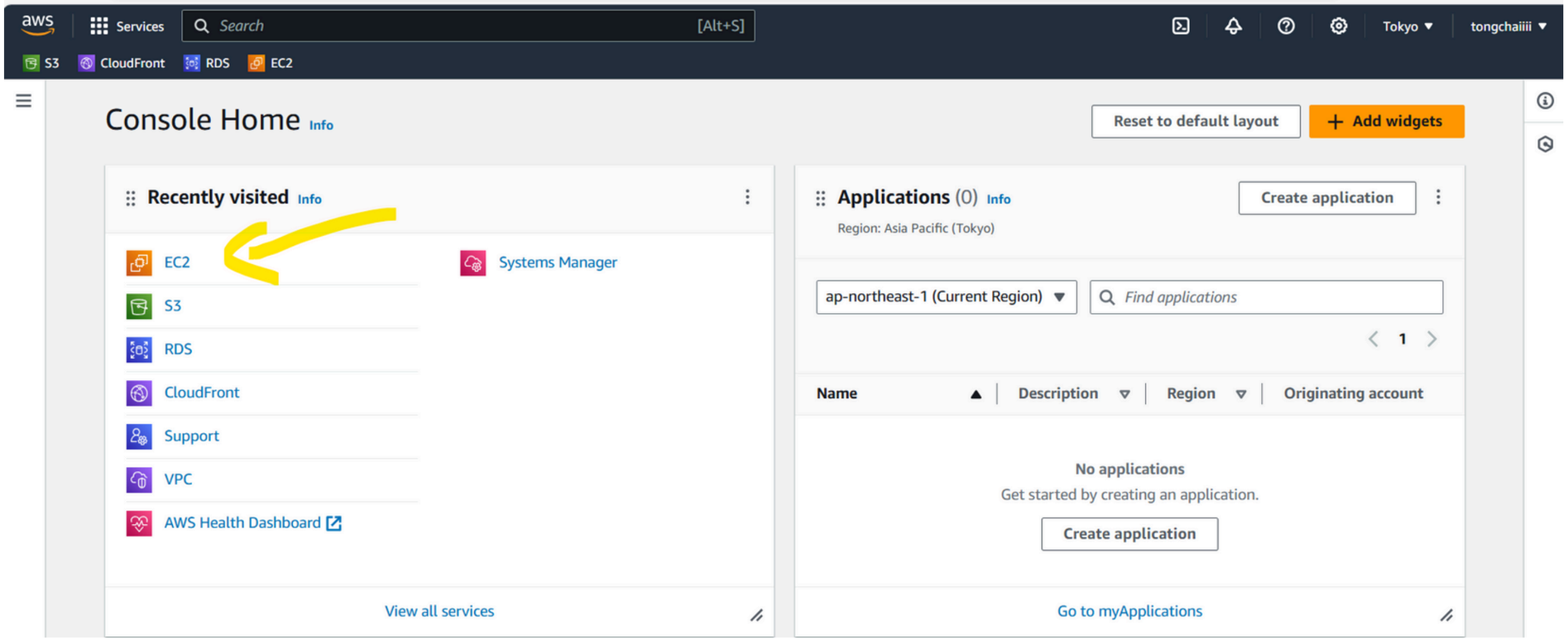
การนำค่าข้อมูลมาใช้งาน tongchaiiii/sbtvc-register-student-api

6. ไปนำข้อมูลชื่อ space และ ชื่อของ repository



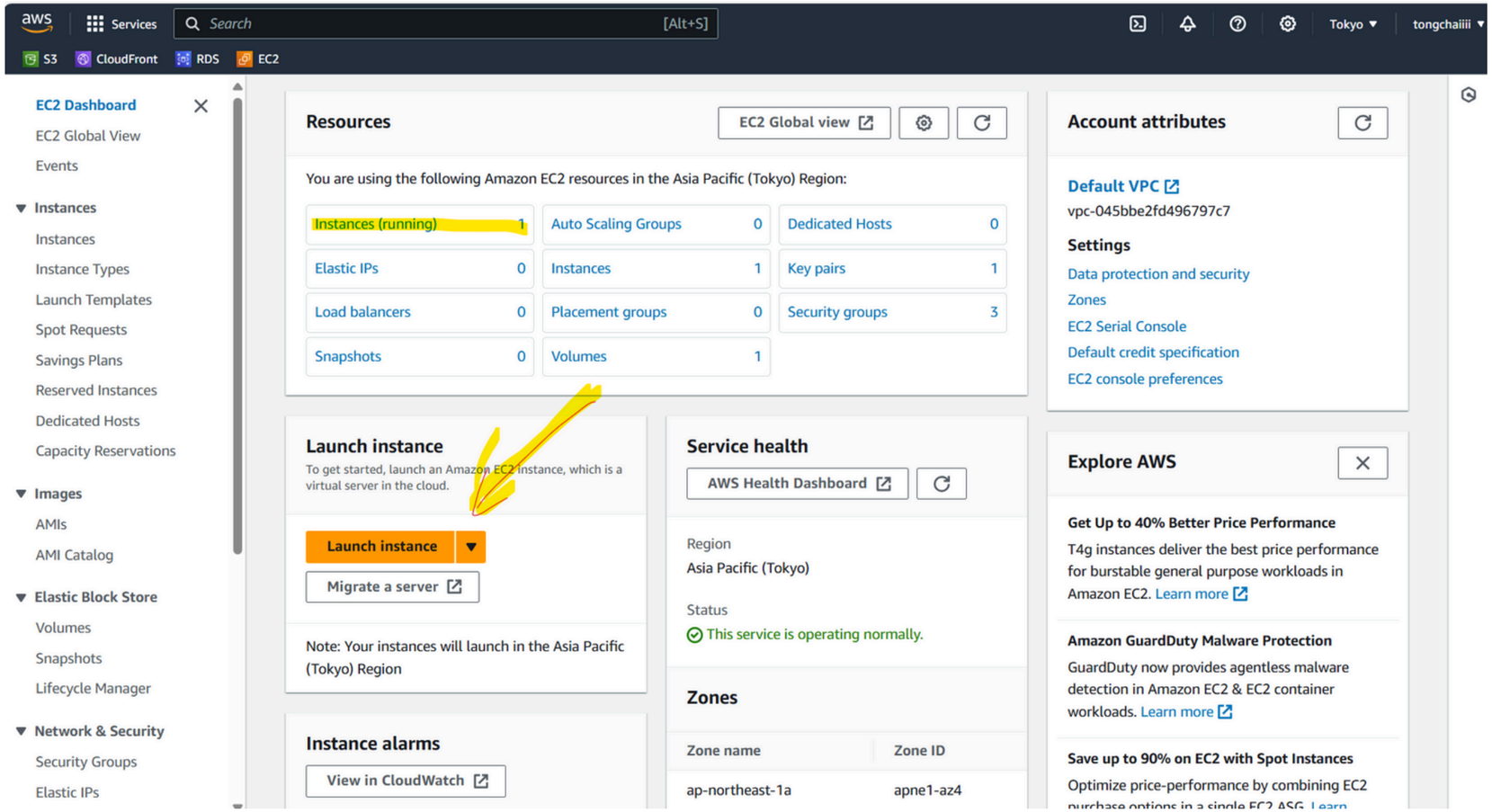
การตั้งค่าใน server ของ AWS

7. login ใน AWS https://aws.amazon.com/marketplace/management/signin?ref_=footer_nav_management_portal
8. เข้าที่ EC2



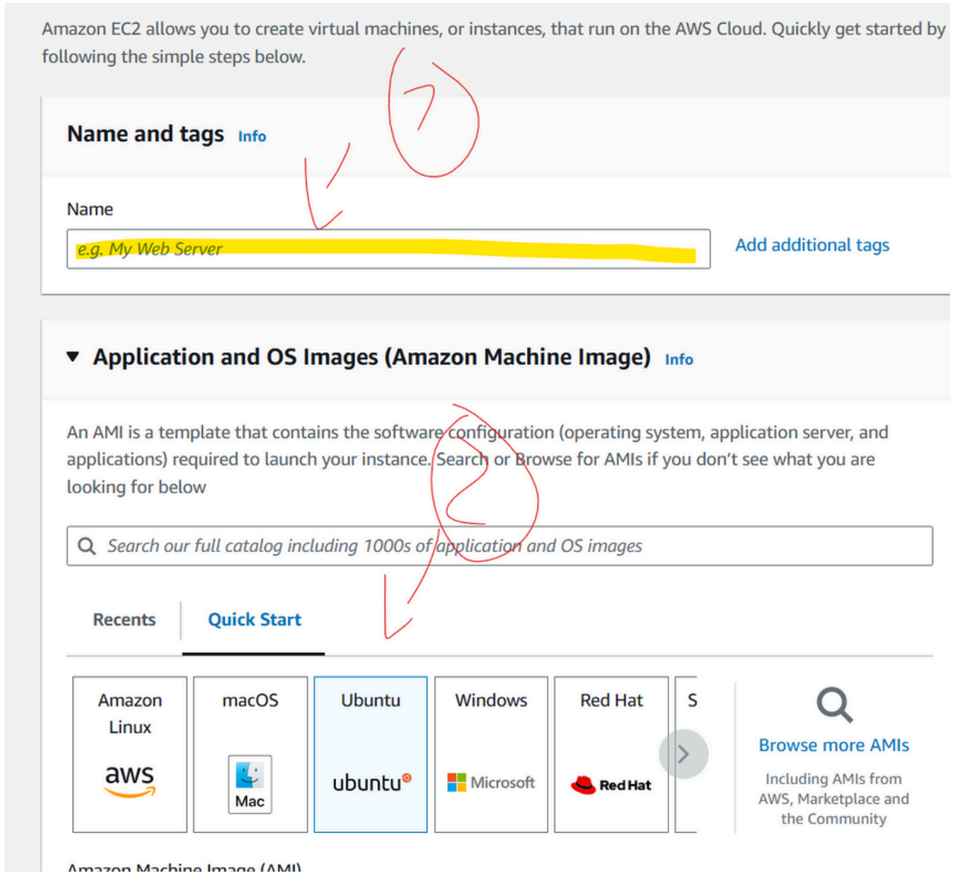
การตั้งค่าใน server ของ AWS

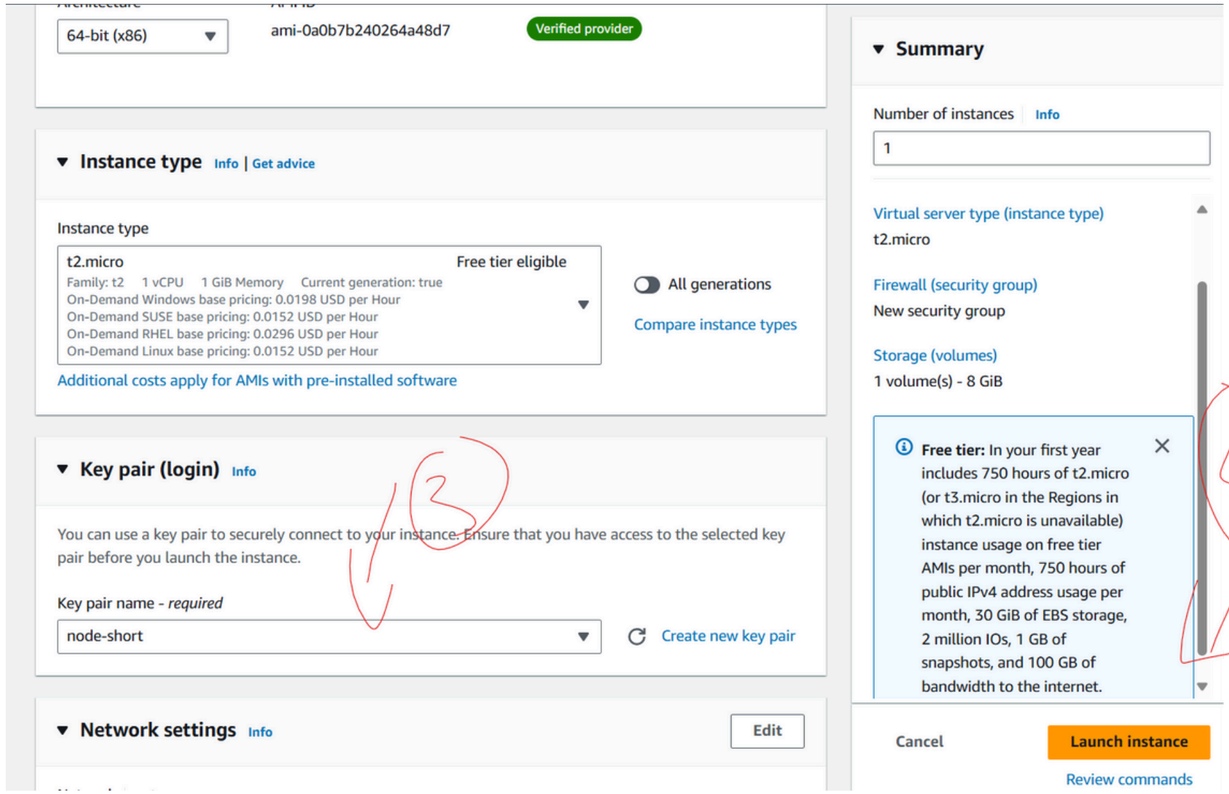
9. คือ ข้อมูลที่ server เรามีอยู่ใน instances (running)
10. สร้าง instances



การตั้งค่าใน server ของ AWS

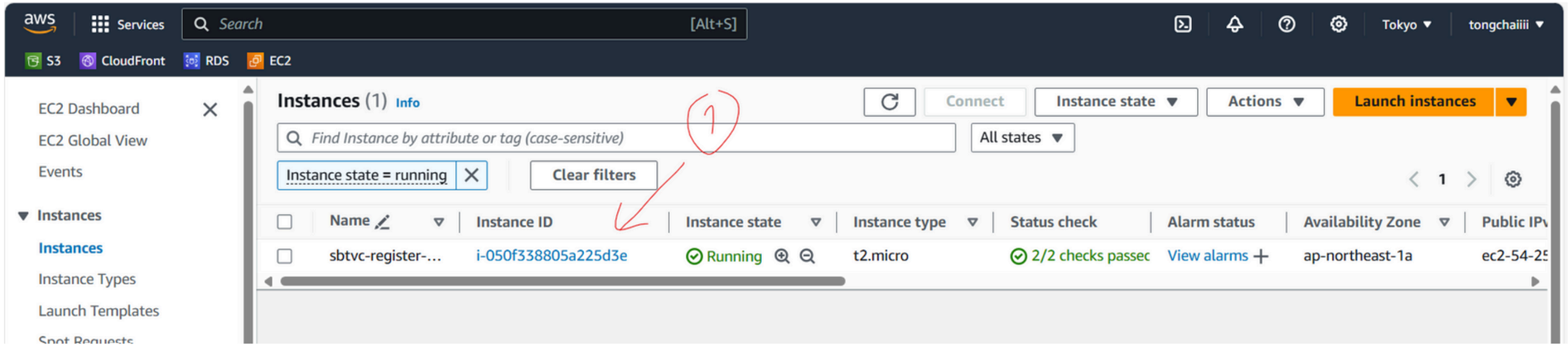
9. คือ ข้อมูลที่ server เรามีอยู่ใน instances (running)
10. สร้าง instances
    1. ตั้งชื่อ โปรเจค
    2. เลือก Server เป็น ubuntu
    3. Key pair เลือกเป็นของเรา ถ้าไม่มีก็ create
    4. Launch instance เพื่อสร้าง





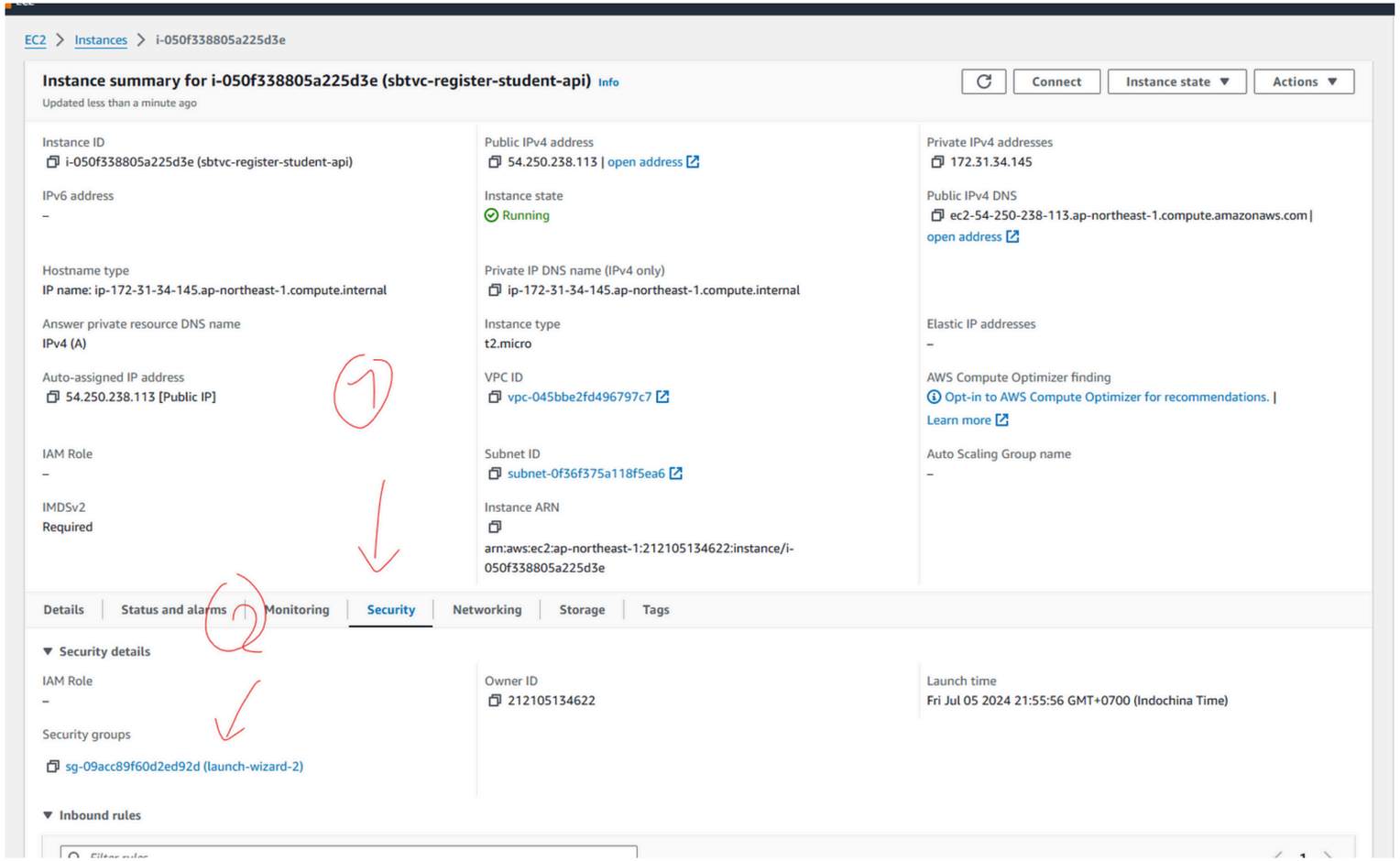
การตั้งค่าใน server ของ AWS

11. .กดเข้าที่ ID ของมัน



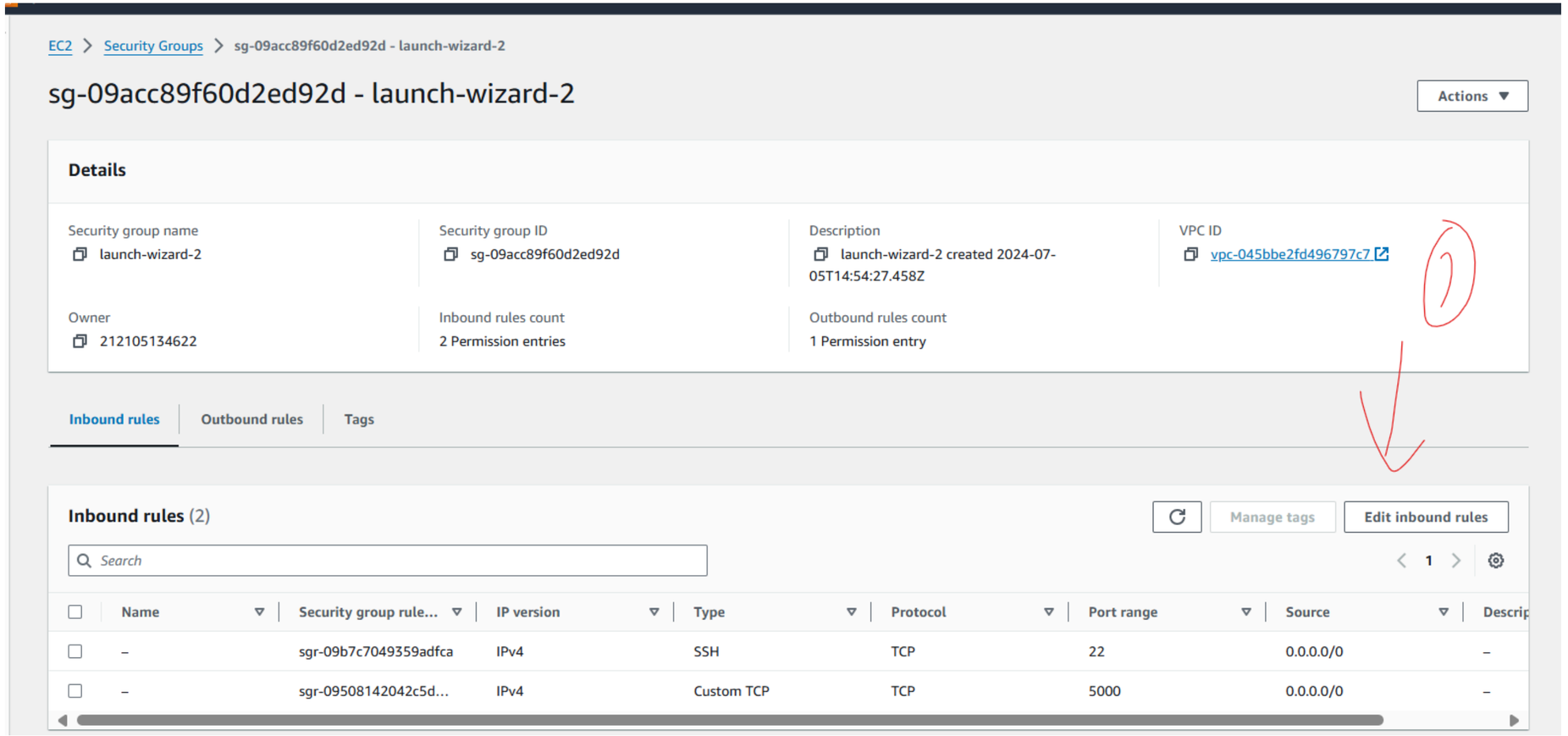
การตั้งค่าใน server ของ AWS

12. เข้าไปตั้งค่าความปลอดภัย



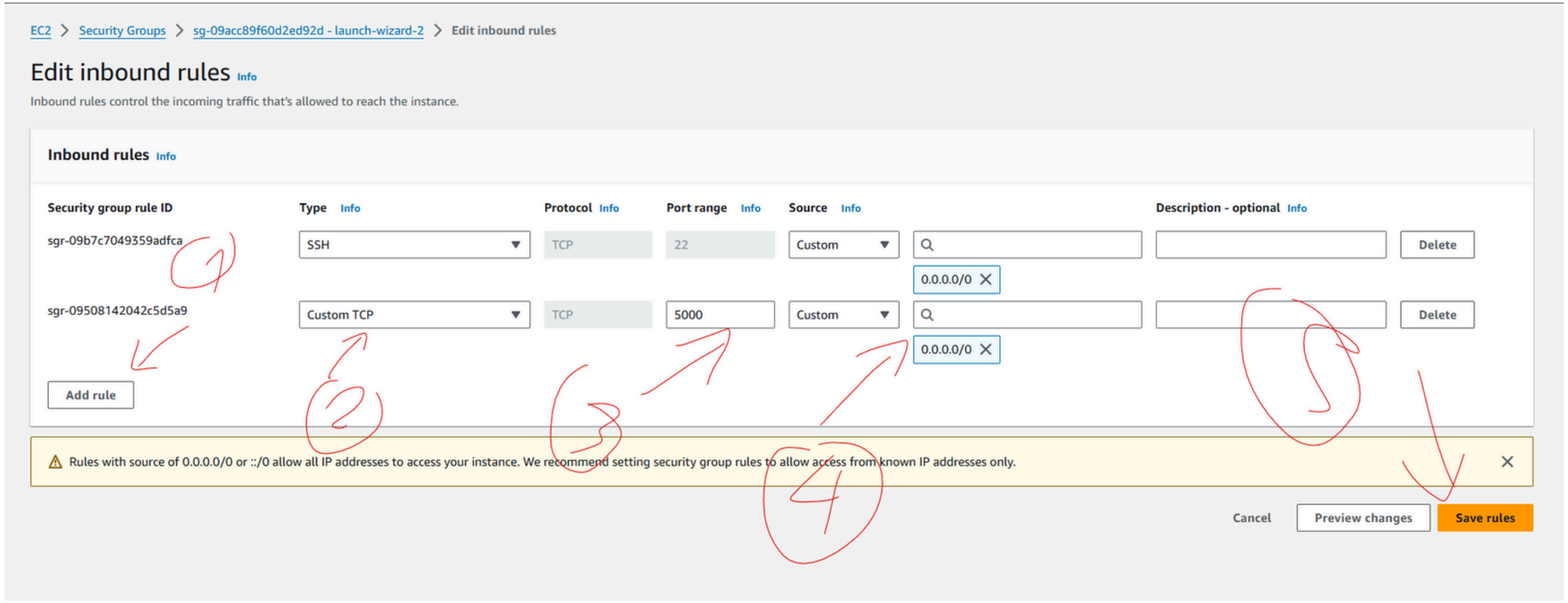
การตั้งค่าใน server ของ AWS

13. เข้าไปที่ Edit Inbound rules



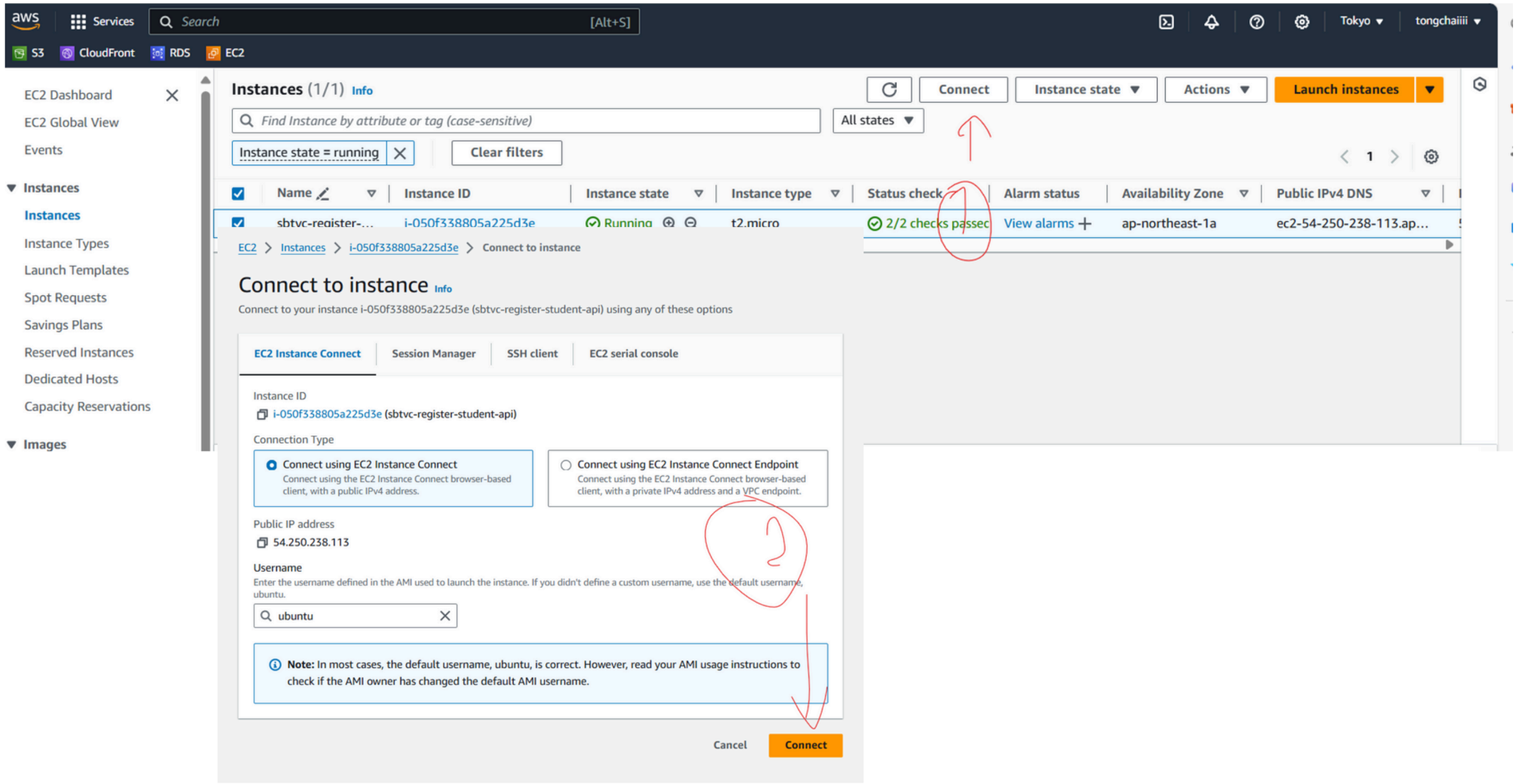
การตั้งค่าใน server ของ AWS

14. เข้าไปที่ Edit Inbound rules
    1. กดที่ Add rule
    2. Type เลือกเป็น Custom TCP
    3. Port range 5000
    4. ตัวของ IP เป็น 0.0.0.0-0
    5. กด Save rules



การเชื่อมต่อไปที่ในการใช้ Docker server ของ AWS

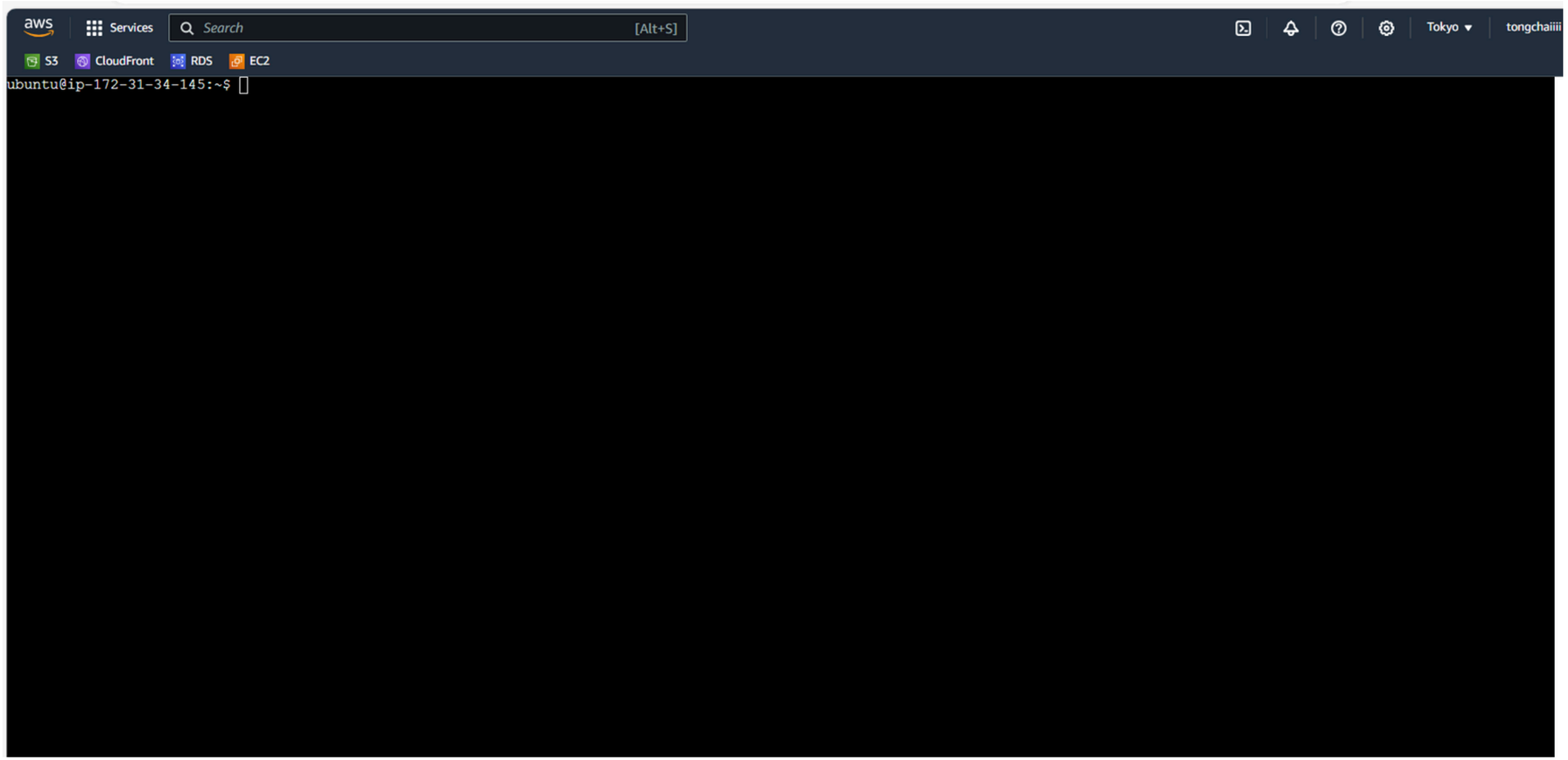
16. ย้อนกลับมาที่หน้าหลักเข้าไปกด ที่ connect และ กด connect อีกทีหนึ่ง และจะได้หน้าจอ command



การเชื่อมต่อไปที่ในการใช้ Docker server ของ AWS

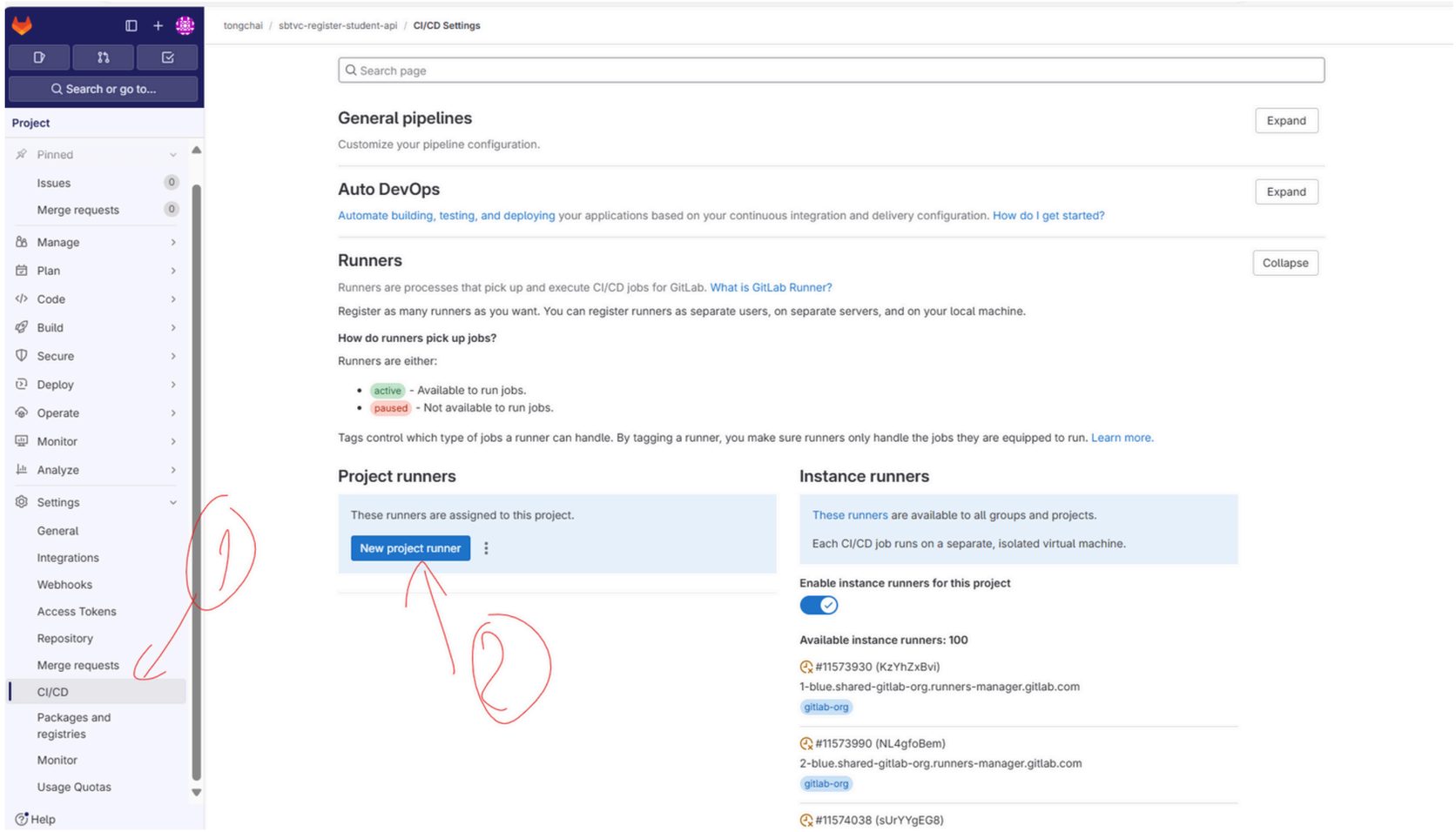
17. ให้ใช้คำสั่งต่อไปนี้ใน command เพื่อติดตั้งการที่เราจะใช้งาน docker

sudo apt-get update && sudo apt-get install docker.io -y && sudo systemctl start docker && sudo chmod 666 /var/run/docker.sock && sudo systemctl enable docker && docker --version



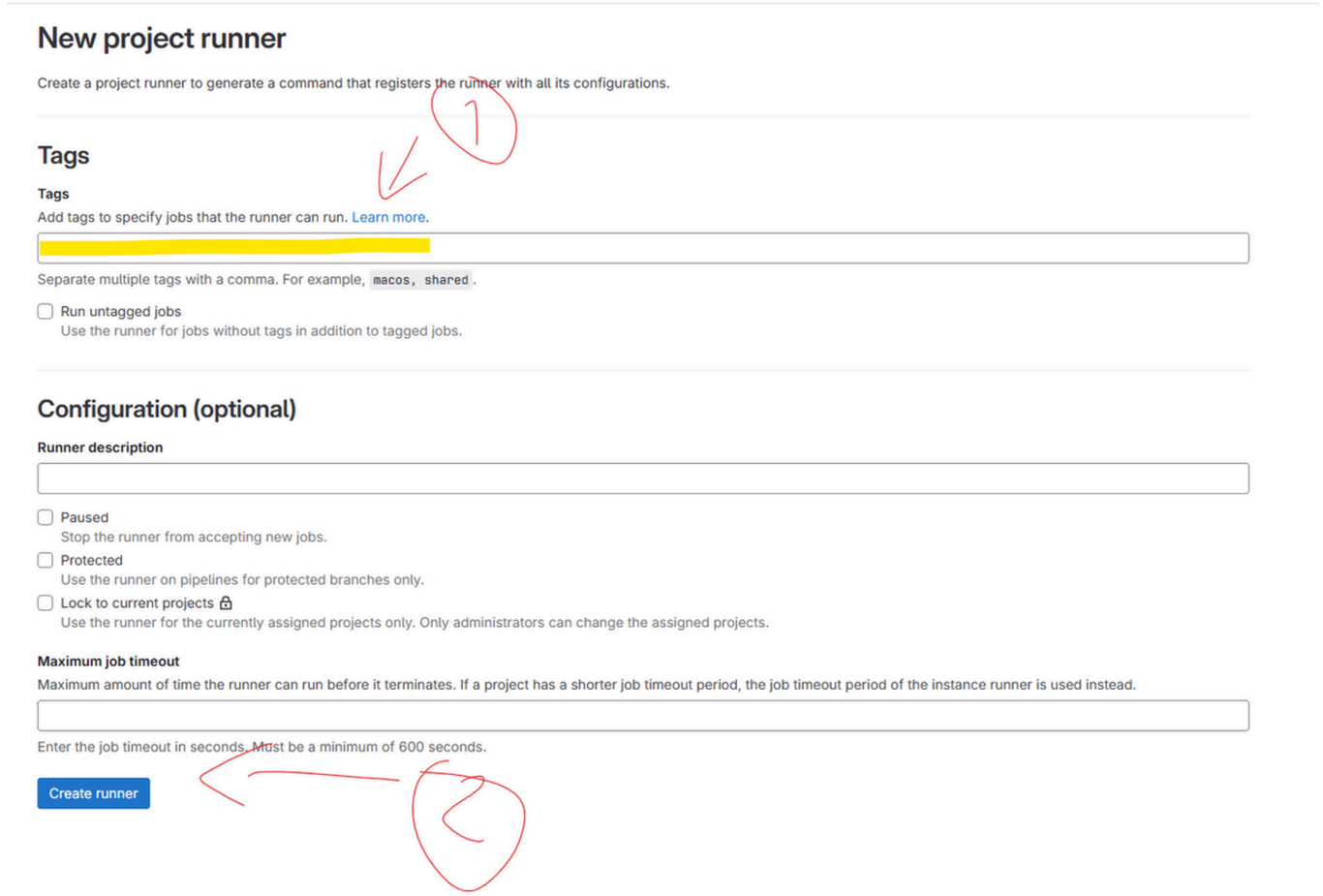
การสร้าง CI/CD เข้าไปยัง Docker

18. ไปตั้งค่า runner CI/CD กดที่คำว่า new project runner



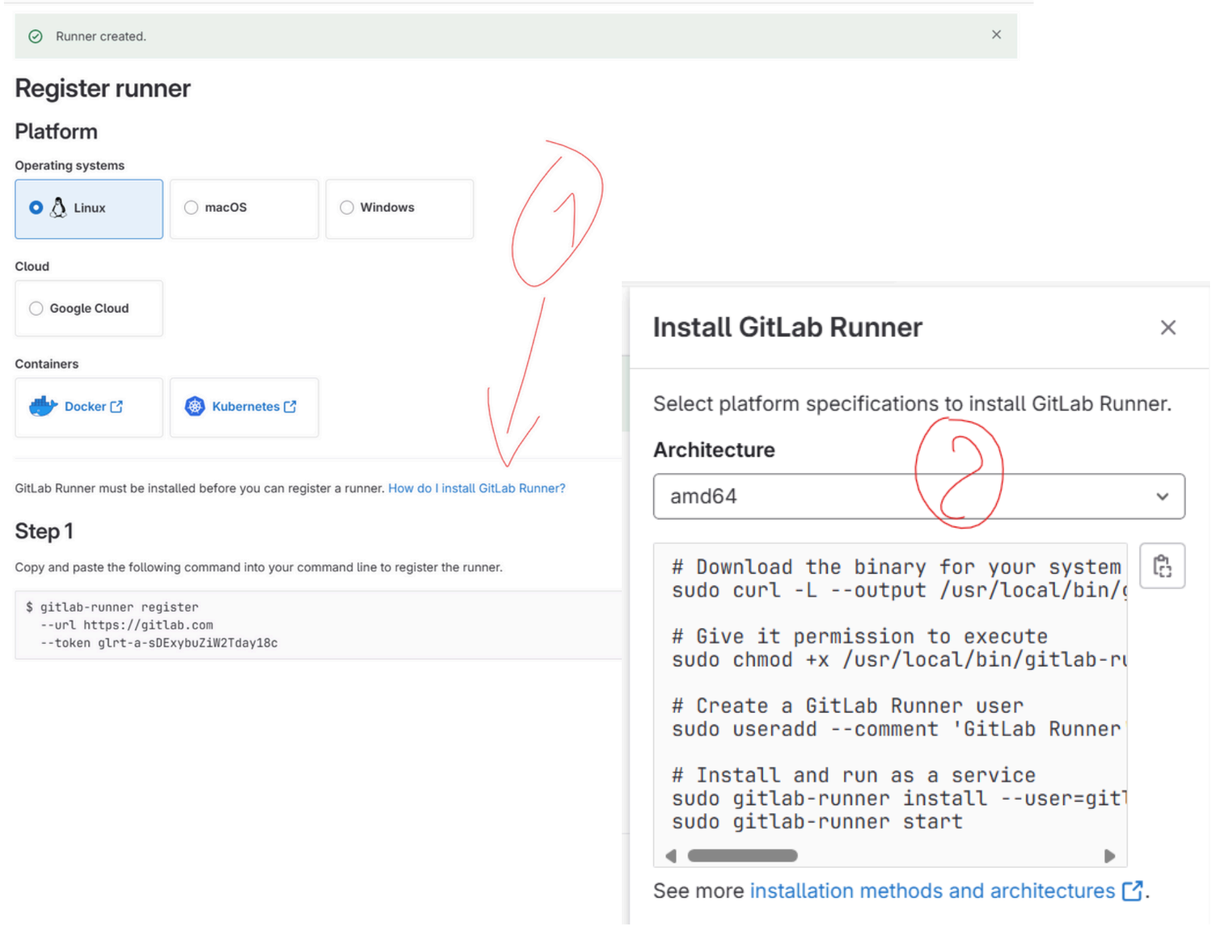
การสร้าง CI/CD เข้าไปยัง Docker

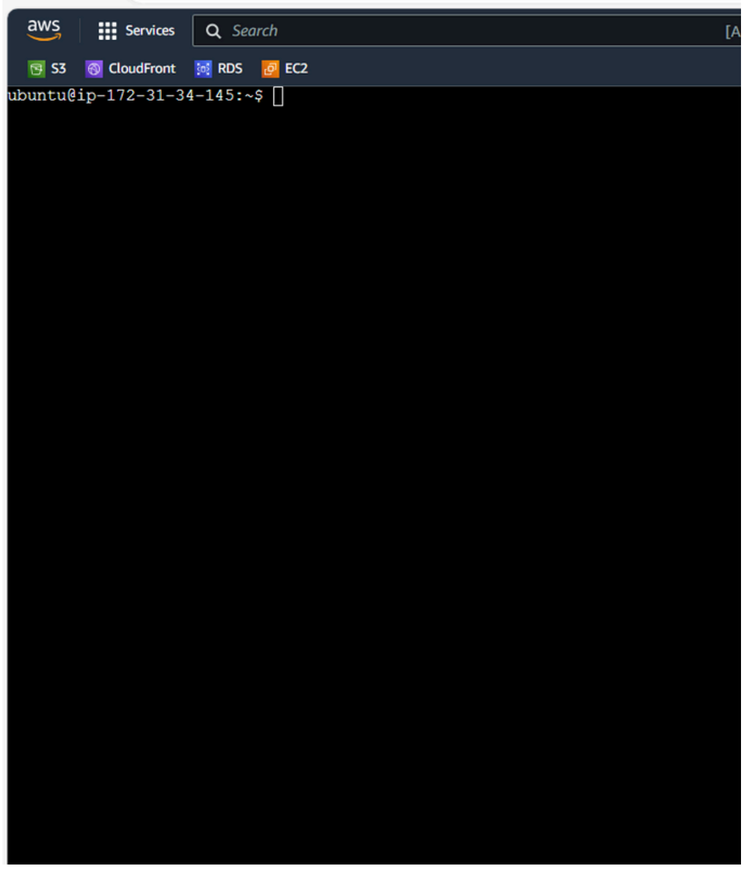
19. ตั้งชื่อ runner และทำการ create runner



การสร้าง CI/CD เข้าไปยัง Docker

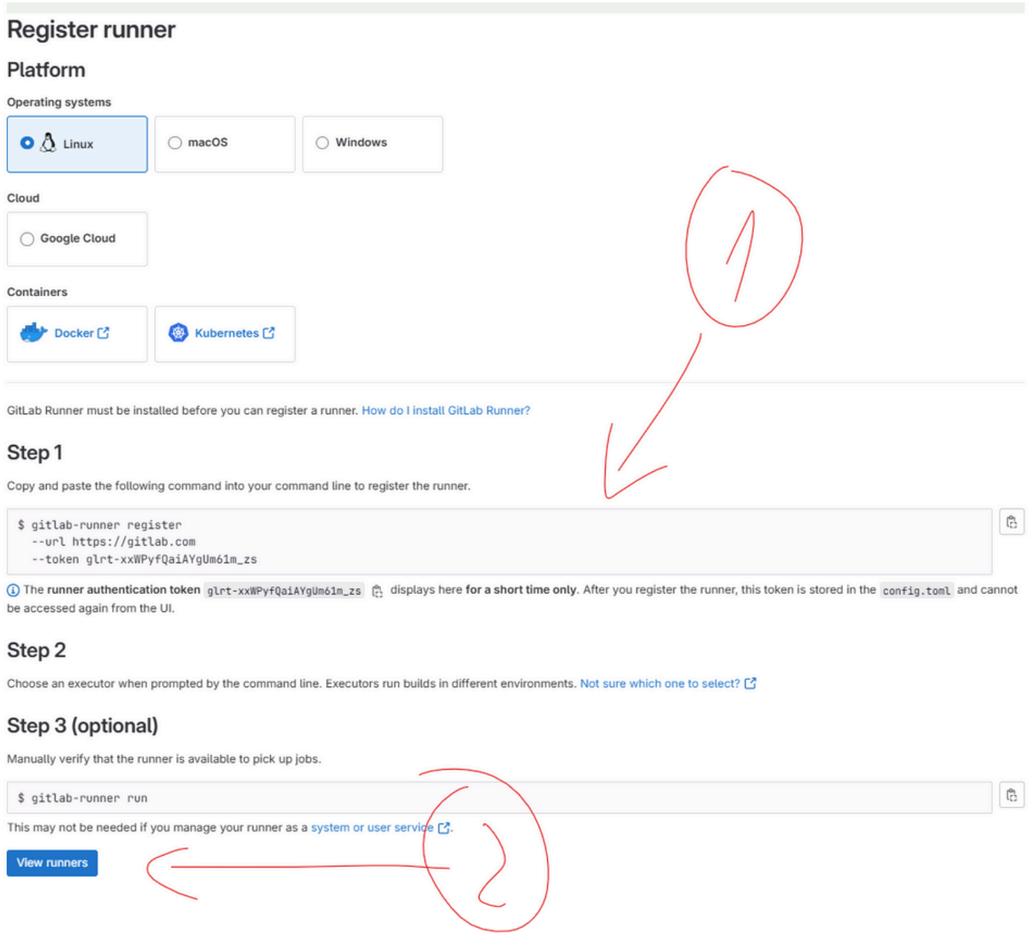
20. กดเลือกเป็น linux และทำการ กด ที่ How do i install Gitlab Runner?
ต่อมา ก็อป install ส่วนที่ 2 ไปที่หน้า command AWS

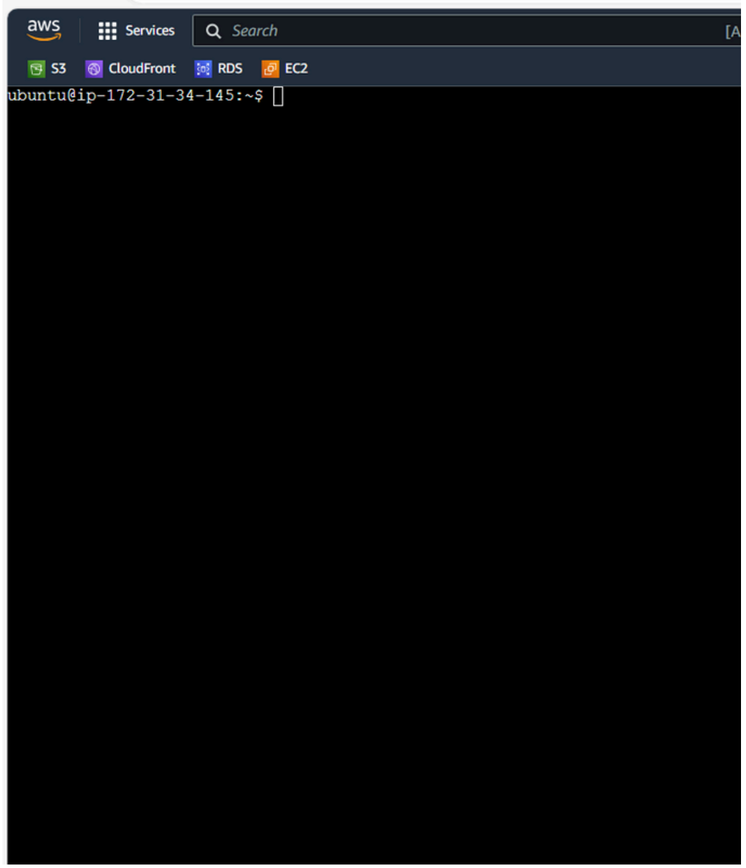




การสร้าง CI/CD เข้าไปยัง Docker

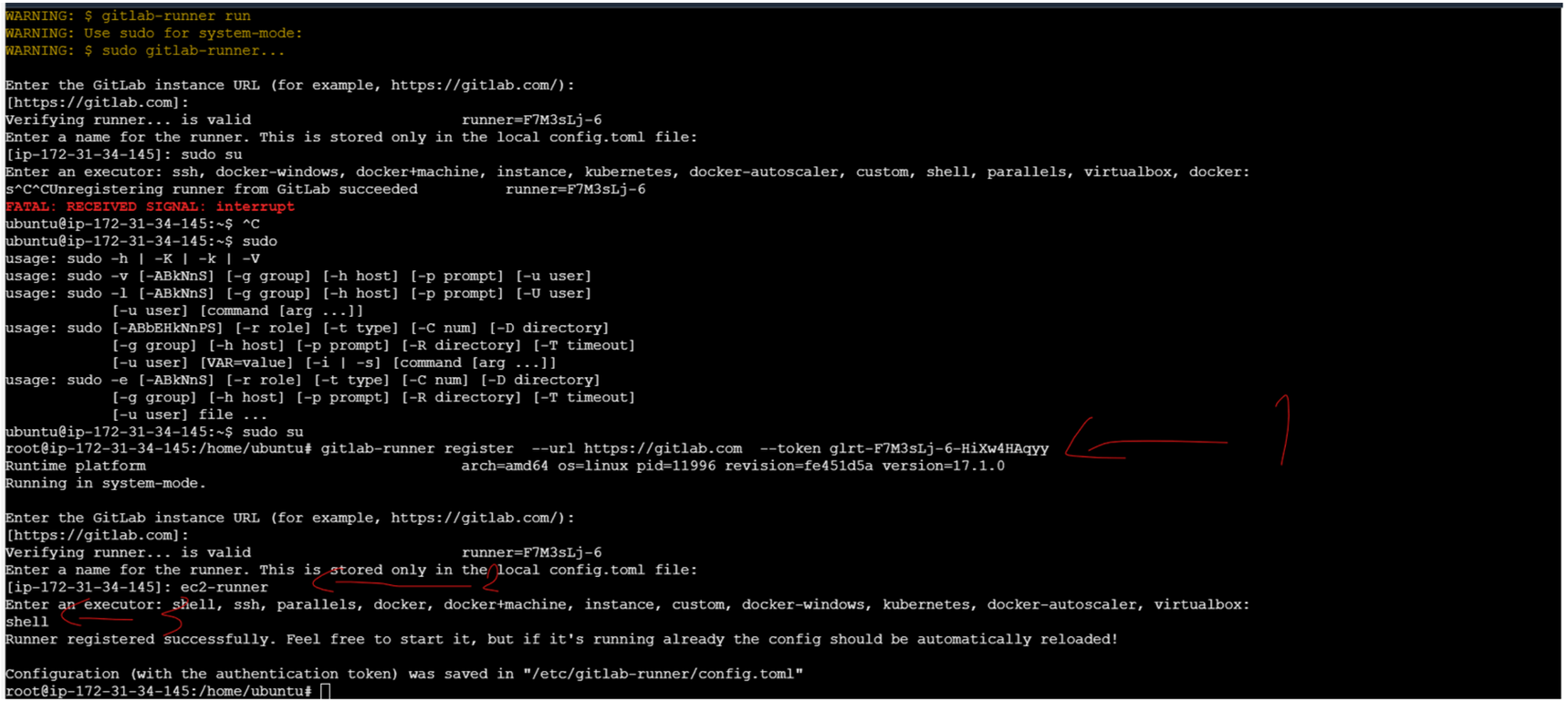
21. กดเลือกเป็น linux และทำการ ก็อป code step 1 นั้นไปวางหน้า ของ aws
ให้ก็อปปี้ เลข 1 เลข และไปทำข้อที่ 21 ก่อนที่จะกลับมา view runners





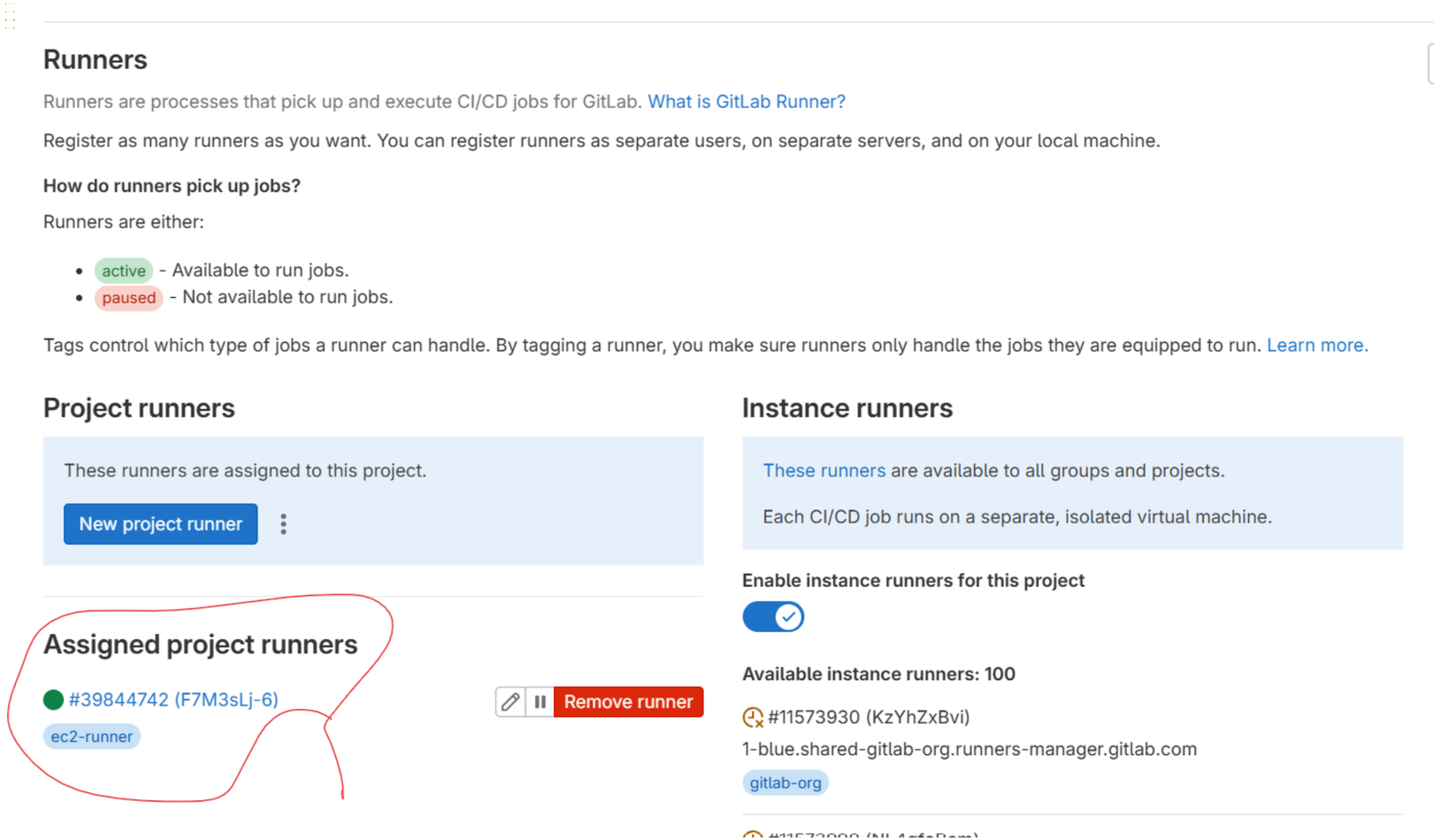
การสร้าง CI/CD เข้าไปยัง Docker

22. ต่อมา ใช้คำสั่ง sudo su เพื่อเข้าไปที่ home/ubuntu
ต่อมา copy จากข้อที่ 21 ที่หมายเลข 1 แล้วทำการวาง แล้วก็ enter และ ทำการ enter อีกครั้ง
ต่อมาใช้คำสั่ง เรียกหาที่เราสร้างก็คือ ec2-runner ecnter และกด shell อีกที enter
ต่อมาใช้คำสั่ง sudo chmod u+r /home/gitlab-runner/.bashrc
sudo systemctl restart gitlab-runner.service
ต่อมาย้อนกลับไปที่ข้อที่ 20 แล้วกด view runner



ตรวจสอบ view runner

23. ต่อมาย้อนกลับไปที่ข้อที่ 21 แล้วกด view runner



ถ้าเกิดปัญหาให้ลบ

sudo rm /etc/gitlab-runner/config.toml
sudo rm /etc/systemd/system/gitlab-runner.service
และไปเริ่มข้อที่ 20 ใหม่อีกครั้ง

gitlab-runner status
gitlab-runner stop
gitlab-runner start
gitlab-runner restart

Deploy NodeJS Application to AWS EC2 Using GitLab CI (youtube.com)

https://www.youtube.com/watch?v=hmbeksdTAls

ทำรูป CI/CD ทีละขั้นตอนให้เข้าใจง่ายที่สุด