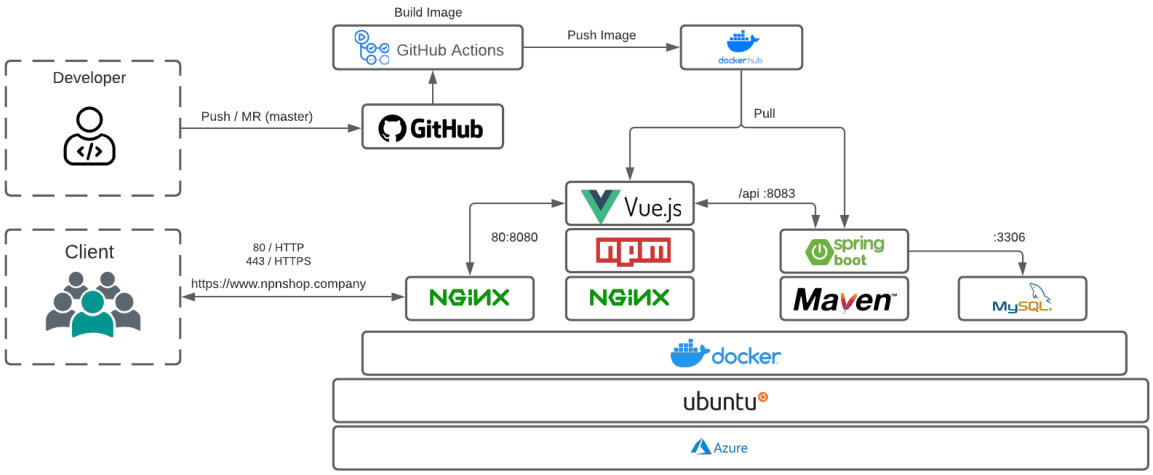
Teams: 62130500071, 62130500080, 62130500086

Responsibility:

62130500071: Reverse-Proxy Implementation
62130500080: Reverse-Proxy, CI/CD Implementation
62130500086: HTTPS Implementation

Infrastructure Diagram



Reverse Proxy (Nginx)

ใช้ Nginx เป็นตัวทำ Reverse Proxy โดยจะส่ง request ไปยัง Frontend หรือ Backend ตาม Path ที่ได้กำหนดไว้ และยังรับ Request ที่วิ่งเข้า Port 80 (HTTP) และ 443 (HTTPS) ซึ่งหาก Request เข้าที่ Port 80 จะทำการ Redirect Request ไปยัง Port 443 โดยอัตโนมัติ

Frontend (Vue.js, NPM, Nginx)

ใช้ Vue.js เป็น Framework สำหรับพัฒนาแอพลิเคชัน build ด้วย Node และ run server บน Nginx ที่ Port 8080:80

Backend (Spring Boot, Spring Maven)

ใช้ Spring Boot ที่มี Dependencies เป็น Spring Web และ Spring Security สำหรับส่ง Rest API และ ทำ Authentication. Build ด้วย Maven และ run server บน Tomcat Server (Built-in) ที่ Port 8083

Database (MySQL)

ใช้ MySQL เป็น Database และ run server ที่ Port 3306

Docker with Docker Compose

ใช้สำหรับ Pull Image, Build และ Run Image บน Docker Container พร้อมทั้งกำหนด Network และลำดับการ Build

Microsoft Azure & Ubuntu

Azure เป็น Cloud Service ที่ใช้สร้าง Virtual Machine (Ubuntu) และนำ Docker ไป run บน Virtual Machine

Version History (GitHub)

ใช้ GitHub สำหรับทำ version history และรวบรวม code

CI/CD Pipeline (GitHub Actions)

ใช้ GitHub Actions ดัก Push event ไปยัง master branch และส่งต่อให้ทำงานตาม Workflow ที่กำหนดไว้

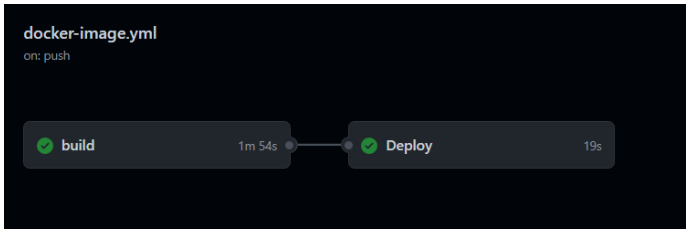
Image Repository (Docker Hub)

ใช้ Docker Hub สำหรับเป็น Image Repository ไว้เก็บ Image ที่ push ขึ้นไปครั้งล่าสุด

CI/CD Pipelines

Frontend Pipeline (./workflows/docker-image.yml)

จะแบ่งเป็น 2 stages ได้แก่ build และ deploy



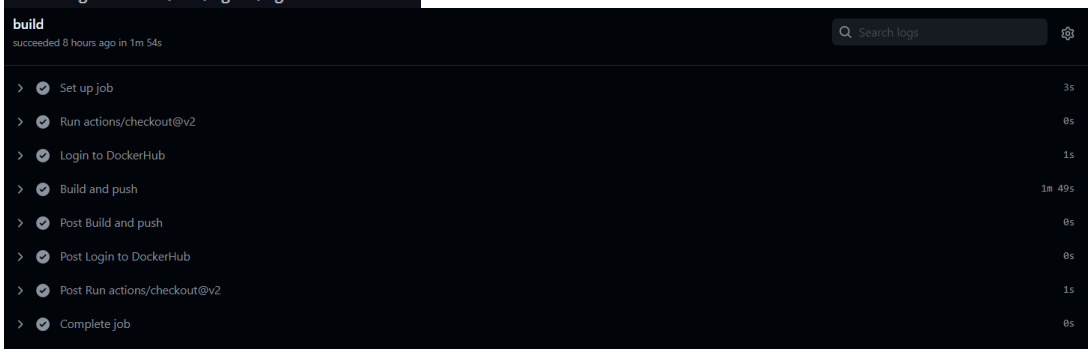
Stage build จะดำเนินการตาม steps ดังต่อไปนี้

- Checkout code in GitHub
- Login ไปยัง DockerHub โดยใช้ Username & Password ที่เก็บใน Repository Secrets
- Build and push โดยใช้ actions ของ docker/build-push-action@v2
    - โดยจะทำการ Build Image ตาม Dockerfile ที่อยู่ใน Repository
    - หลังจาก Build สำเร็จก็จะ Push Image ไปยัง DockerHub ที่กำหนดพร้อมติด Tag latest
    - build-args ใช้สำหรับใส่ Argument แทนค่าตัวแปรใน ARG ใน Dockerfile

```
FROM node:latest as build-stage
WORKDIR /app
ARG backend_url
ENV VUE_APP_BACKEND_URL=$backend_url
COPY package*.json ./
ENV NODE_OPTIONS=--openssl-legacy-provider
RUN npm install
COPY ./ .
RUN npm run build

FROM nginx as production-stage
RUN mkdir /app
COPY --from=build-stage /app/dist /app
COPY nginx.conf /etc/nginx/nginx.conf
```

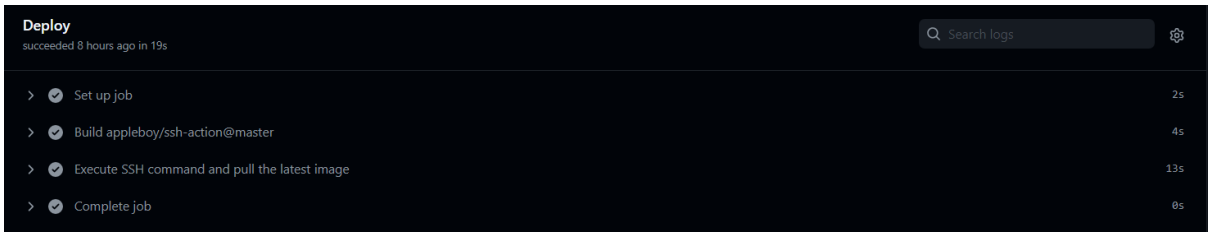
Dockerfile



Stage deploy จะรอ Stage build run jobs สำเร็จ แล้วจึงดำเนินการต่อดังนี้

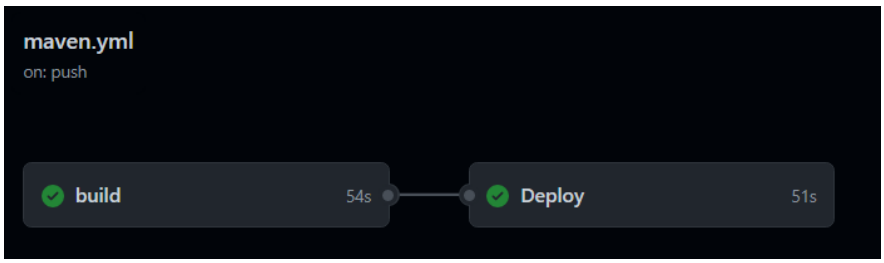
- ใช้ Actions ของ appleboy/ssh-action@master สำหรับ SSH เข้าไปยัง VM

- เรียกใช้คำสั่งตาม Script ที่เขียนไว้ ให้ดึง Image มาจาก Image Repository และ Recreate Container ใหม่



Backend Pipeline (./workflows/maven.yml)

จะแบ่งเป็น 2 stages ได้แก่ build และ deploy



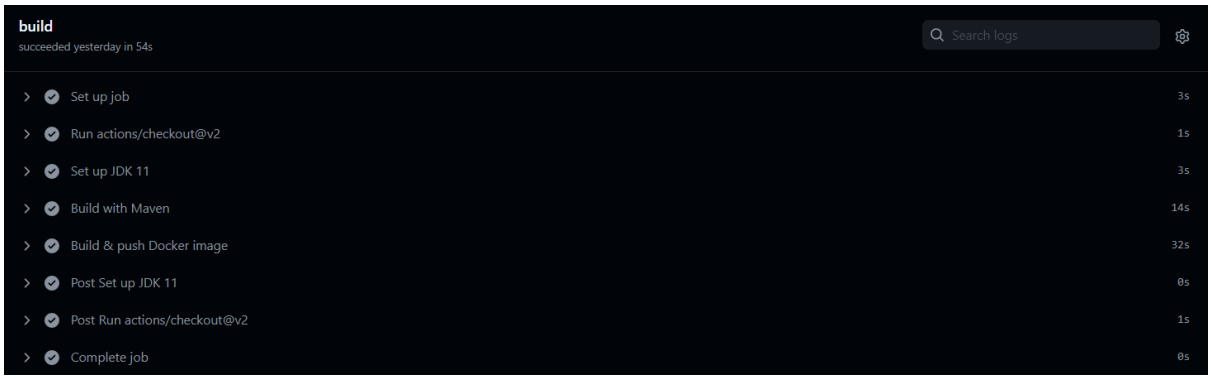
Stage build จะดำเนินการตาม steps ดังต่อไปนี้

- Checkout code in GitHub

- ติดตั้ง JDK 11 เพื่อที่จะ Build แบบ Maven

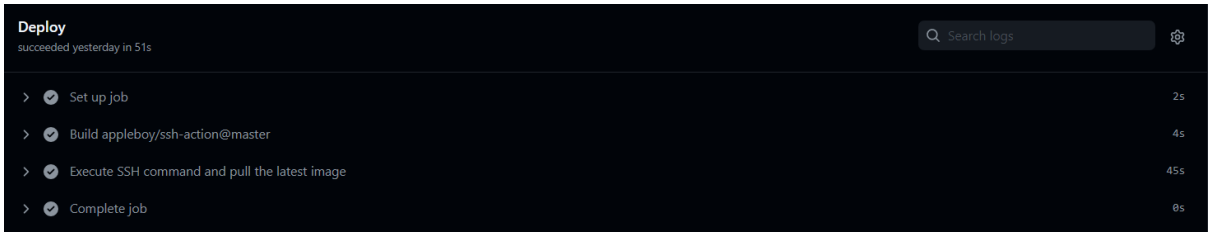
- Build โดยใช้คำสั่ง *mvn clean package -DskipTests*

- Login ไปยัง DockerHub โดยใช้ Username & Password ที่เก็บใน Repository Secrets และ Push image ขึ้นไปยัง Docker Hub



Stage deploy จะรอ Stage build run jobs สำเร็จ แล้วจึงดำเนินการต่อดังนี้

- ใช้ Actions ของ appleboy/ssh-action@master สำหรับ SSH เข้าไปยัง VM
- เรียกใช้คำสั่งตาม Script ที่เขียนไว้ ให้ดึง Image มาจาก Image Repository และ Recreate Container ใหม่



---

docker-compose.yml (ที่ root ของ Project file ใน VM)

- ตั้งค่า Pull image จาก Image Repository และกำหนด Environment ให้ Image ได้

```
  backend:
    image: varotsk137/my_springboot:latest
    restart: on-failure
    container_name: backend
    ports:
      - "8083:8083"
    depends_on:
      - mysql
    environment:
      SPRING_DATASOURCE_URL: ${SPRING_DB_URL}
      SPRING_DATASOURCE_USERNAME: ${SPRING_DB_USER}
      SPRING_DATASOURCE_PASSWORD: ${SPRING_DB_PSW}
      SERVER_FE_IP_ADDR: ${SERVER_FE_IP_ADDR}
      SERVER_FE_PORT: ${SERVER_FE_PORT}
      JWT_TOKEN_SECRET: ${INT222_JWT_SECRET}
    networks:
      project:
        ipv4_address: 172.23.20.2
  frontend:
    image: varotsk137/int222-vue3:latest
    restart: on-failure
    container_name: frontend
    ports:
      - "8080:80"
    depends_on:
      - backend
    networks:
      project:
        ipv4_address: 172.23.10.1
networks:
```

Reverse Proxy & HTTPS Implementation

ใน default.conf

HTTPS Implementation: Port 443 SSL, SSL On, Using Certificate & Certificate Key
Reverse Proxy: Frontend request to “/”, Backend request to “/api”

```
    server {
  listen 443 ssl;                              # รับ request เข้าที่ port 443
  server_name www.npnshop.company;             # ระบุ server_name เป็น domain name ที่กำหนด

  ssl on;                                      # ระบุว่ามีการใช้งาน Secure Socker Layer
  ssl_certificate /ssl/www.npnshop.company.crt; # ระบุตำแหน่งไฟล์ Certicate
  ssl_certificate_key /ssl/www.npnshop.company.key;  # ระบุตำแหน่งไฟล์ Cert Key
  ssl_session_cache shared:SSL:1m;             # กำหนด Session Cache
  ssl_session_timeout 5m;                      # กำหนด Session Timeout
  ssl_protocols SSLv2 SSLv3 TLSv1.3;           # กำหนด Protocol เป็น SSLv2/v3 TLSv1.3
  ssl_ciphers HIGH:!aNULL:!MD5;                # ระบุ Password Encryption Method
  ssl_prefer_server_ciphers on;                # web-server กำหนดว่าจะใช้ cipher ไหนได้

  location ~* \.(eot|ttf|woff|woff2)$ {   # Add Header ให้กับทุกๆ Request
    add_header Access-Control-Allow-Origin *;
    }

  location / {                                 # สำหรับทุก Request ที่มายัง Path ‘/’
    proxy_pass      http://frontend:80;   # ระบุให้ proxy ไปยัง ip frontend:80
    proxy_http_version  1.1;              # ระบุ Version Http
    proxy_cache_bypass  $http_upgrade;    # กำหนดให้ bypass cache สำหรับ Upgrade
    # Set Header ให้ Proxy Request
    proxy_set_header Upgrade           $http_upgrade;
    proxy_set_header Connection        "upgrade";
    proxy_set_header Host              $host;
    proxy_set_header X-Real-IP         $remote_addr;
    proxy_set_header X-Forwarded-For   $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Forwarded-Host  $host;
    proxy_set_header X-Forwarded-Port  $server_port;
  }
  location ^~ /api/ {                          # สำหรับทุก Request ที่มายัง Path ‘/api’
    proxy_pass      http://backend:8083;  # ระบุให้ proxy ไปยัง ip backend:8083
    proxy_http_version  1.1;              # ระบุ Version Http
    proxy_cache_bypass  $http_upgrade;    # กำหนดให้ bypass cache สำหรับ Upgrade
    # Set Header ให้ Proxy Request
    proxy_set_header Upgrade           $http_upgrade;
    proxy_set_header Connection        "upgrade";
    proxy_set_header Host              $host;
    proxy_set_header X-Real-IP         $remote_addr;
```

```
      proxy_set_header X-Forwarded-For   $proxy_add_x_forwarded_for;
      proxy_set_header X-Forwarded-Proto $scheme;
      proxy_set_header X-Forwarded-Host  $host;
      proxy_set_header X-Forwarded-Port  $server_port;
    }
  }
  server{
    listen 80;                                  # รับ request เข้าที่ port 80
    server_name www.npnshop.company;            # ระบุ server_name เป็น domain name ที่กำหนด
    rewrite ^(.*) https://$host$1 permanent;   # เขียนทับ request แก้จาก http เป็น https
  }
```

ใน nginx.conf ซึ่งเป็นตัวหลักที่ nginx จะเรียกทำงาน

```
user root;                          # กำหนด user เป็น root
worker_processes 1;                 # ระบุ process เป็น 1
error_log /var/log/nginx/error.log warn;     # ระบุตำแหน่งไฟล์ Log
pid /var/run/nginx.pid;             # ระบุ pid เป็นไฟล์ /var/run/nginx.pid


  events {
    worker_connections 1024;    # กำหนดจำนวนการเชื่อต่อสูงสุดเป็น 1024
  }
  http {
      # กำหนดให้ include MIME Type
    include /etc/nginx/mime.types;
    default_type application/octet-stream;

      # กำหนด Logs format
    log_format main '$remote_addr - $remote_user [$time_local] "$request" '
    '$status $body_bytes_sent "$http_referer" '
    '"$http_user_agent" "$http_x_forwarded_for"';
    access_log /var/log/nginx/access.log main;
    client_max_body_size 10m;
    sendfile on;

      # กำหนดเวลา Timeout ของ request
    keepalive_timeout 65;

      # กำหนดให้ include file .conf ทุกไฟล์ใน directory ที่กำหนด (ที่มีคือ default.conf ด้านบน)
    include /etc/nginx/conf.d/*.conf;
  }
```