



รายงาน การทำกระบวนการ ETL

นำข้อมูลคุณภาพอากาศจากเว็บไซต์เข้าสู่ฐานข้อมูล MySQL

ด้วย Apache Airflow

จัดทำโดย

นางนิจรา เจียมประดิษฐ์กุล 63606014

นำเสนอ

ดร.ชยานนท์ ทรัพย์อาภา

รายงานนี้เป็นส่วนหนึ่งของวิชา 06017468 BIG DATA ENGINEERING

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าคุณทหารลาดกระบัง

ภาคเรียนที่ 1 ปีการศึกษา 2564

# คำนำ

รายงานฉบับนี้จัดทำขึ้นประกอบการศึกษาในระดับปริญญาโท รายวิชา 06017468 BIG DATA ENGINEERING เนื้อหาของรายงานประกอบด้วย 3 ส่วนคือ

ส่วนที่ 1 รายละเอียดของแหล่งที่มาของข้อมูล

ส่วนที่ 2 โครงสร้างของข้อมูล ฐานข้อมูล และกระบวนการ ETL

ส่วนที่ 3 ประโยชน์ที่ได้รับจากการรวบรวมข้อมูล

โดยข้อมูลที่ใช้ประกอบการทำรายงาน เป็นข้อมูลสถิติด้านคุณภาพอากาศของประเทศไทยที่รวบรวมโดยกองจัดการคุณภาพอากาศและเสียง กรมควบคุมมลพิษ ในลักษณะเรียลไทม์ ในรูปแบบ JavaScript Object Notation (JSON file) ทำให้สามารถนำไปใช้ประโยชน์ในการวิเคราะห์ด้านต่างๆ ต่อไปได้

อนึ่งแล้ว มีหลายวิธีการในการดึงข้อมูลจากเว็บไซต์มาใช้ แต่วิธีการที่จะเป็นประโยชน์และลดภาระงานให้กับผู้เกี่ยวข้องทุกฝ่าย ควรเป็นวิธีการอัตโนมัติที่ต้องสามารถให้ผลลัพธ์การทำงานที่น่าเชื่อถือได้ มีความเสถียรและมีการรายงานสถานะการทำงานให้แก่ Data Engineer ได้ทราบด้วย โดยเครื่องมือหนึ่งที่น่าสนใจและมีประโยชน์ ได้แก่ Apache Airflow ซึ่งผู้จัดทำได้ศึกษา ทดลองใช้งาน และสรุปประเด็นสำคัญไว้ในรายงานฉบับนี้โดยสังเขป

นางนิจฉรา เจียมประดิษฐ์กุล

ผู้จัดทำ

# สารบัญ

## หน้า

|   |    |
|---|----|
| คำนำ.....                                   | 1  |
| สารบัญ.....                                 | 2  |
| สารบัญตาราง.....                            | 3  |
| สารบัญรูปภาพ.....                           | 4  |
| 1. รายละเอียดของแหล่งที่มาของข้อมูล .....   | 5  |
| 2. โครงสร้างของข้อมูล และฐานข้อมูล .....    | 6  |
| 2.1 โครงสร้างของข้อมูล และฐานข้อมูล .....   | 6  |
| 2.2 กระบวนการ ETL.....                      | 8  |
| 2.3 การทำ ETL โดย Apache Airflow .....      | 9  |
| 2.3.1 Apache Airflow.....                   | 9  |
| 2.3.2 Docker.....                           | 11 |
| 3. ประโยชน์ที่ได้รับจากการรวบรวมข้อมูล..... | 17 |
| เอกสารอ้างอิง.....                          | 13 |

## สารบัญตาราง

| ตารางที่  | หน้า |
|---|------|
| ตารางที่ 2.1 แสดงรายละเอียดตาราง TablePM25..... | 7    |

## สารบัญรูปภาพ

| รูปที่   | หน้า |
|--|------|
| รูปที่ 1.1 แสดงภาพเว็บไซต์ <a href="http://air4thai.pcd.go.th/webV2/">http://air4thai.pcd.go.th/webV2/</a> ..... | 5    |
| รูปที่ 2.1 แสดงโครงสร้างของ JSON File ที่ได้จากแหล่งข้อมูล.....  | 6    |
| รูปที่ 2.2 แสดงโปรแกรม Apache Airflowและการแสดงสถานะงาน.....   | 9    |
| รูปที่ 2.3 แสดงขั้นตอนของงานย่อย(Task) ภายใน DAG.....  | 10   |
| รูปที่ 2.4 โพลเดอร์หลักและโพลเดอร์ย่อยที่เก็บ Source Code.....   | 11   |
| รูปที่ 2.5 เนื้อหาภายในไฟล์ docker-compose.yml.....  | 12   |
| รูปที่ 2.6 เนื้อหาภายในไฟล์ dockerfile.yml สำหรับ Apache Airflow.....  | 13   |
| รูปที่ 2.7 เนื้อหาภายในไฟล์ dockerfile.yml สำหรับ MySQL.....   | 14   |
| รูปที่ 2.8 แสดงวิธีการตั้งค่า Airflow Connection.....  | 14   |
| รูปที่ 2.9 แสดงหน้าจอ ผลการทำงานของ DAGS : air_pm25 บน Apache Airflow.....                                       | 16   |

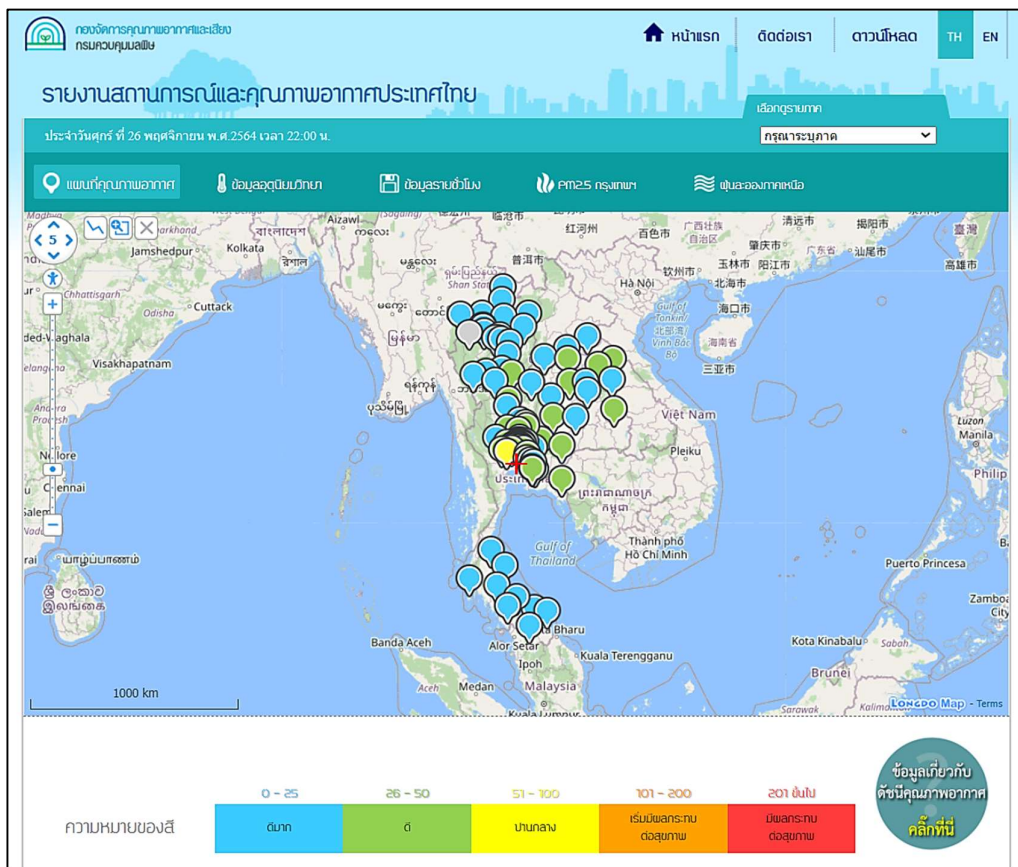
## 1. รายละเอียดของแหล่งที่มาของข้อมูล

ข้อมูลที่ใช้ประกอบการทำรายงาน เป็นข้อมูลสถิติด้านคุณภาพอากาศประเทศไทยที่รวบรวมโดยกองจัดการคุณภาพอากาศและเสียง กรมควบคุมมลพิษ โดยแสดงผลให้ผู้ใช้งานทั่วไปสามารถเข้าชมหน้าเว็บไซต์ <http://air4thai.pcd.go.th/webV2/> ในรูปแบบของกราฟฟิก ลักษณะดังภาพที่ 1.1

และได้จัดทำข้อมูลในรูปแบบ JSON เพื่อประชาชนผู้สนใจสามารถนำข้อมูลไปใช้งานต่อได้ และในรูปแบบ JavaScript Object Notation (JSON file )

แหล่งที่มาของข้อมูล :: [http://air4thai.pcd.go.th/services/getNewAQI\\_JSON.php](http://air4thai.pcd.go.th/services/getNewAQI_JSON.php)

ความถี่ในการรวบรวมข้อมูล :: ข้อมูล Update ทุกต้นชั่วโมง



รูปที่ 1.1 แสดงภาพเว็บไซต์ <http://air4thai.pcd.go.th/webV2/>

## 2. โครงสร้างของข้อมูล ฐานข้อมูล และกระบวนการ ETL

### 2.1 โครงสร้างของข้อมูล และฐานข้อมูล

JSON File ที่ได้จากการเรียกใช้งานลิงค์ข้อมูลจากข้อ1 มีโครงสร้างดังนี้

```
{
  stations: [
    {
      stationID: "02t",
      nameTH: "มหาวิทยาลัยราชภัฏบ้านสมเด็จเจ้าพระยา",
      nameEN: "Bansomdejchaopraya Rajabhat University",
      areaTH: "แขวงหิรัญรูจี เขตธนบุรี, กรุงเทพฯ",
      areaEN: "Hiran Ruchi, Khet Thon Buri, Bangkok",
      stationType: "GROUND",
      lat: "13.732846",
      long: "100.487662",
      forecast: [ ],
      LastUpdate: {
        date: "2021-11-26",
        time: "22:00",
        PM25: {
          value: "29",
          unit: "µg/m³"
        },
        PM10: {
          value: "49",
          unit: "µg/m³"
        },
        O3: {
          value: "44",
          unit: "ppb"
        },
        CO: {
          value: "-",
          unit: "ppm"
        },
        NO2: {
          value: "9",
          unit: "ppb"
        },
        SO2: {
          value: "-",
          unit: "ppb"
        },
        AQI: {
          Level: "2",
          aqi: "40"
        }
      }
    }
  ]
}
```

รูปที่ 2.1 แสดงโครงสร้างของ JSON File ที่ได้จากแหล่งข้อมูล

เนื่องจากผู้จัดทำมีความประสงค์ใช้งานในส่วนของ PM2.5 และ AQI เท่านั้น ดังนั้นตารางในฐานข้อมูลจะมีโครงสร้างตาราง ที่เกี่ยวข้องกับข้อมูลชุดนี้ ได้แก่

ตาราง : TablePM25

โครงสร้างข้อมูลแสดงในตารางที่ 2.1

ตารางที่ 2.1 แสดงรายละเอียดตาราง TablePM25

| TablePM25 |                       | ตารางข้อมูล PM2.5 ในแต่ละสถานีตรวจวัดสภาพอากาศ |            |    |                                  |                      |
|-----------|-----------------------|--|------------|----|----------------------------------|----------------------|
| Seq       | Field Name            | Field Type                                     | Field Size | PK | FK                               | Description          |
| 1         | rowid                 | varChar  | 10         | *  |                                  | Row ID               |
| 2         | stationID             | varChar  | 10         |    | TableStationnameTH.<br>stationID | รหัสสถานีฯ           |
| 3         | StationIDTH           | varChar  | 80         |    |                                  |                      |
| 3         | LastUpdate_date       | Date   | -          |    |                                  | วันที่ Update        |
| 4         | LastUpdate_time       | varChar  | 10         |    |                                  | เวลา Update          |
| 5         | LastUpdate_PM25_value | varChar  | 10         |    |                                  | ค่า PM 2.5           |
| 6         | LastUpdate_AQI_Level  | varChar  | 10         |    |                                  | ระดับคุณภาพ<br>อากาศ |
| 7         | LastUpdate_AQI_aqi    | varChar  | 10         |    |                                  | ดัชนีคุณภาพ<br>อากาศ |

- โครงสร้างข้อมูลในฐานข้อมูลจะแตกต่างจาก JSON File ต้นทาง ไฟล์ต้นทางมีจำนวนถึง 26 คอลัมน์ ดังนั้นจะผ่านกระบวนการ ETL เบื้องต้นเพื่อปรับให้ได้โครงสร้างตามที่ต้องการ



## 2.2 กระบวนการ ETL

**กระบวนการ ETL (Extract-Transform-Load) คือ** กระบวนการหนึ่งในระบบ Data Warehouse โดยระบบที่ออกแบบเอาไว้จะดึงข้อมูลออกมาจากหลายๆ แหล่ง และมีกระบวนการตรวจสอบคุณภาพของข้อมูลมาประยุกต์, มีการเชื่อมโยงและปรับข้อมูลให้เป็นไปในรูปแบบเดียวกันเพื่อให้ ข้อมูลจากหลายๆ แหล่ง สามารถใช้งานร่วมกันได้ และท้ายที่สุดทำการส่งมอบ (Delivery)

- **Extract (กระบวนการดึงข้อมูล) :** เป็นกระบวนการดึงข้อมูลจากแหล่งของข้อมูลภายนอกหรือจากภายในองค์กรด้วยตัวเอง ทั้งนี้อาจมีรูปแบบแตกต่างกันก็ได้ แต่จะต้องผ่านกระบวนการตรวจสอบและปรับรูปแบบให้มีมาตรฐานเดียวกันต่อไป
- **Transforming :** กระบวนการแปลงข้อมูลเพื่อให้ได้ตรงตามกับความต้องการ ซึ่งประกอบด้วย
  - Selection คือ เลือกเฉพาะ Column ที่ต้องการที่จะ Load
  - Translation คือ การแปลข้อมูล
  - Encoding free form
  - Deriving a new calculated value คือ กระบวนการคำนวณเพื่อให้ได้ข้อมูลที่ต้องการ
  - Filtering คือ กระบวนการกรองเฉพาะข้อมูลที่กำหนด
  - Sorting คือ กระบวนการเรียงข้อมูลที่ต้องการ
  - Joining คือ กระบวนการเชื่อมโยงข้อมูลระหว่างตารางข้อมูล
  - Aggregation คือ กระบวนการรวบรวม และ สรุปชุดข้อมูล
  - Generating surrogate key
  - Transposing or pivoting คือ การสลับทิศทางตำแหน่งของการแสดงข้อมูล
  - Splitting column into multiple columns คือ กระบวนการหั่นข้อมูลออกเป็นสดมภ์ย่อยๆ
  - Disaggregation คือ กระบวนการยกเลิกการรวมข้อมูล
  - Lookup and validate data กระบวนการ Lookup ข้ามตารางเพื่อตรวจสอบความถูกต้องของข้อมูล
  - Applying any form of simple or complex data validation กระบวนการนี้หากพบว่าการตรวจสอบข้อมูลผิด พลาดจะมีการปฏิเสธข้อมูลบางส่วนหรือทั้งหมดของข้อมูล แล้วแต่การกำหนดของผู้ที่ออกแบบการประมวลผลนี้
- **Loading :** นำข้อมูลเข้าสู่ระบบปลายทางที่ต้องการ ซึ่งโดยทั่วไปจะหมายถึงระบบ Data Warehouse หรือ ฐานข้อมูลอื่นใด

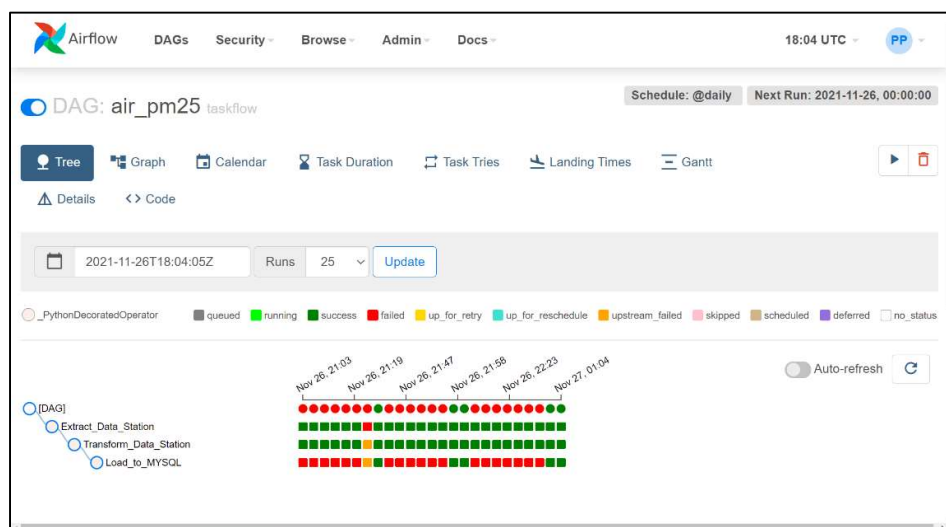
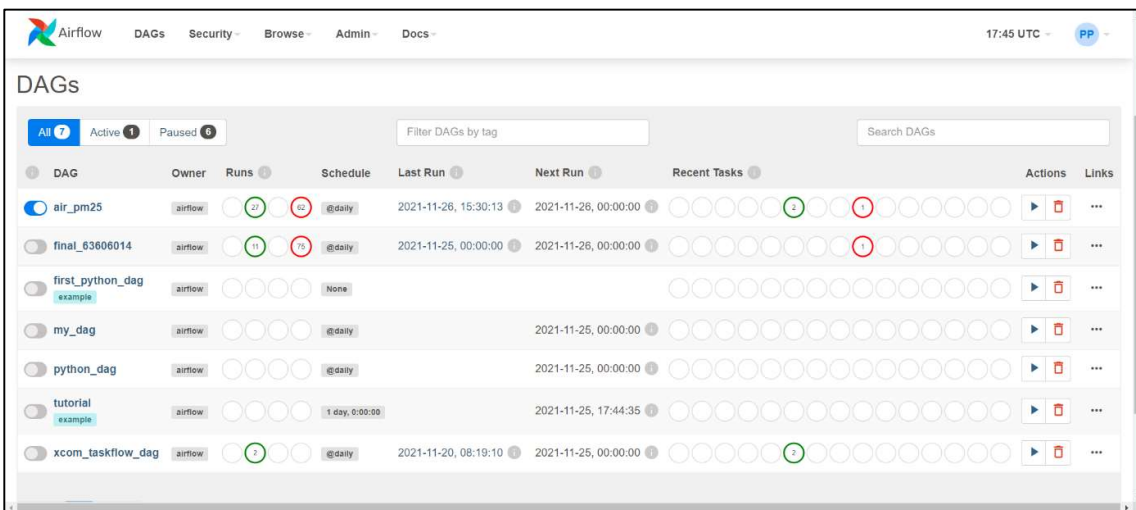
## 2.3 การทำ ETL โดย Apache Airflow

### เครื่องมือสำหรับการทำงาน

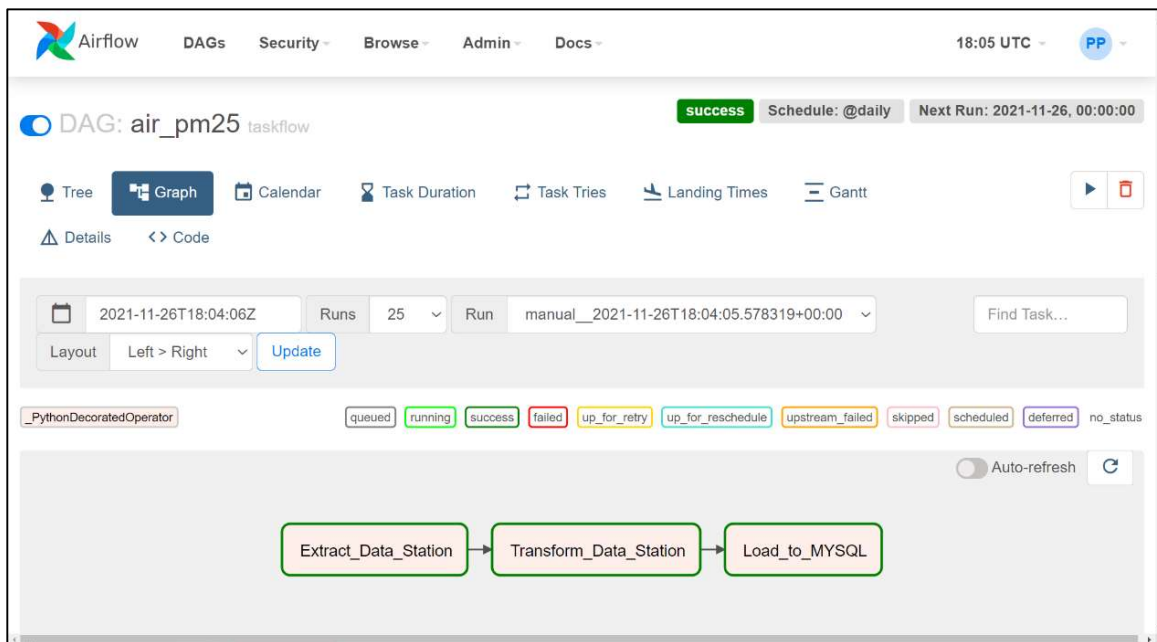
- Apache Airflow
- Docker
- PhpMyAdmin
- Microsoft Visual Studio

### 2.3.1 Apache Airflow

คือ management platform ที่ใช้ในการจัดการ workflow และ data pipelines แบบ programmatically ซึ่งจะทำให้กระบวนการที่ซับซ้อนสามารถถูกจัดการได้ง่ายขึ้น



รูปที่ 2.2 แสดงโปรแกรม Apache Airflow และการแสดงสถานะงาน



รูปที่ 2.3 แสดงขั้นตอนของงานย่อย(Task) ภายใน DAG

- แต่ละงานของ Airflow (DAG) จะถูกแสดงในรูปแบบ Graph เพื่อให้เห็นความสัมพันธ์และการเรียงลำดับของกิจกรรมย่อยๆ(Task) ภายในงานนั้น และผู้ใช้สามารถเขียนคำสั่งภาษาไพทอนจัดการข้อมูลภายในแต่ละงานได้

DAG (Directed Acyclic Graph) ประกอบไปด้วยหลายๆ Task ที่เชื่อมต่อกันและในแต่ละ Task นั้นก็มีความสามารถที่แตกต่างกัน

3 ประเภทหลักที่มักจะใช้ใน DAG

- Sensor เพื่อรอการทำงาน
  - Operator ใช้สำหรับสั่งงาน เช่น BashOperator, PythonOperator, MySqlOperator
  - Hook เชื่อมต่อกับระบบภายนอกมักจะเรียกใช้ผ่าน Sensor และ Operator เช่น GCSHook
- เราสามารถเขียนโปรแกรม เพื่อจัดการข้อมูลรันบน Airflow ได้โดยใช้ภาษาไพทอน ด้วยเครื่องมือเช่น MS-Visual Studio โดยเก็บ Source code ไว้ภายใต้ Folder Dags

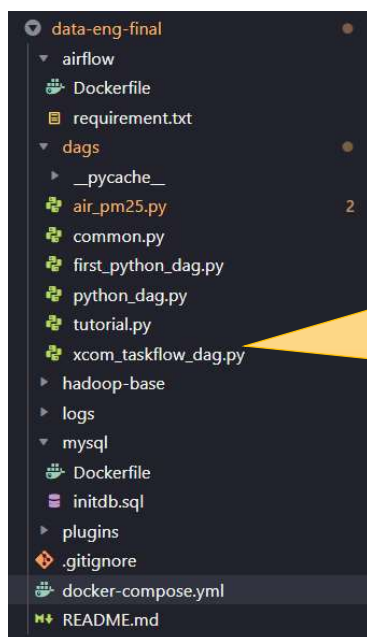
### 2.3.2 Docker

**Docker** เป็น “Software Container” ที่เป็นการสร้าง “สภาพแวดล้อมเฉพาะ” ให้กับซอฟต์แวร์ต่าง ๆ และทำให้ซอฟต์แวร์เหล่านั้น สามารถทำงานได้โดยไม่ไปรบกวนกับ ซอฟต์แวร์ตัวอื่นในระบบปฏิบัติการเดียวกัน เราสามารถนำ Container ไปติดตั้งบนคอมพิวเตอร์หรือเซิร์ฟเวอร์เครื่องอื่น ๆ ได้เลยทันที โดยที่โปรแกรมในนั้นยังทำงานได้ตามปกติ ไม่ผิดเพี้ยนไปจากเดิม เป็นแพลตฟอร์มซอฟต์แวร์ที่แยกแอปพลิเคชันที่กำลังพัฒนาออกจากกัน โดยการเรียกใช้งานในส่วนเฉพาะที่เรียกว่าคอนเทนเนอร์ (Container) โดย Docker Container จะมีส่วนที่คล้ายกันกับ Virtual Machine (VM) เพียงเล็กน้อย ตรงที่มันจะแยก Content ออกจากซอฟต์แวร์อื่นๆ ที่กำลังทำงานอยู่ในเครื่อง

Docker Container จะเก็บเฉพาะสิ่งที่จำเป็นในการเรียกใช้บริการซอฟต์แวร์เดี่ยวๆ เช่น ฐานข้อมูล (Database) หรือเว็บเซิร์ฟเวอร์ (Web Server) ซึ่งรวมถึงบริการของตัวมันเอง ตลอดจน Dependency ต่างๆ ที่จำเป็นต้องเรียกใช้ เช่น ไลบรารีซอฟต์แวร์ (Software Library) ดังนั้น จึงไม่มีเซอร์วิสของระบบปฏิบัติการหลัก (Underlying Operating System Services) ใน Container แต่ Docker จะได้รับสิ่งเหล่านี้จากเคอร์เนล (Kernel) ของระบบปฏิบัติการหลักของคอมพิวเตอร์โฮสต์แทน โดยเราสามารถใช้งาน Airflow ควบคู่กับการใช้งาน Docker สรุปรูวิธีการดังนี้

- **สรุปขั้นตอนการทำงานด้วย Docker**

1. เก็บรวบรวม Source Code ไว้ที่ Folder หนึ่ง : ภายใน Folder อาจแบ่งออกเป็น Folder ย่อยตามองค์ประกอบของ Source Code เช่น



ภายในโฟลเดอร์หลัก แบ่งเป็น  
โฟลเดอร์ย่อยตามองค์ประกอบของ  
ระบบที่กำลังพัฒนา เช่น

- Source Code
- Airflow
- MySQL

รูปที่ 2.4 โฟลเดอร์หลักและโฟลเดอร์ย่อยที่เก็บ Source Code

2. สร้างไฟล์ docker-compose.yml ไว้ที่ Folder ข้อ1.

ตัวอย่างเนื้อหาภายในไฟล์ docker-compose.yml

```
version: "3.9"
services:
  airflow:
    build: ./airflow
    volumes:
      - ./dags:/opt/airflow/dags
      - ./logs:/opt/airflow/logs
      - ./plugins:/opt/airflow/plugins
    ports:
      - 8080:8080
    command: bash -c "airflow webserver --port 8080 & airflow scheduler"

  mysql:
    #image: mysql/mysql-server:5.7
    build: ./mysql
    ports:
      - 3306:3306

  phpmyadmin:
    image: phpmyadmin/phpmyadmin:5.1
    depends_on:
      - mysql
    restart: always
    ports:
      - '8088:80'
    environment:
      PMA_HOST: mysql
      PMA_PORT: 3306
```

รูปที่ 2.5 เนื้อหาภายในไฟล์ docker-compose.yml

3. สร้างไฟล์ dockerfile.yml สำหรับแต่ละ Container เช่น dockerfile สำหรับ Apache Airflow และ MySQL

- ตัวอย่างเนื้อหาภายในไฟล์ dockerfile.yml สำหรับ Airflow

```
FROM ubuntu:20.04
RUN apt-get update
RUN apt-get install -y python3.8 python3-pip libmysqlclient-dev
RUN mkdir /opt/airflow
ENV AIRFLOW_HOME=/opt/airflow
ENV AIRFLOW__CORE__LOAD_EXAMPLES=False

RUN pip3 install apache-airflow==2.2.1 --constraint
https://raw.githubusercontent.com/apache/airflow/constraints-2.2.1/constraints-3.8.txt
RUN pip3 install pandas beautifulsoup4 sklearn
RUN pip3 install apache-airflow-providers-mysql==2.1.1

RUN airflow db init
RUN airflow users create \
  --username admin \
  --password password \
  --firstname Peter \
  --lastname Parker \
  --role Admin \
  --email spiderman@superhero.org

#-----
RUN pip3 install selenium && \
  pip3 install bs4 && \
  pip3 install elasticsearch && \
  pip3 install lxml && \
  pip3 install beautifulsoup4
#-----
RUN pip3 install xmldict
```

รูปที่ 2.6 เนื้อหาภายในไฟล์ dockerfile.yml สำหรับ Apache Airflow

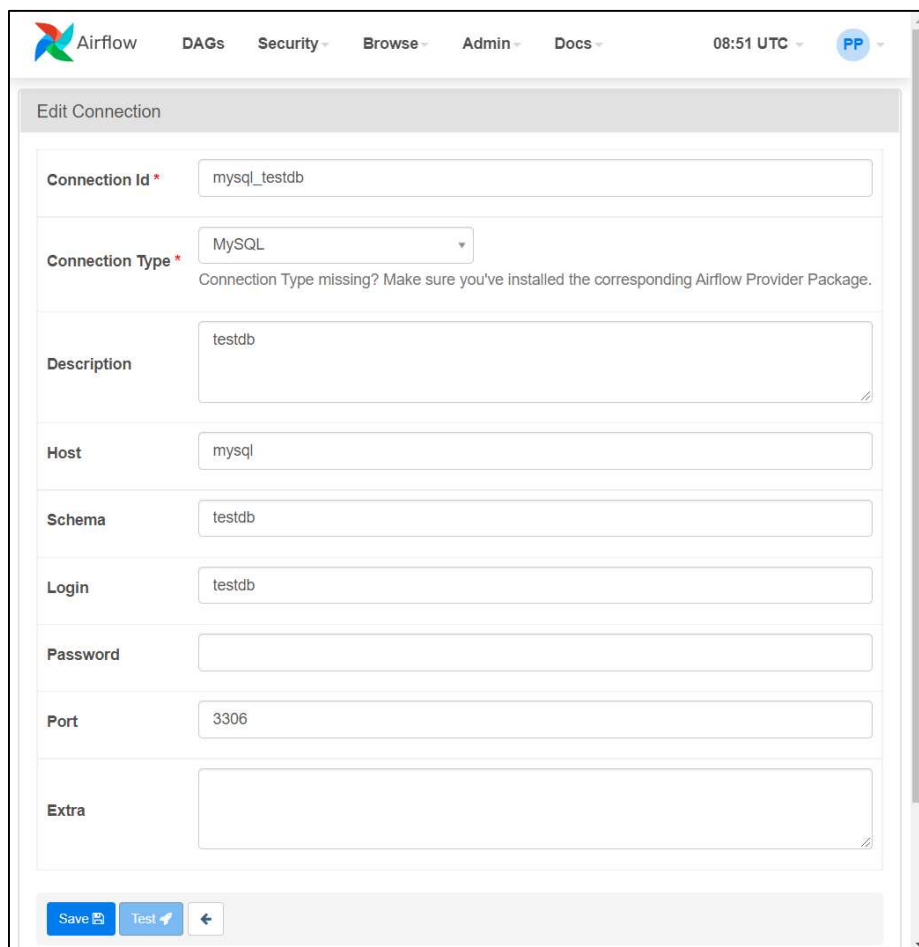
- ตัวอย่างเนื้อหาภายในไฟล์ dockerfile.yml สำหรับ MySQL

```
FROM mysql/mysql-server:5.7

ENV MYSQL_ROOT_PASSWORD=password
ENV MYSQL_DATABASE=testdb
ENV MYSQL_USER=testdb
ENV MYSQL_PASSWORD=testdb
```

รูปที่ 2.7 เนื้อหาภายในไฟล์ dockerfile.yml สำหรับ MySQL

4. ตั้งค่า Connection สำหรับ Apache Airflow ให้สัมพันธ์กับค่าในไฟล์ docker-compose.yml และ ไฟล์ dockerfile.yml สำหรับ MySQL



The screenshot displays the 'Edit Connection' form in the Apache Airflow web interface. The form includes the following fields:

- Connection Id \***: mysql\_testdb
- Connection Type \***: MySQL (with a dropdown arrow)
- Description**: testdb
- Host**: mysql
- Schema**: testdb
- Login**: testdb
- Password**: (empty field)
- Port**: 3306
- Extra**: (empty text area)

At the bottom of the form, there are three buttons: 'Save' (with a disk icon), 'Test' (with a lightning bolt icon), and a back arrow.

รูปที่ 2.8 แสดงวิธีการตั้งค่า Airflow Connection

5. เขียนโปรแกรมใช้งาน DAGS ของ Apache Airflow ด้วยภาษาไพทอนบน Ms Visual Studio
6. ใช้งาน terminal บนโปรแกรม Ms Visual Studio โดยที่ต้องเปลี่ยนพาธการใช้งานไปที่ folder ของ docker-compose.yml แล้วรันคำสั่งบน terminal ดังนี้

- สรุปคำสั่งสำคัญสำหรับ Docker

A) เลือก image ที่ต้องการ เช่น apache ใช้คำสั่ง :

```
FROM php:7.2-apache
```

B) การสั่งเริ่มต้น docker ( Up-> Build -> run )

```
docker-compose up
```

```
docker build -t my_webapp .
```

```
docker run -d -p 8080:80 --name webapp2 my_webapp
```

C) การสั่ง up airflow โดยใช้ชื่อไฟล์ที่ต้องการ เช่น ชื่อไฟล์ docker-compose-airflow-local.yml

```
docker-compose -f docker-compose-airflow-local.yml up
```

D) การสั่ง remover image :

```
docker rm webapp
```

E) การสั่ง stop container :

```
docker stop docker_name
```

F) การสั่ง build image จากไฟล์ docker-compose.yml :

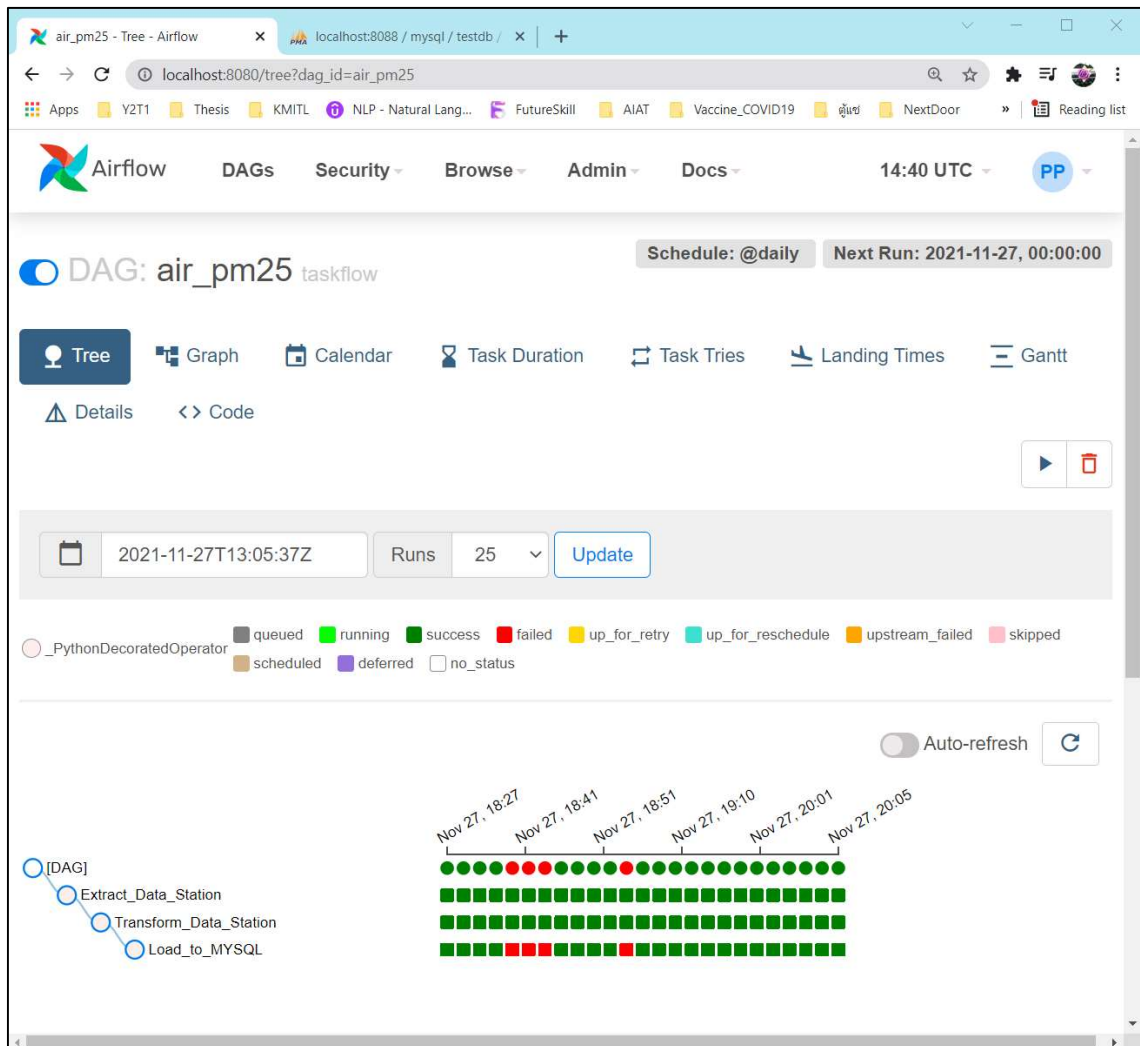
```
docker-compose build
```

7. ดูผลการจัดการข้อมูลผ่าน Apache Airflow ตาม Port ที่ระบุไว้ใน Docker-compose.yml ได้แก่

- Apache Airflow      ➔      Localhost : 8080

- PHPMySQL            ➔      Localhost : 8088





รูปที่ 2.9 แสดงหน้าจอ ผลการทำงานของ DAGS : air\_pm25 บน Apache Airflow

### 3. ประโยชน์ที่ได้รับจากการรวบรวมข้อมูล

ข้อมูลชุดนี้เป็นข้อมูลประกอบในการวางแผนงานการบริหารความเสี่ยงองค์กร และเป็นข้อมูลเพื่อสายงานการตลาด

โดยจะนำไปใช้งานเพื่อการวางแผนในด้านต่างๆ ได้แก่

1. การประเมินความเสี่ยง
2. การวางแผนงบประมาณองค์กร
3. การวางแผนประชาสัมพันธ์และการตลาด
4. การวิจัยผลิตภัณฑ์ของบริษัท เป็นต้น

## บรรณานุกรม

“รายงานสถานการณ์และคุณภาพอากาศประเทศไทย”, 2564. <http://air4thai.pcd.go.th/webV2/>

“ข้อมูลสถานการณ์และคุณภาพอากาศประเทศไทย” 2564.

[http://air4thai.pcd.go.th/services/getNewAQI\\_JSON.php](http://air4thai.pcd.go.th/services/getNewAQI_JSON.php)

“การทำ Data Pipeline ด้วย Apache Airflow”, 2564.

<https://medium.com/@pakkadkittika/%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B8%97%E0%B8%B3-data-pipeline-%E0%B8%94%E0%B9%89%E0%B8%A7%E0%B8%A2-apache-airflow-cf1bedd4d9f1>

“แบบจำลองฐานข้อมูลเชิงสัมพันธ์.”, Retrieved 2010.

<https://www.ict.up.ac.th/worrakits/Database.files/charpter4.pdf>