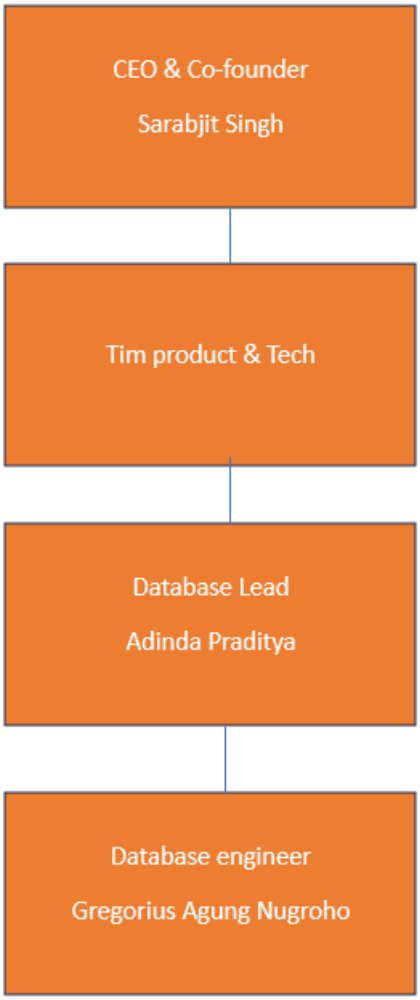
BAB III
PELAKSANAAN KERJA MAGANG

Selama program kerja magang di Cube Asia, penulis bekerja pada tim *product & tech* pada divisi data engineer. Penulis berada dibawah koordinasi CEO Sarabjit Singh, dan diatur juga dibimbing oleh proyek manager Adinda Praditya. Tim database *engineer* memiliki fungsi untuk menyimpan data, menjaga data yang masuk tetap terpercaya, dan meningkatkan kualitas data yang dimiliki. Tim database *engineer* berkolaborasi dengan beberapa tim lain seperti tim data *extraction*, dan tim data analyst.



Gambar 3.1 Kedudukan Penulis

Peyimpanan data dan pengecekan kualitas data merupakan salah satu hal terpenting dalam berjalannya perusahaan, karena Cube Asia merupakan Perusahaan yang menganalisis data dan memberikan *insights* ke client. Sehingga data yang akan diolah tersebut harus tersimpan dengan baik, dan merupakan data yang reliabilitas dan kualitasnya tinggi. Apabila data yang disimpan tidak baik maka akan menurunkan kualitas dari *insights* tersebut.

Penulis diberikan tugas untuk membuat sistem monitoring data yang masuk ke database, dan meningkatkan kualitas data di dalam database kita. Script yang dikembangkan menggunakan bahasa pemrograman Ruby on Rails, PostgreSQL, dan Python. Untuk monitoring sendiri, hasil monitoring akan dikirimkan lewat *email* karena *email* merupakan alat komunikasi utama pada Cube Asia. Penulis juga berkoordinasi dengan tim lain mengenai data apa saja yang harus di monitor dan data apa saja yang harus dinaikan kualitasnya.

Penulis berkoordinasi dengan tim lain pada divisi data engineer dengan metode sprint. Setiap sprint dirancang agar tim dapat berkerja dengan cepat dan efisien. Setiap development life cycle terdapat beberapa pertemuan:

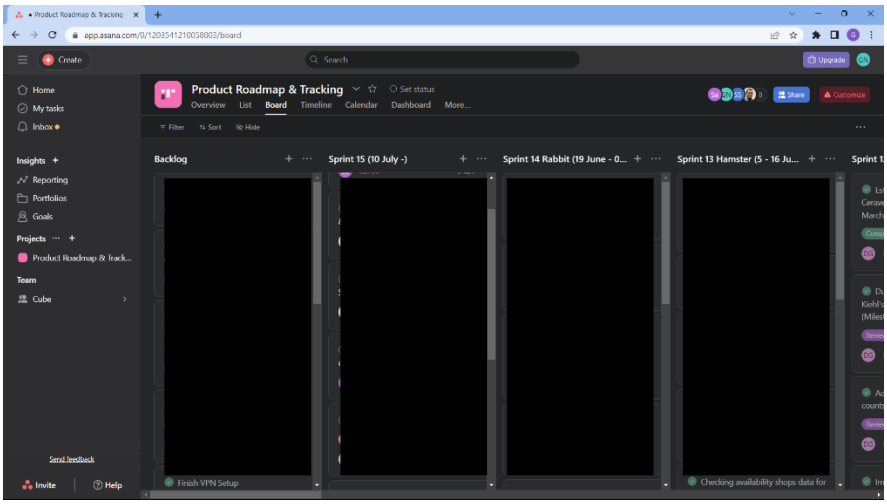
1. Daily check in: diskusi dengan mentor mengenai yang akan dilakukan hari ini
2. Sprint planning: diskusi dengan seluruh tim membahas apa yang akan dilakukan minggu ini
3. Sprint review: diskusi dengan seluruh tim membahas apa yang telah dilakukan minggu ini
4. Adhoc meeting: diskusi dengan sebagian orang membahas hal-hal penting dan *urgent*

Untuk proyek-proyek khusus tertentu, database manager dapat membuat tiket untuk penulis, meskipun biasanya penulis yang membuat tiket sendiri. Pembuatan tiket ini dilakukan selama sprint planning, dan selanjutnya, penulis akan menangani tugas-tugas yang tercantum dalam tiket tersebut. Setiap hari pada saat

daily check-in, penulis akan melaporkan tugas-tugas yang sudah diselesaikan. Estimasi waktu pengerjaan tiket selalu menjadi topik pembahasan antara penulis, database manager, dan CEO.

Pada pertemuan sprint mingguan, seluruh tim akan berkumpul untuk membahas tiket-tiket yang telah dibuat selama sprint planning bersama CEO. Hal ini mencakup tiket-tiket yang sudah selesai serta tiket-tiket yang perlu diperpanjang ke sprint berikutnya. Seluruh tiket dan tugas yang sudah diselesaikan akan dirangkum dan disimpan menggunakan aplikasi bernama Asana sebagai alat pemantauan, yang nantinya akan disampaikan kepada CEO.

Gambar 3.2 Logo Asana



Gambar 3.3 Dashboard Asana

3.2 Tugas dan Uraian Dalam Kerja Magang

Beberapa tugas, uraian, kendala dan solusi yang dikerjakan penulis selama kegiatan magang dijelaskan pada sub bab dibawah ini

3.2.1 Tugas yang Dilakukan

Tugas-tugas yang dilakukan penulis selama kegiatan magang di Cube Asia, sebagai berikut:

Tabel 3.1. Tugas Kerja Magang

Week	Onboarding	Update script import product	Membuat system *performance tracking*	Test & Bug fixing	Menigkatkan kualitas data	Weekly Meeting
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						

Dari linimasa tugas kerja magang, dapat dijabarkan tugas yang dikerjakan penulis, yaitu:

1. Onboarding

   Tahap ini, penulis belajar bagaimana cara kerja script importing product berjalan. Penulis juga diminta untuk mempelajari baha Ruby on Rails dan PostgreSQL
2. Update script import product

   Penulis diminta untuk memperbaiki script importing product untuk menambahkan data-data yang belum di import ke database.
3. Membuat sistem *performance tracking*

Penulis diminta untuk membuat data rangkuman dari data product yang di import lalu data dari rangkuman yang sudah dibuat penulis diminta mengirimkan data data penting sebagai sistem monitoring melalui email.

4. Testing & Bug Fixing

   Penulis teentunya akan melakukan kegiatan testing dan bug fixing pada scripts yang dibuatnya

5. Meningkatkan kualitas data

   Penulis diminta meningkatkan kualitas data dalam database. Karena dalam hasil monitoring data terdapat dapat yang kurang bagus dalam database

6. Weekly Meeting

   Penulis akan menghadiri weekly meeting di setiap hari selasa dan jumat untuk menceritakan progress pekerjaannya

Tabel 3.2 Ringkasan Kerja Magang per Minggu

Minggu	Tugas yang Dilakukan
1-2	1. Pengenalan dengan Tim dan Mentor 2. Mempelajari Script import product yang diberikan 3. Mempelajari Ruby on Rails dan PostgreSQL 4. Menghadiri Weekly Meeting
3-4	1. Mengupdate script import product 2. Menambah data-data yang dimasukkan ke database 3. Menghadiri Weekly Meeting
5-6	1. Membuat script yang menghasilkan rangkuman dari data produk yang diimport 2. Menghadiri Weekly Meeting
7-8	1. Test & bug fixing untuk script yang membuat rangkuman 2. Menghadiri Weekly Meeting 3. Mempelajari sistem monitoring dan menentukan hal terbaik

9-10	1. Mencoba menggunakan layanan AWS Lambda untuk mengirimkan email 2. Membuat script yan mengirimkan data pilihan dari rangkuman ke email 3. Menghadiri weekly meeting
11-12	1. Testing dan Bug fixing scripts monitoring lewat email 2. Menghadiri weekly meeting
13-14	1. Testing dan bug fixing scripts monitoring lewat email 2. Mencari cara terbaik untuk mengambil data account yang tidak ada di database 3. Menghadiri weekly meeting
15-16	1. Mencari cara terbaik untuk mengambil data account yang tidak ada di database 2. Menghadiri weekly meeting
17-18	1. Mengambil dan memverifikasi data yang account yang benar untuk dimasukkan ke database 2. Menghadiri weekly meeting

3.2.2 Uraian Pelaksanaan Kerja Magang

Uraian dan gambaran proyek yang dijelaskan dibawah merupakan ilustrasi yang dibuat penulis agar pembaca mendapatkan gambaran hasil proyek. Hal ini dilakukan untuk menjaga kerahasiaan data perushaan Cube Asia.

Penyimpanan data pada Cube Asia dilakukan untuk menyimpan data-data dari produk-produk yang sudah diambil oleh tim data *extraction* yang nantinya data ini akan dipakai oleh data *analyst* untuk di analisis dan menghasilkan *insights* untuk client. Karena sistem analisis di Cube Asia itu beragam dan biasanya membutuhkan data minimal 1 bulan terakhir, sehingga data 1 bulan terakhir itu harus disimpan dengan baik di database, dan juga harus dijaga kualitas nya agar dapat menghasilkan *insights* dengan kualitas yang baik pula.

Dalam pelaksaan kerja magang di Cube Asia sebagai database engineer intern, pada awalnya Perusahaan dihadapi dengan suatu masalah, yaitu tidak adanya proses yang memonitor data yang masuk kedalam database Perusahaan sehingga penulis dan tim lainnya tidak tahu apapun tentang data yang masuk ke database. Oleh masalah tersebut, penulis diminta untuk membuat sebuah proses monitoring dari sistem *import product data* yang sudah ada.

Setelah penulis berhasil membuat proses monitor data, muncul masalah baru yaitu dari hasil proses monitoring tersebut ditemukan bahwa muncul masalah baru dimana terdapat data tiktok *shop* yang saat dimasukkan datanya, ternyata tidak dapat dihubungkan dengan tiktok *account* yang ada didalam database, sehingga membuat data untuk tiktok *shop* menjadi tidak lengkap dan membuat data di database perusahaan kurang *reliable*.

Oleh karena itu, penulis diminta oleh CEO untuk mengatasi permasalahan tersebut, yaitu dengan mengurangi tiktok *shop* yang tidak mempunyai tiktok *account* dengan cara mencari tiktok *account* yang benar dimiliki oleh *shop* tersebut, dan memasukkan kembali data tiktok. Proses mencari tiktok *accounts* ini harus dilakukan dengan seefisien mungkin.

Sebelumnya masalah ini juga pernah terjadi namun penyelesaian dari permasalahan ini adalah dengan meminta seorang *reviewer* untuk mencari menggunakan aplikasi tiktok nama *shop* tersebut dan mengembalikannya ke data *collection* untuk diambil datanya lalu baru dimasukkan ke database. Proses ini memakan waktu yang lama, sehingga penulis mengusulkan untuk membuat otomasi dari proses tersebut, Oleh karena itu penulis mengusulkan 2 algoritma untuk mencari tiktok *accounts* ini. Berikut merupakan rincian dari proses penulis membuat sistem monitoring dan 2 algoritma untuk meningkatkan kualitas data di database pada Cube Asia

1. Onboarding

   Penulis diberikan scripts oleh mentor untuk dipelajari, scripts yang diberikan oleh mentor berisi bagaimana proses import data ke dalam database. Scripts yang diberikan adalah scripts Ruby on Rails, sehingga penulis diminta untuk mempelajari Ruby on Rails, dan cara scritps itu berjalan. Ruby on Rails atau biasa disebut rails merupakan framework aplikai web yang mencakup semua yang diperlukan untuk membuat aplikasi web yang didukung database dengan pola *Model View Controller* (MVC). Sejalan dengan penulis belajar Rails dan scripts yang telah diberikan, penulis juga diminta untuk mempersiapkan database PostgreSQL pada *device* penulis. Setelah itu penulis akan diberikan *pgdump* yang nantikan akan dimasukkan di database penulis, database ini yang akan menjadi tempat penulis untuk melakukan pengembangan script import product sebelum script tersebut diterapkan di database aslinya di server.

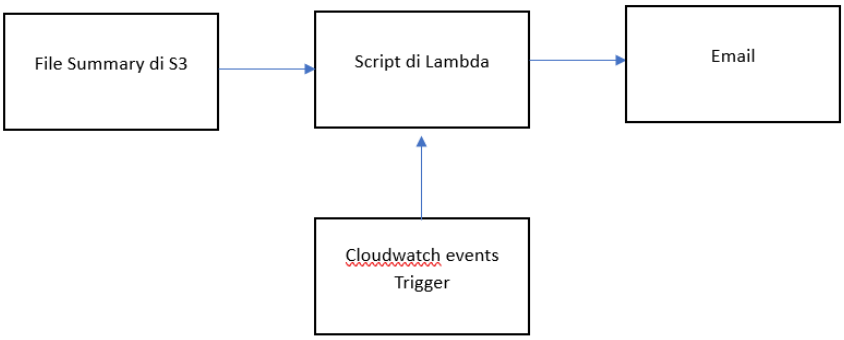
2. Update script import product

   Penulis diminta untuk membuat sedikit perubahan pada script import produk. Perubahan yang dilakukan adalah menambahkan data-data yang harus disimpan, seperti jumlah toko dan akun yang diproses selama import script produk itu berjalan. Hal ini dilakukan karena data-data tersbut akan digunakan untuk sistem monitoring nantinya, sehingga data tersebut harus kita simpan.

3. Pembuatan sistem *monitoring import data*

   Sistem *monitoring import data* ini dibuat karena dalam *importing product data* pada Cube Asia, kita tidak hanya memproses 1 json *files* saja, namun banyak json *files* dan dibagi dalam beberapa batch. Sehingga dalam pembuatan sistem *monitoring* ini kita harus memikirkan untuk mengambil data yang penting saja agar tidak memakai *resource* yang banyak. Sehingga penulis membuat *files summary* atau rangkuman dari seluruh proses untuk mendapatkan data-data yang kita inginkan. Lalu setelah itu penulis menentukan

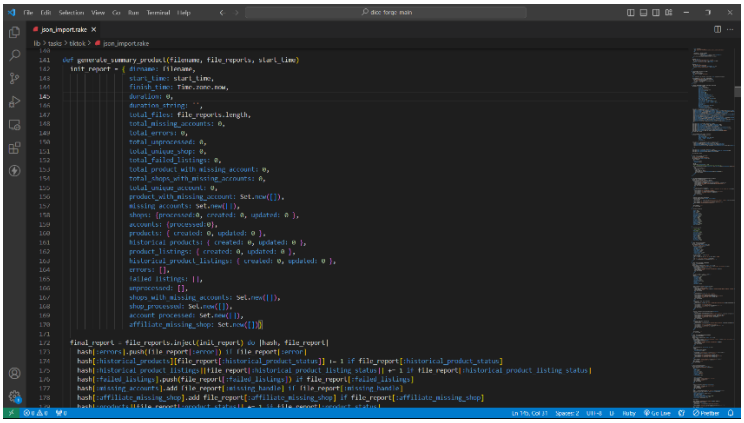
resource apa yang akan dipakai dan sekaligus membuat *script* untuk memonitoring impor data tersebut.



Gambar 3.4 Alur proses monitoring

a. Pembuatan Summary

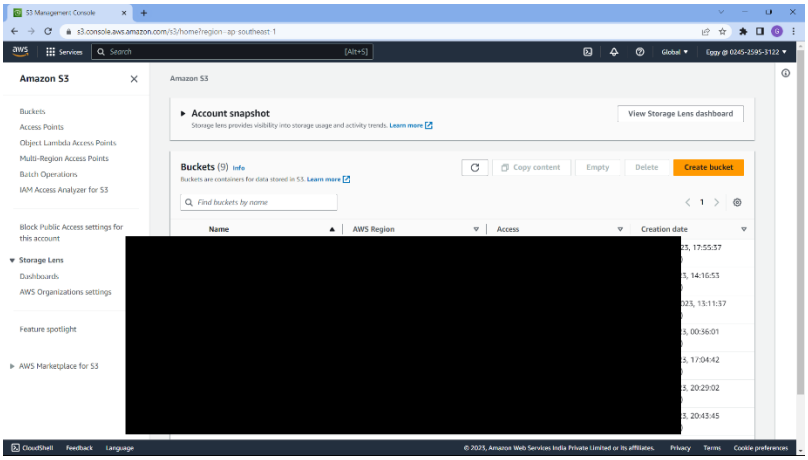
Pembuatan sistem *performance tracking* diawali dengan pembuatan *summary* atau rangkuman dari proses import product data. *Summary* ini berisi tentang berapa produk yang berhasil masuk ke database, berapa shop dan akun yang di proses, error apa saja yang dihasilkan, dan data penting lainnya.



Gambar 3.5 Script Summary

Script *summary* ini dibuat dengan rails, dan nantinya akan berjalan secara otomatis, setelah script import produk selesai, lalu *summary* itu akan disimpan di S3. S3 merupakan layanan dari AWS yang memberikan penyimpanan data berbasis cloud. Summary ini akan terbentuk setelah *script* untuk *import product* selesai berjalan pada setiap *batch*. Sehingga apabila ada 5 *batch*

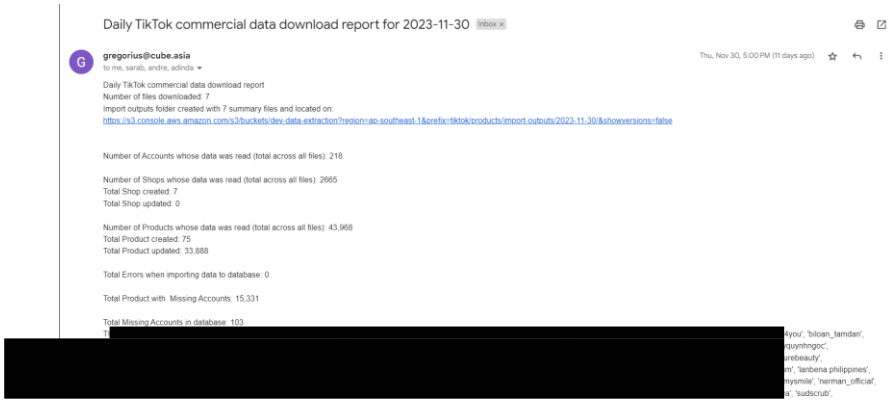
data yang di *import* ke database, maka akan terbentuk pula 5 *files summary* di S3.



Gambar 3.6 AWS S3 Dashboard

b. Pemilihan Resource dan Pembuatan *Script*

Setelah itu penulis diminta untuk dapat membuat data-data dari *summary* tersebut dapat juga dilihat oleh tim-tim lain tanpa harus membuka S3. Oleh karena itu, penulis mengusulkan untuk membuat sistem yang mengirimkan data summary tersebut melalui email. Dimana setiap harinya pukul 5 WIB, setiap tim akan mendapatkan sebuah email, yang berisi tentang performa dari import script hari ini.

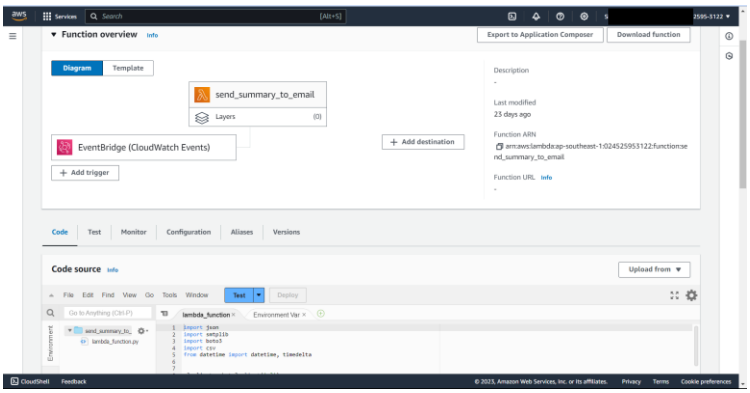


Gambar 3.7 Contoh Email

Penulis memilih menggunakan email untuk media informasi dikarenakan email merupakan alat komunikasi utama pada Cube

Asia. Selain itu, perusahaan tidak perlu mengeluarkan biaya apapun untuk membuat monitoring melalui email ini karena untuk mengirimkan email kita tidak membutuhkan biaya apapun.

Dalam menjalankan script ini, penulis menggunakan layanan AWS Lambda. Lambda adalah layanan yang disediakan oleh AWS yang memungkinkan eksekusi skrip tanpa server, menghilangkan kebutuhan untuk menggunakan sumber daya yang dimiliki oleh engineer dan membebaskan sumber daya tersebut untuk keperluan lainnya. Beberapa aspek yang perlu dicatat mengenai Lambda meliputi fakta bahwa layanan ini memungkinkan eksekusi skrip tanpa perlu memiliki server. Selain itu, Lambda memiliki batas waktu eksekusi selama 15 menit, dan setelah melewati batas tersebut, skrip akan dihentikan secara otomatis. Selain itu, Lambda juga menyediakan file sementara untuk melakukan pemrosesan file.



Gambar 3.8 AWS Lambda Dashboard

Untuk mengkoneksikan Lambda dengan S3 kita akan menggunakan library boto3 yang sudah disiapkan oleh AWS.

Gambar 3.9 Code Mengambil Data dari S3

Code pada gambar 3.9 merupakan code untuk kita dapat mengambil data dari s3 menggunakan boto3, karena file yang akan kita proses adalah file json dengan nama khusus, maka kita harus membuat algoritma yang hanya mengambil file json dengan nama khusus tersebut.

```
s3.Bucket(bucket_name).download_file(key,local_file_handled_name)

lists = [i,'https://tiktok.com/@' + handle,handle,followers,following,

# write the data into '/tmp' folder
with open(local_file_handled_name,'r') as infile:
    reader = list(csv.reader(infile))
    reader = reader[::-1] # the date is ascending order in file
    reader.insert(0,lists)

with open(local_file_handled_name, 'w', newline='') as outfile:
    writer = csv.writer(outfile)
    for line in reversed(reader): # reverse order
        writer.writerow(line)

# upload file from tmp to s3 key
bucket.upload_file(local_file_handled_name, key)
```

Gambar 3.10 Code Memasukkan data ke S3

Kode yang terdapat pada gambar 3.10 menjelaskan bagaimana kita memasukkan data ke S3. Dalam memasukkan data ke S3 kita memakai salah satu kelebihan lambda, yaitu menggunakan file sementara lambda. Kita menggunakan file sementara, karena kita tidak dapat mengganti file S3 secara langsung. Setelah kita memasukkan semua ke file sementara,

kita akan kembali mengupload file sementara it uke S3 dengan nama yang sama. Karena di S3, jika kita mengupload dengan nama file yang sama, maka file itu akan ke *overwrite*. Setelah penulis berhasil menghubungkan Lambda dan S3, penulis menambahkan algoritma untuk mengambil data yang diperlukan.

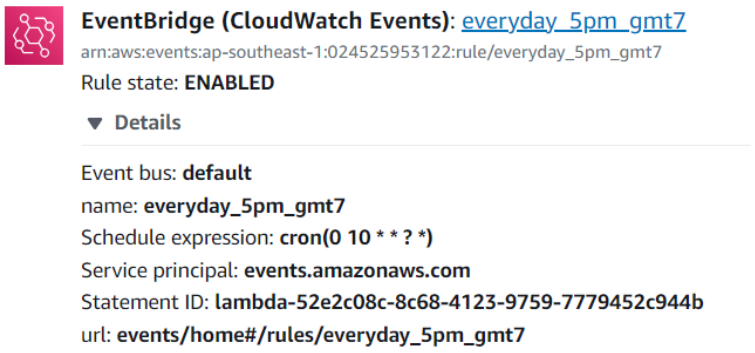
Untuk mengirimkan email, penulis menggunakan library smtplib, smtp atau *Simple Mail Transfer Protokol* sendiri merupakan sebuah protokol untuk mengirimkan pesan apapun ke semua perangkat internet yang menggunakan protokol smtp dengan mudah.

Gambar 3.11 Code SMTP

Pada gambar 3.11 merupakan bagaimana kita dapat mengirimkan email menggunakan protokol SMTP di Lambda. Untuk menggunakan SMTP kita harus memberikan user gmail kita, dan app password gmail kita. Nantinya smtplib akan login dengan gmail kita dan mengirimkan email dari akun kita.

Setelah scripts berhasil berjalan sesuai fungsinya, penulis akan menambahkan suatu *trigger* yang menentukan waktu pelaksanaan skrip. Di AWS, terdapat layanan yang disebut *Cloudwatch Events* yang berfungsisebagai *trigger* untuk Lambda agar menjalankan skrip sesuai keinginan kita. Penulis menentukan waktu pemicu dengan menggunakan CRON, di

mana penulis memberikan CRON *expression* kepada *Cloudwatch Events* sesuai petunjuk CEO untuk menentukan kapan skrip tersebut dijalankan.



Gambar 3.12 Trigger

Contoh *trigger* pada gambar 3.12 adalah *scripts* akan berjalan setiap hari pukul 5 sore gmt +7, dengan CRON *expression* (0 10 * * ? *).

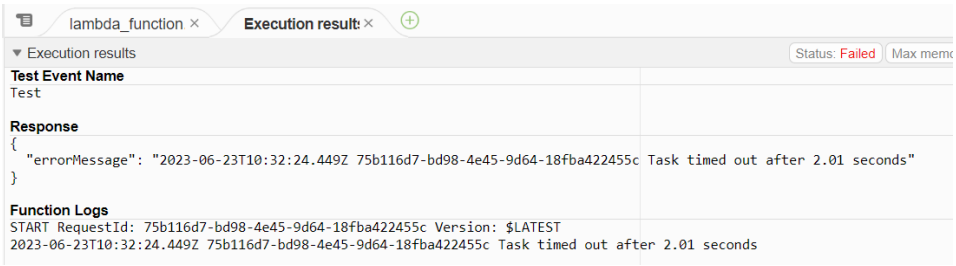
Dari beberapa penjelasan yang sudah dijelaskan berikut merupakan alur sistem *email performance tracking* yang sudah dirancang oleh penulis, yaitu dari saat import product akan terbuat sebuah *summary* yang berisi hal-hal penting yang terjadi saat import product, setelah itu *summary* akan tersimpan di S3 dalam format JSON. Script lambda akan mengambil file *summary* JSON di S3 dan akan mengolahnya, lalu akan mengirimkan email ke setiap tim mengenai isi file summary tersebut ke setiap tim. Lalu isi dari email tersebut akan di save kembali di S3.

4. Testing & Bug Fixing

Setelah menyelesaikan pembuatan script, penulis akan melakukan uji coba terhadap script tersebut. Pengujian yang dilakukan adalah apakah penulis dapat mengambil data JSON dari S3, lalu apakah

penulis dapat mengirimkan email menggunakan lambda. Saat *testing* mengirim email, penulis mencoba menggunakan email pribadi terlebih dahulu sebagai pengirimnya lalu setelah berhasil, penulis baru merubah menggunakan email perusahaan mengirim ke berbagai email sekaligus.

Testing juga dipermudah karena terdapat log dalam AWS Lambda yang menunjukkan apabila terdapat error dalam script penulis,



Gambar 3.13 Lambda Log

Berikut contoh log pada AWS Lambda yang memudahkan penulis untuk mengetahui dimana error pada scripts tersebut.

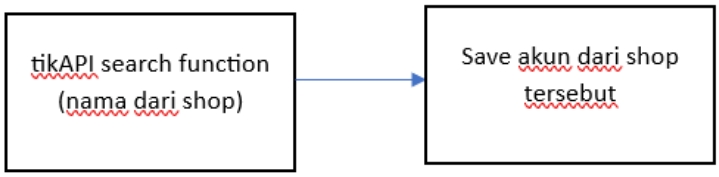
Setelah menemukan kesalahan dalam skrip, penulis akan segera mencari solusi untuk memperbaiki kesalahan tersebut. Setelah itu, penulis akan melakukan pengujian ulang hingga tidak ada kesalahan yang muncul lagi.

5. Meningkatkan Kualitas Data

Dari hasil *email performance tracking* yang dibuat oleh penulis, penulis menemukan bahwa ada data yang kurang lengkap di dalam database Cube Asia. Salah satu masalah yang penulis temukan bahwa pada data tiktok *shop* di database kita, sekitar 80% tiktok *shop* tidak memiliki *account* tiktok yang terkoneksi dengan *shop* tersebut. Kita menyebut kondisi ini dengan *shop with missing account*. Oleh karena itu penulis diminta untuk mencari cara bagaimana kita dapat mengatasi *shop with missing account* ini.

Penulis mengusulkan 2 cara untuk mengatasi *shop with missing account* ini, cara pertama adalah kita akan menggunakan layanan

tikAPI. TikAPI sendiri merupakan sebuah API yang dapat memudahkan kita dalam mengambil data tiktok, didalam tikAPI terdapat fungsi *search* sehingga cara pertama yang saya usulkan adalah menggunakan fungsi *search* tersebut untuk mencari nama shop yang tidak mempunyai *account*, lalu mengambil data *account* pertama yang muncul di *list*.



Gambar 3.14 Alur Algoritma fungsi search tikAPI

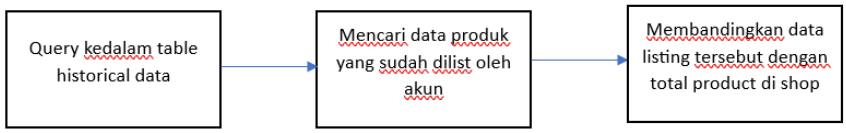
Berikut merupakan code yang digunakan untuk fungsi *search* di tikAPI, *code* ini diambil dari dokumentasi tikAPI.

```
from tikapi import TikAPI, ValidationException, ResponseException

api = TikAPI("myAPIKey")

try:
    response = api.public.search(
        category="general",
        query="lilyachty"
    )

    print(response.json())

    while(response):
        nextCursor = response.json().get('nextCursor')
        print("Getting next items ", nextCursor)
        response = response.next_items()

except ValidationException as e:
    print(e, e.field)

except ResponseException as e:
    print(e, e.response.status_code)
```

Gambar 3.15 Code tikAPI

Cara kedua yang penulis usulkan adalah, dengan cara kita *query* ke database Perusahaan untuk mencari produk-produk yang sudah di list oleh shop-shop tersebut. Di database terdapat 1 table yang menyimpan *historical data* sehingga kita bisa mencari produk produk yang sudah

di list oleh shop tersebut berserta account yang menglistnya. Sehingga kita bisa membandingkan total product yang dimiliki oleh shop tersebut, dengan total product yang dimiliki oleh account yang list produk milik shop tersebut. Apabila total product account mencapai 90% dari total product shop, maka kita dapat menyebut account tersebut merupakan account dari shop tersebut.



Gambar 3.16 Alur algoritma query database

Setelah itu penulis melakukan testing terhadap 500 *shop with missing account* Indonesia untuk mencari akurasi dari 2 algoritma tersebut. Cara penulis untuk menentukan akurasi tersebut, dengan penulis menggunakan aplikasi tiktok lalu menggunakan *feature search* pada tiktok dengan mencari nama dari *shop* tersebut. Lalu penulis akan melihat akun milik *shop* tersebut, apakah sama dengan akun yang penulis temukan dengan menggunakan algoritma yang sudah dirancang. Dengan cara tersebut penulis mendapatkan hasil bahwa untuk algoritma pertama mendapatkan akurasi 76% dan untuk algoritma kedua mendapatkan 56%. Sehingga penulis akan menggunakan algoritma pertama untuk mencari *shop with missing account* agar tidak ada data lagi yang hilang dan kualitas data di database meningkat, begitu pula dengan kualitas *insights* yang akan dihasilkan.

Selama dalam sisa waktu kerja magang penulis di Cube Asia, penulis menemukan bahwa terdapat sekitar 80.000 *shops with missing account* dan setelah penulis menerapkan algoritma pertama atau menggunakan tikAPI, penulis berhasil menemukan sekitar 20.000 *accounts* yang benar dalam kurun waktu 1 bulan. Sehingga penulis

berhasil mencari 25% *accounts* yang benar dari total *shops with missing account*.

6. Weekly Meeting

   Penulis akan berpartisipasi dalam rapat mingguan yang diselenggarakan secara daring setiap hari Selasa dan Jumat. Dalam pertemuan mingguan ini, penulis akan membagikan informasi mengenai tugas-tugas yang telah diselesaikan atau yang akan diselesaikan dalam satu minggu ke depan. Selain itu, pertemuan mingguan ini juga menjadi kesempatan bagi penulis untuk menanyakan data apa yang harus dicari dan mendemonstrasikan *email performance tracking* buatannya kepada tim dan CEO Cube Asia. Penulis juga dapat menggunakan kesempatan ini untuk menanyakan apakah terdapat kebutuhan tambahan dari CEO terkait dengan *email performance tracking* tersebut dan penambahan data pada database.

3.2.3 Kendala dan Solusi yang Ditemukan

Penulis mengalami beberapa tantangan selama menjalani magang di Cube Asia, termasuk kurangnya pemahaman penulis terhadap teknologi yang digunakan, yang menghambat kemajuan dalam pengembangan skrip.

Untuk mengatasi hal tersebut, penulis mengambil langkah dengan mencari informasi terkait dan melakukan uji coba (trial and error) pada skrip menggunakan teknologi yang digunakan. Jika penulis tidak dapat menemukan solusi, langkah selanjutnya adalah bertanya kepada tim atau mentor untuk mendapatkan petunjuk dalam menyelesaikan permasalahan tersebut.

Salah satu kendala yang ditemukan penulis adalah saat mengambil JSON files dengan nama khusus, karena JSON files di S3 tersebut disimpan di folder spesifik dan nama yang berbeda-beda tetapi ada kode khususnya. Sehingga penulis harus mencari cara untuk hanya mengambil folder yang khusus itu saja tanpa harus mengambil keseluruhan folder. Hal ini berhasil

penulis lakukan denganm menggunakan fungsi *filter* dimana fungsi ini akan memisahkan nama-nama file yang memiliki kode khusus didalamnya.