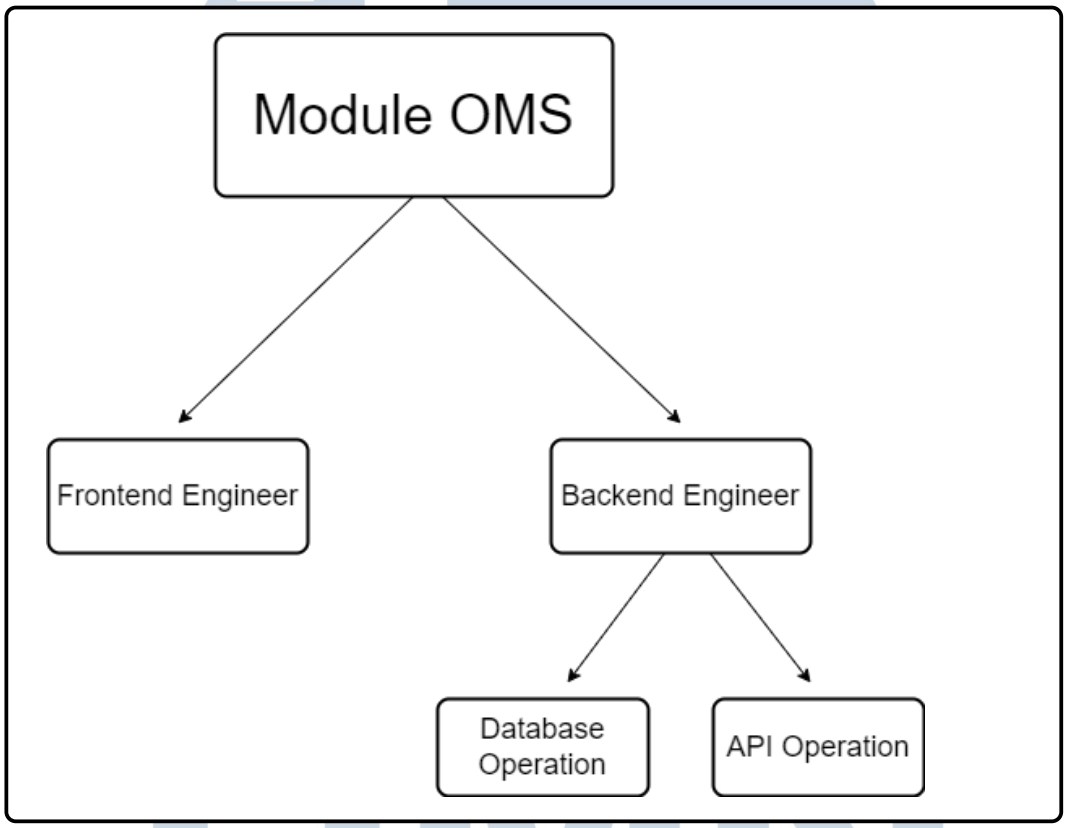
BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Organisasi



Gambar 3.1. Flow Module OMS

Pada pembuatan *module* OMS yang berbasis *website*, terdapat dua bagian dalam pembuatan *module* tersebut. Pada pembuatan tampilan halaman *module* dilakukan oleh bagian *frontend engineer* dengan menggunakan bahasa pemrograman PHP dengan menggunakan *framework* CodeIgniter3 dan menggunakan Visual Studio Code dalam melakukan penulisan kode. *Frontend engineer* juga memiliki tugas untuk mengintegrasikan API agar dapat terlihat pada tampilan, lalu membuat respon untuk API, dan melakukan permintaan kepada API untuk memahami kapan dan bagaimana menggunakan metode HTTP seperti GET, POST, PUT, DELETE. Pada bagian belakang terdapat *backend engineer* yang memiliki tugas untuk mengolah data yang masuk maupun data yang ada

pada *database*. *Backend engineer* dibagi dua yaitu *database operation* dan *Aplication Programming Interface (API) Operation*, *database operation* memiliki tugas mengatur dan mengolah database untuk mengolah kolom, mengatur fungsi pada *database*, dan mengubah nama *database* ataupun menghapus *database* dengan menggunakan Postgresql dan tools Dbeaver untuk mengolah *database*. *API Operation* memiliki tugas membuat API yang akan dipakai pada *module* OMS, menyambungkan fungsi pada *database* terhadap API yang akan dibuat, memastikan validasi yang ada pada API dan dapat digunakan di tampilan *module*. *API Operation* menggunakan *tools* Postman untuk melakukan *testing* API

Kedudukan sebagai *API Operation* pada bagian dari *backend engineer* dalam divisi *application core specialist* memiliki tugas untuk membuat API yang akan digunakan untuk *module* OMS (Order Management System), dimulai dengan melakukan *meeting* terkait data yang dibutuhkan, lalu melakukan latihan dan eksplorasi terkait bahasa pemrograman yang dipakai, melakukan contoh pembuatan API dari *template* yang diberikan agar sesuai dengan standar perusahaan. Melakukan konfigurasi *credential* terhadap *database* yang akan digunakan, melakukan GET API dari *database*, melakukan POST API dengan membuat *function* dengan mengambil data dari file Excel dan dijadikan dalam bentuk JSON, membuat *input validation* dari data excel yang dikirimkan menjadi JSON.

Backend engineer yang dibimbing langsung oleh *IT Head* memiliki tugas untuk mengawasi dan melakukan *checking* terhadap tugas yang diberikan yaitu merancang dan membuat API yang akan dipakai pada *module* Order Management System (OMS). Selain membuat API yang akan dipakai, *backend engineer* juga memastikan bahwa tidak ada kesalahan dalam data yang akan digunakan dan dikirimkan kepada *frontend engineer*.

3.2 Tugas yang Dilakukan

Pelaksanaan kerja magang dapat dilihat pada tabel dibawah ini

Tabel 3.1. Tabel Uraian Pelaksanaan Magang

Minggu ke-	Tugas yang dilakukan
1	Melakukan eksplorasi dan pembelajaran terkait Go-Lang dan *database*
2	Melakukan eksplorasi dan pembelajaran terkait API, Postman. Melakukan diskusi tentang *database* dan tabel yang akan digunakan untuk OMS (Order Management System)

Tabel 3.1. Tabel Uraian Pelaksanaan Magang (lanjutan)

Minggu ke-	Tugas yang dilakukan
3	Melakukan eksplorasi dan pembelajaran terkait *framework* Go-lang yaitu GORM, GIN, dan Logrus
4	Melakukan eksplorasi dan pembelajaran terkait Postgresql
5	Melakukan eksplorasi dan pembelajaran terkait HTTP *response code* dan membuat *custom* API
6	Melakukan eksplorasi dan pembelajaran terhadap *template code* yang diberikan oleh perusahaan dan membuat *custom* API dari *template code* perusahaan
7	Membuat GET API dari *database*, membuat POST API dengan mengambil data dari Excel
8	Menambahkan fungsi untuk melakukan *input validation* terhadap data yang akan dikirimkan oleh API
9	Membuat fungsi untuk GET API berdasarkan kebutuhan yang diperlukan
10	Menambahkan respon pada API dan membuat *input validation*, menambahkan fungsi untuk GET API berdasarkan tanggal
11	Membuat fungsi GET API untuk mengambil data dengan lengkap berdasarkan *primary key*, membuat fungsi POST API untuk menghapus data berdasarkan *primary key* yang diinginkan.
12	Mengubah POST API agar dapat menghapus lebih dari satu data, mengubah format tanggal pada fungsi POST API untuk masuk kedalam *database*, menambahkan validasi pada GET API Mengubah respon pada GET API.

- Minggu pertama sampai kelima dalam pelaksanaan magang adalah

melakukan eksplorasi dan pembelajaran mengenai bahasa pemrograman GO yang dimulai dari konsep bahasa Go, lalu mengenai fungsi pada bahasa Go. Selain itu, melakukan eksplorasi terhadap *database* yang dimulai dari dasar pemahaman *database*, hingga fungsi dalam *database*. Lalu melakukan pembelajaran mengenai API dengan menggunakan bahasa pemrograman GO, selain mengenai API, dilakukan juga eksplorasi dan pembelajaran tentang *database* yang akan digunakan yaitu Postgresql. Setelah melakukan eksplorasi, dilakukan pembuatan *custom* API dengan membuat *database* baru.

- Minggu keenam dalam pelaksanaan kerja magang adalah melakukan eksplorasi dan pembelajaran dengan *template* code yang diberikan oleh perusahaan agar pengerjaannya sesuai dengan standar perusahaan.
- Minggu ketujuh dalam pelaksanaan kerja magang adalah membuat fungsi yang digunakan untuk POST API dengan mengambil data dari excel dan mengkonversi menjadi JSON,
- Minggu kedelapan dalam pelaksanaan magang adalah membuat *input validation* untuk API yang dikirimkan dengan membuat validasi beberapa kolom tidak boleh kosong. Apabila kosong maka akan terdapat peringatan bahwa kolom tersebut tidak boleh kosong.
- Minggu kesembilan dalam pelaksanaan kerja magang adalah membuat fungsi untuk GET API berdasarkan kebutuhan yang diperlukan dengan menambahkan parameter pada pemanggilan API.
- Minggu kesepuluh dalam pelaksanaan kerja magang adalah menambahkan respon pada API, membuat fungsi yang digunakan untuk GET API berdasarkan tanggal awal dan tanggal akhir yang dicari dengan validasi tanggal akhir tidak boleh lebih kecil dari tanggal awal.
- Minggu kesebelas dalam pelaksanaan kerja magang adalah membuat API dengan method GET untuk mengambil data secara lengkap berdasarkan *primary key* yang dicari dengan validasi apabila *primary key* yang diambil tidak ditemukan maka akan memunculkan error. Membuat POST API menghapus data berdasarkan *primary key* yang diinginkan dengan validasi apabila *primary key* yang dicari tidak ada maka akan memunculkan error.

- Minggu keduabelas dalam pelaksanaan kerja magang adalah mengubah POSY API agar dapat menghapus data lebih dari satu berdasarkan *primary key* yang dicari dengan validasi kalau salah satu *primary key* ada yang tidak terdapat dalam *database* maka akan memunculkan respon pesan bahwa *primary key* yang terdapat pada *database* berhasil terhapus namun *primary key* yang tidak ditemukan gagal dihapus.

3.3 Uraian Pelaksanaan Magang

Tugas yang dilakukan dalam proses kerja magang adalah sebagai berikut bisa dilihat pada tabel 3.2

Tabel 3.2. Tabel Ringkasan Tugas Pelaksanaan Magang

Tugas ke-	Pekerjaan yang dilakukan
1	Membuat dan membahas tentang *Entity Relationship Diagram* yang akan dipakai pada Order Management System.
2	Mengembangkan dan membahas tentang *Database Table* yang akan dipakai pada Order Management System dengan berpacu pada *database* yang sudah ada sebelumnya.
3	Membuat fungsi GET API dengan mengambil seluruh data yang ada
4	Membuat fungsi GET API sesuai dengan kebutuhan yang diperlukan
5	Membuat fungsi GET API dengan mengambil data berdasarkan tanggal awal dan tanggal akhir yang diinginkan dengan validasi tanggal akhir tidak boleh lebih kecil dari tanggal awal
6	Membuat fungsi POST API yang mengambil data dari file Excel dijadikan JSON untuk selanjutnya data nya akan dipakai sebagai API
7	Membuat fungsi GET API untuk mengambil data secara detail pada berdasarkan *primary key*
8	Membuat fungsi POST API untuk menghapus data pada berdasarkan *primary key* yang diinginkan
9	Membuat POST API untuk menyimpan dan memperbarui data secara manual

3.3.1 Fungsi Memanggil API

A. API GET

```
GET http://BASE_URL/oms/v1/api/upload-excel
```

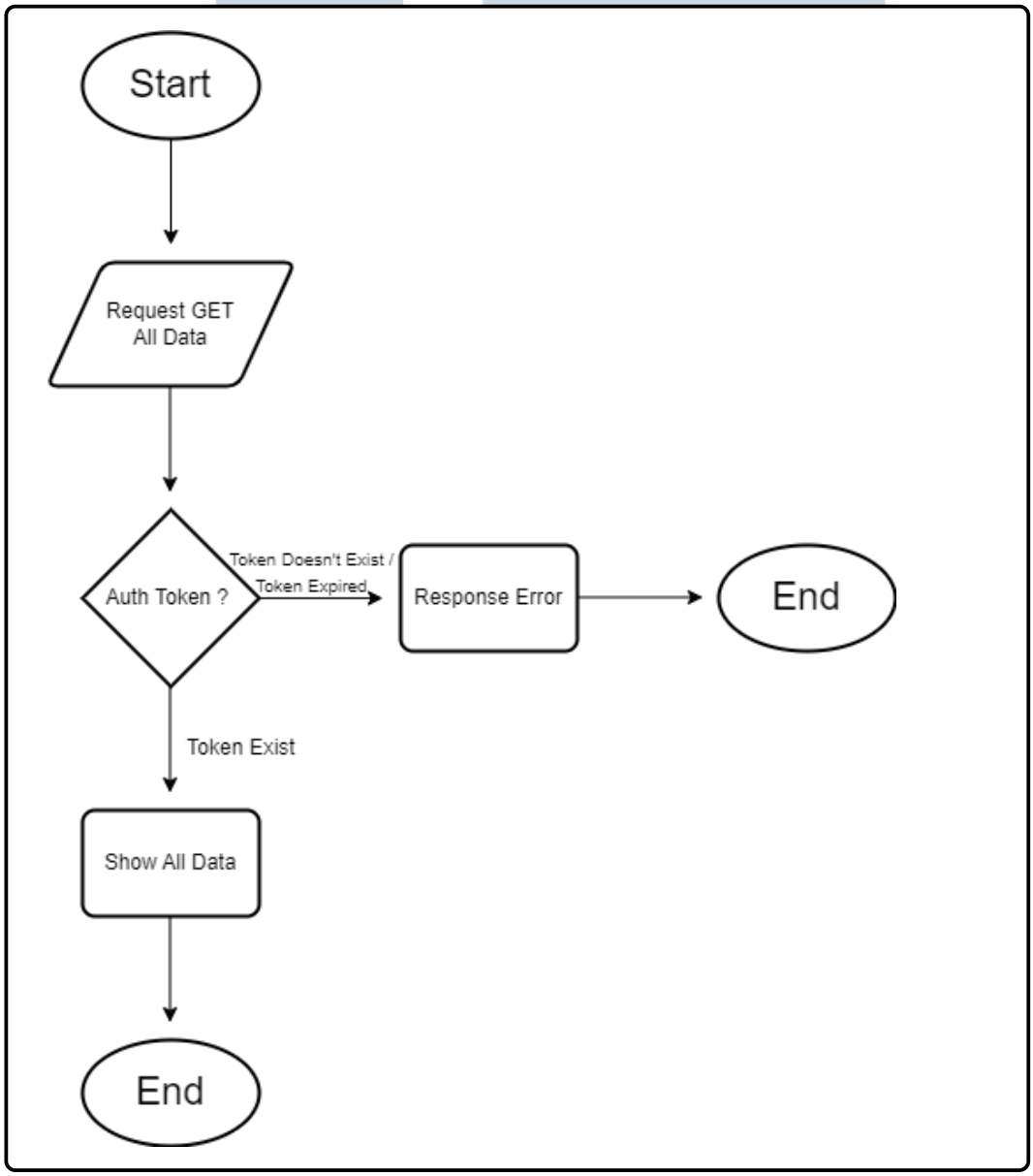
Kode 3.1: Pemanggilan API GET pada Postman


```
{
  "state": 200,
  "message": "Data berhasil dikirimkan",
  "data": [
      {
          "nomor_primary_key" : "11111122222333333444445",
          "nomor_foreign_key" : "000000111112233",
          "nama" : "James Supriadi",
          "umur" : "20 tahun",
          "tanggal_masuk" : "12/02/2024",
          "nomor_telepon" : "081722911234"
      },
      {
          "nomor_primary_key" : "11111122222333333444445",
          "nomor_foreign_key" : "000000111112234",
          "nama" : "Andri Wahyu",
          "umur" : "21 tahun",
          "tanggal_masuk" : "22/02/2024",
          "nomor_telepon" : "08144778412"
      },
      {
          "nomor_primary_key" : "11111122222333333444445",
          "nomor_foreign_key" : "000000111112235",
          "nama" : "Tsabit Danendra",
          "umur" : "20 tahun",
          "tanggal_masuk" : "04/02/2024",
          "nomor_telepon" : "081144559622"
      },
      {
          "nomor_primary_key" : "11111122222333333444445",
          "nomor_foreign_key" : "000000111112236",
          "nama" : "Yohannes Keane",
          "umur" : "17 tahun",
          "tanggal_masuk" : "11/02/2024",
          "nomor_telepon" : "0877885544128"
      }
```

```
37        ]
38 }
```

Kode 3.2: Respon API GET

Pada pemanggilan semua data melalui method GET menggunakan perintah ”{{BASE_URL}}/oms/v1/api/upload-excel” dengan *base url* mengambil dari server yang dijalankan. Respon dari fungsi ini adalah mengembalikan dan menampilkan seluruh data yang terdapat pada *database*.



Gambar 3.2. Flow API GET

Flow pada fungsi ini adalah dengan memasukkan *request* dengan *endpoint* yang diinginkan, lalu akan melakukan cek apakah terdapat token yang terdaftar atau tidak, apabila tokennya terdaftar maka akan langsung menampilkan semua data, apabila tokennya tidak ada ataupun sudah *expired*, maka akan respon eror yang dikembalikan

B. API GET berdasarkan tanggal

```
GET http://BASE_URL/oms/v1/api/upload-excel?tanggal_awal
  =2024-02-12&tanggal_akhir=2024-02-22
```

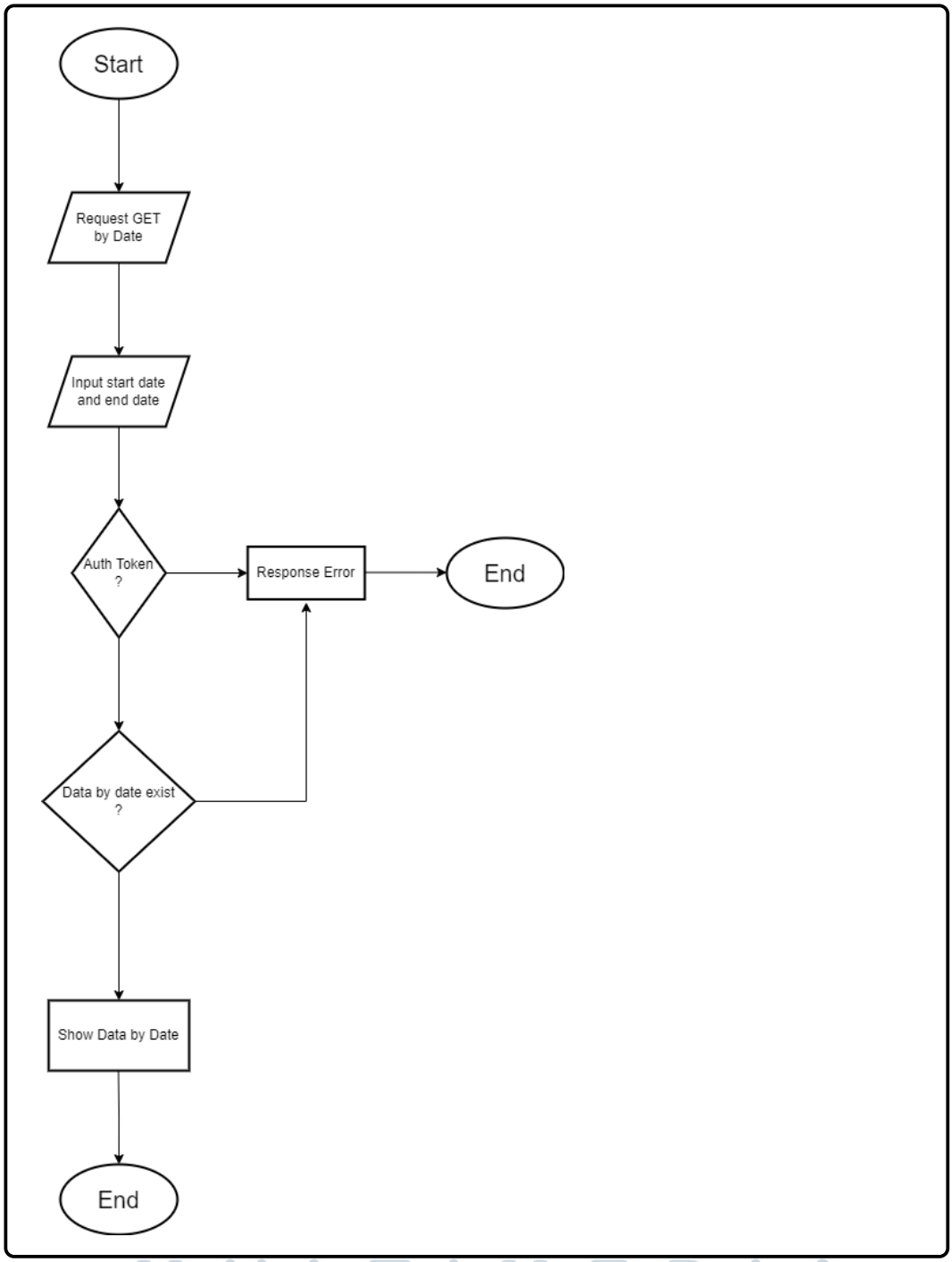
Kode 3.3: Pemanggilan API GET berdasarkan tanggal pada Postman

```
{
  "state": 200,
  "message": "Data berdasarkan tanggal berhasil ditemukan",
  "data": [
      {
          "nomor_primary_key" : "11111122222333333444445",
          "nomor_foreign_key" : "000000111112233",
          "nama" : "James Supriadi",
          "umur" : "20 tahun",
          "tanggal_masuk" : "12/02/2024",
          "nomor_telepon" : "081722911234"
      },
      {
          "nomor_primary_key" : "11111122222333333444445",
          "nomor_foreign_key" : "000000111112234",
          "nama" : "Andri Wahyu",
          "umur" : "21 tahun",
          "tanggal_masuk" : "22/02/2024",
          "nomor_telepon" : "08144778412"
      }
  ]
}
```

Kode 3.4: Respon API GET berdasarkan tanggal

Pada pemanggilan data berdasarkan tanggal melalui GET menggunakan perintah ”{{BASE_URL}}/oms/v1/api/upload-excel” ditambahkan parameter *tanggal_awal,* dan *tanggal_akhir* sehingga perintahnya menjadi ”{{BASE_URL}}/oms/v1/api/upload-order/?branch=&tanggal_awal=&tanggal_akhir=” dengan *base url* mengambil

dari server yang dijalankan. Fungsi ini mengambil tanggal awal, dan tanggal akhir dari data yang dicari. Respon dari fungsi ini adalah mengembalikan dan menampilkan data berdasarkan tanggal yang terdapat pada *database*. Apabila tanggal yang dicari tidak ada, maka responnya adalah mengirimkan pesan bahwa tanggal yang dicari tidak ditemukan.



Gambar 3.3. Flow API GET by Date

Flow pada fungsi ini adalah dengan memasukkan *request* dengan *endpoint* yang diinginkan dan memasukkan data dengan berisikan tanggal awal dan tanggal akhir yang ingin dicari, lalu akan melakukan cek apakah terdapat token yang terdaftar atau tidak, apabila tokennya terdaftar dan data berdasarkan tanggal yang

dicari ada maka akan langsung menampilkan semua data berdasarkan tanggal yang dicari, apabila tokennya tidak ada ataupun sudah *expired* dan data berdasarkan tanggal yang dicari tidak ada, maka akan respon eror yang dikembalikan.

C. API POST berdasarkan *file* Excel

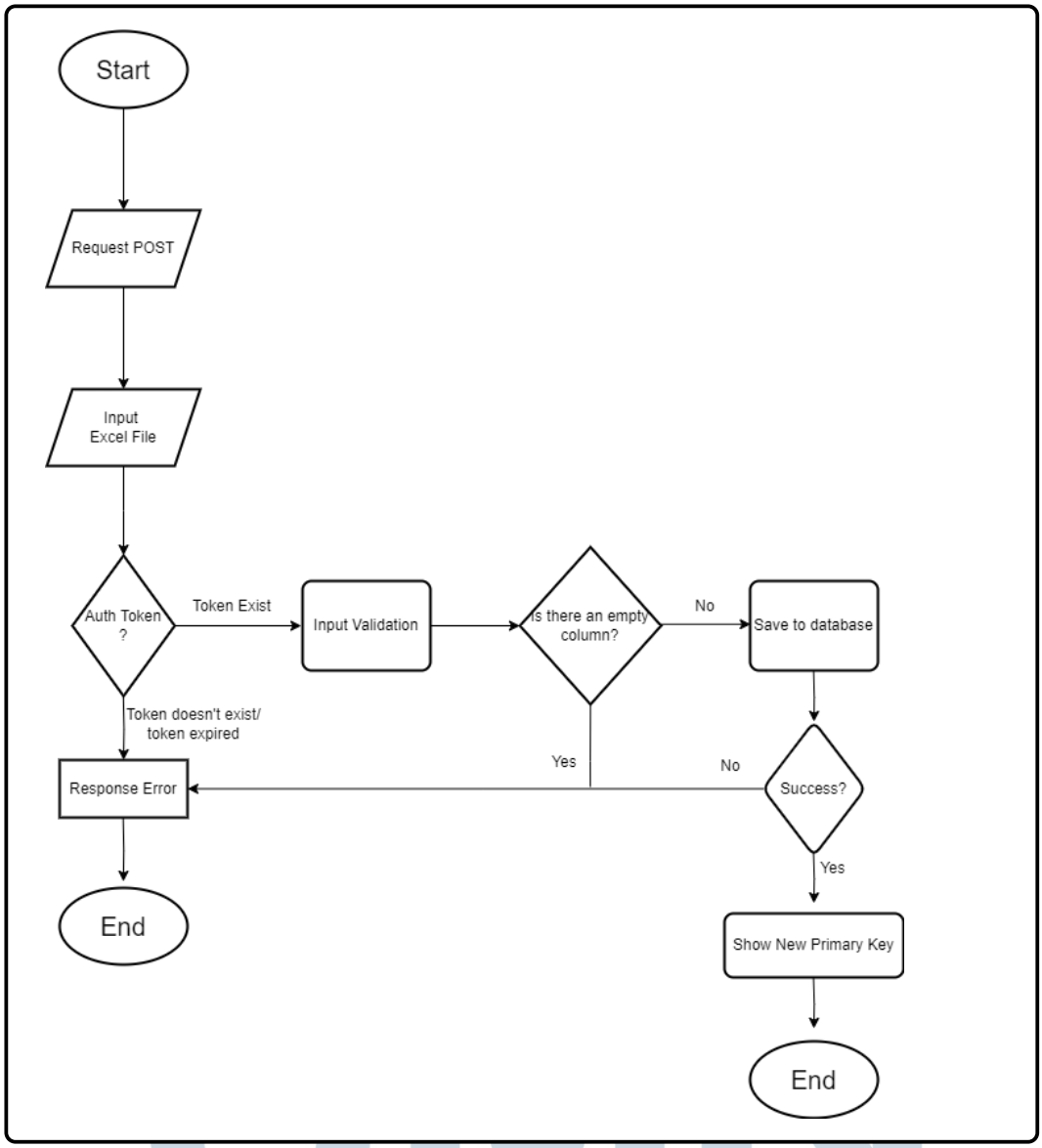
```
POST http://BASE_URL/oms/v1/api/upload-excel
```

Kode 3.5: Pemanggilan API POST berdasarkan file Excel pada Postman

```
{
    "state": 200,
    "message": "Data berhasil dikirimkan",
    "data": {
        "nomor_primary_key" : "000000111112232"
    }
}
```

Kode 3.6: Respon API POST berdasarkan file Excel

Pada pemanggilan semua data melalui POST menggunakan perintah ”{{BASE_URL}}/oms/v1/api/upload-excel” dengan *base url* mengambil dari server yang dijalankan. Respon dari fungsi ini adalah mengembalikan dan menampilkan nomor *primary key* yang baru ketika sudah masuk kedalam *database*.



Gambar 3.4. Flow API POST from Excel

Flow pada fungsi ini adalah dengan memasukkan *request* dengan *endpoint* yang diinginkan dan memasukkan *file* Excel yang telah berisikan data yang dibutuhkan, lalu akan melakukan cek apakah terdapat token yang terdaftar atau tidak, apabila tokennya terdaftar maka akan masuk kedalam pengecekan yang terdapat pada *input validation*, setelah melakukan pengecekan maka akan melakukan proses untuk menyimpan data kedalam *database*, jika sukses akan menampilkan respon berhasil dengan menampilkan nomor *primary key* yang baru. Apabila tokennya tidak ada ataupun sudah *expired* dan ada salah satu kolom yang kosong pada *input validation*, maka akan respon eror yang dikembalikan.

D. API GET detail berdasarkan *foreign key*

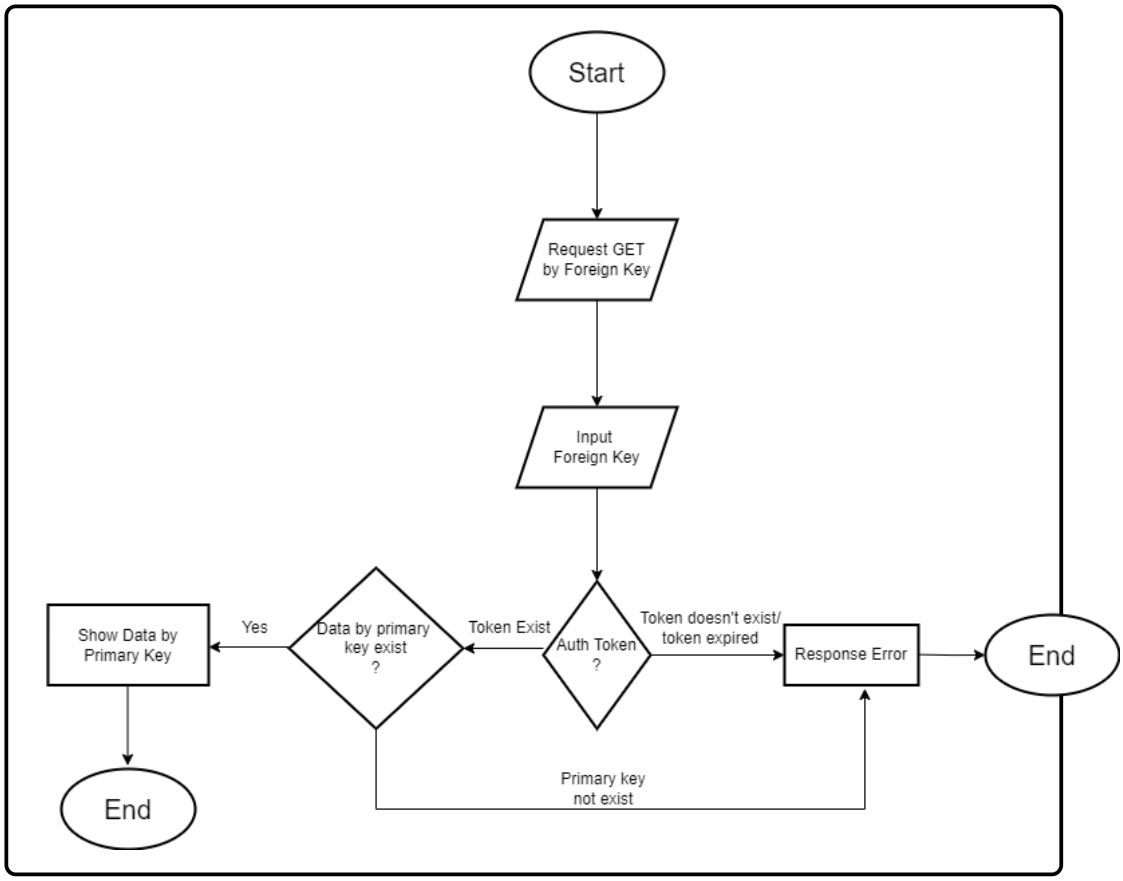
```
GET http://BASE_URL/oms/v1/api/upload-excel/primarykey
  /000000111112234
```

Kode 3.7: Pemanggilan API GET berdasarkan foreign key pada Postman

```
{
    "state": 200,
    "message": "Data berdasarkan foreign key berhasil ditemukan",
    "data": {
            "nomor_primary_key" : "11111122222333333444447",
            "nomor_foreign_key" : "000000111112234",
            "nama" : "Tsabit Danendra",
            "umur" : "20 tahun",
            "tanggal_masuk" : "04/02/2024",
            "nomor_telepon" : "081144559622"
    }
}
```

Kode 3.8: Respon API GET berdasarkan foreign key pada Postman

Pada pemanggilan data berdasarkan *foreign key* melalui GET menggunakan perintah ”{{BASE_URL}}/oms/v1/api/upload-excel/detail” dengan menggunakan *dynamic route* sehingga perintahnya menjadi ”{{BASE_URL}}/oms/v1/api/upload-excel/detail/:foreign-key” dengan *base url* mengambil dari server yang dijalankan. Respon dari fungsi ini adalah mengembalikan dan menampilkan data berdasarkan *foreign key* yang terdapat pada *database*. Apabila *foreign key* yang dicari tidak ditemukan, maka akan mengembalikan respon eror.



Gambar 3.5. Flow API GET by foreign key

Flow pada fungsi ini adalah dengan memasukkan *request* dengan *endpoint* yang diinginkan dan memasukkan data dengan berisikan nomor *foreign key* yang ingin dicari, lalu akan melakukan cek apakah terdapat token yang terdaftar atau tidak, apabila tokennya terdaftar dan data berdasarkan *foreign key* yang dicari ada maka akan langsung menampilkan semua data berdasarkan *foreign key* yang dicari, apabila tokennya tidak ada ataupun sudah *expired* dan data berdasarkan *foreign key* yang dicari tidak ada, maka akan respon eror yang dikembalikan.

E. API POST menghapus data berdasarkan *foreign key*

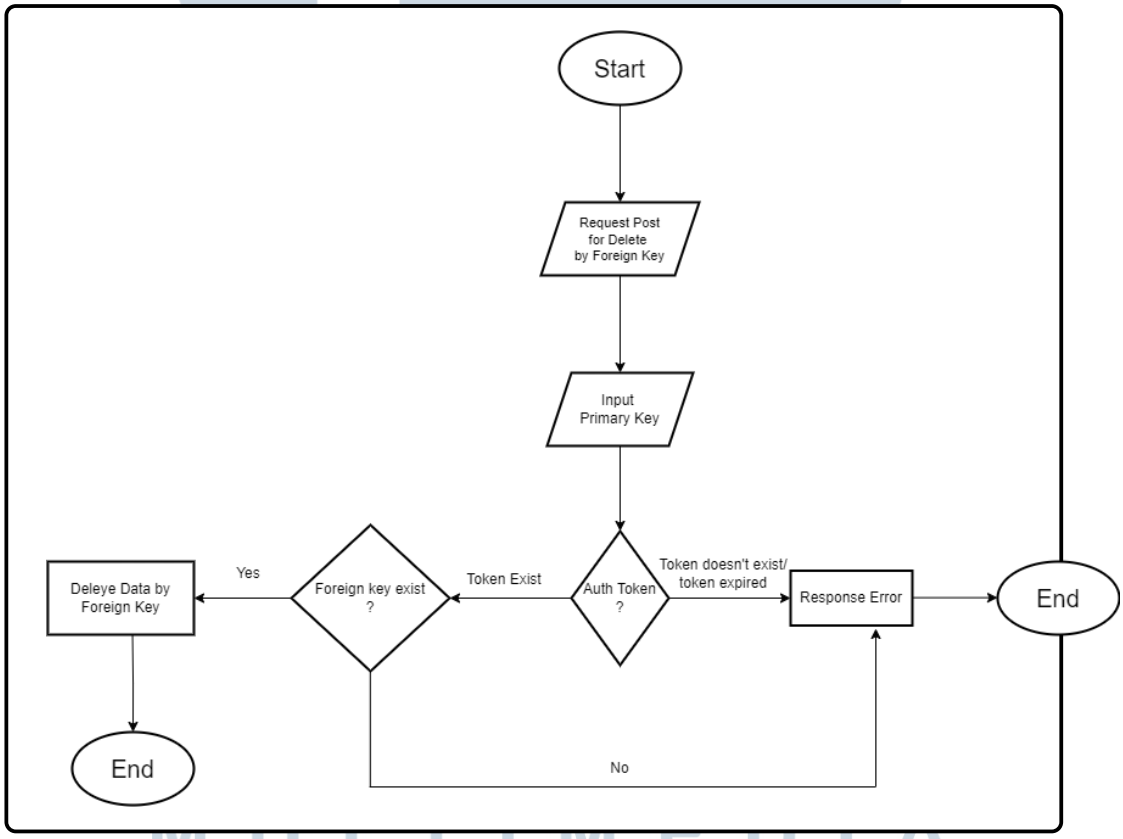
```
POST http://BASE_URL/oms/v1/api/upload-excel-delete
  /000000111112233
```

Kode 3.9: Pemanggilan API POST untuk menghapus berdasarkan foreign key pada Postman

```
{
    "state": 200,
    "message": "Data berhasil dihapus",
    "data": []
}
```

Kode 3.10: Respon API POST untuk menghapus berdasarkan foreign key pada Postman

Pada penghapusan data berdasarkan *primary key* melalui POST menggunakan perintah ”{{BASE_URL}}/oms/v1/api/upload-excel-delete” dengan *base url* mengambil dari server yang dijalankan dan memasukkan *primary key* yang ingin dihapus. Respon dari fungsi ini adalah mengembalikan dan menampilkan pesan bahwa data berdasarkan *primary key* yang terdapat pada *database* sudah berhasil terhapus.



Gambar 3.6. Flow API POST for delete by foreign key

Flow pada fungsi ini adalah dengan memasukkan *request* dengan *endpoint* yang diinginkan dan memasukkan data dengan berisikan nomor *foreign key* yang ingin dihapus, lalu akan melakukan cek apakah terdapat token yang terdaftar atau

tidak, apabila tokennya terdaftar dan data berdasarkan *foreign key* yang ingin dihapus ada maka akan langsung menghapus data berdasarkan *foreign key* yang dicari, apabila tokennya tidak ada ataupun sudah *expired* dan data berdasarkan *foreign key* yang dicari tidak ada, maka akan respon eror yang dikembalikan.

F. API menyimpan dan memperbarui data secara manual

```
POST http://BASE_URL/oms/v1/api/upload-excel/detail
```

Kode 3.11: Pemanggilan API POST untuk memperbarui data pada Postman

```
{
    "nomor_foreign_key" : "000000111112233",
    "nama" : "Tsabit Danendra",
    "umur" : "20 tahun",
    "tanggal_masuk" : "04/02/2024",
    "nomor_telepon" : "081144559622"
}
```

Kode 3.12: Data sebelum diperbarui

```
{
    "nomor_foreign_key" : "000000111112233",
    "nama" : "Tsabit Danendra",
    "umur" : "20 tahun",
    "tanggal_masuk" : "04/02/2024",
    "nomor_telepon" : "087809665548"
}
```

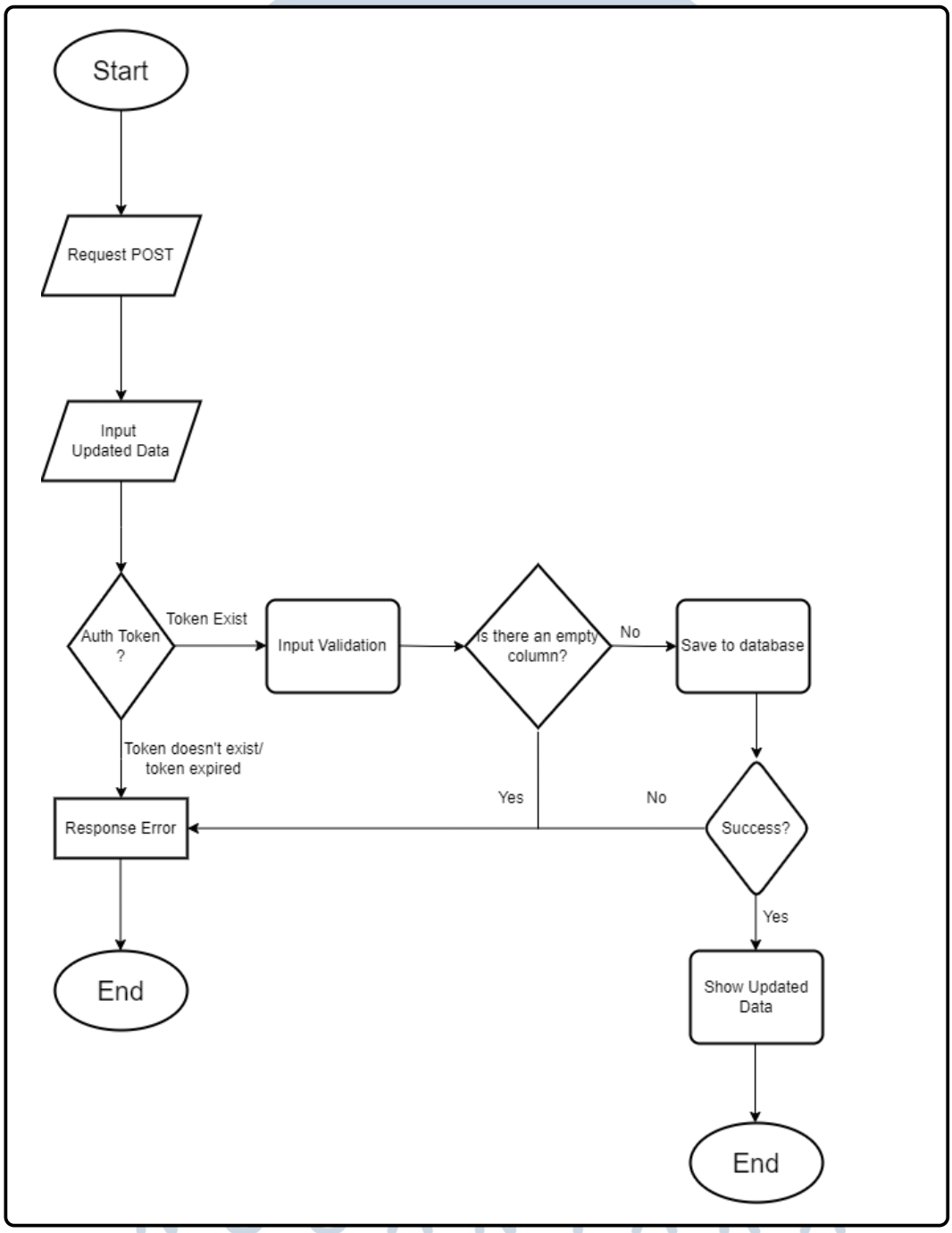
Kode 3.13: Pengisian data yang akan diperbarui pada API POST pada

```
{
    "state": 200,
    "message": "Data dengan nomor foreign key : 000000111112233
   berhasil diperbarui",
    "data": []
}
```

Kode 3.14: Pengisian data yang akan diperbarui pada API POST pada

Fungsi ini dapat melakukan pembaruan pada data berdasarkan *foreign key* yang diinginkan. Cara menggunakan fungsi ini adalah dengan mengisi data yang ingin diubah dengan memasukkan nomor *foreign key* nya, lalu akan dilakukan pengecekan apakah nomor *foreign key* tersebut sudah terdapat dalam database atau

belum untuk melakukan pembaruan. Respon dari fungsi ini adalah mengembalikan dan menampilkan data yang telah diperbarui dalam *database* dan sudah termasuk data yang valid.



Gambar 3.7. Flow API POST untuk update data

Flow pada fungsi ini adalah dengan memasukkan *request* dengan *endpoint*

yang diinginkan dan memasukkan data yang ingin diubah dengan memasukkan *foreign key* dalam data tersebut, lalu akan dilakukan pengecekan pada *database* apakah nomor *foreign key* tersebut terdapat pada *database* atau tidak, apabila terdaftar maka data akan segera diperbarui. Saat tokennya tidak ada ataupun sudah *expired* dan data yang ingin diubah berdasarkan nomor *foreign key* tidak ditemukan, atau ada kesalahan pada *input validation* yang, maka akan respon eror yang dikembalikan.

3.3.2 Validasi API

Terdapat dua jenis validasi error pada API yang digunakan

A. Validasi data tidak boleh kosong atau duplikat

```
1  KOLOM_REQUIRED = "Kolom tidak boleh kosong"
2  DATA_EXIST = "Data sudah ada"
```

Kode 3.15: Validasi API terhadap kolom kosong dan sudah ada

Validasi ini digunakan terutama ketika melakukan POST dalam *upload* Excel dan melakukan pembaruan pada *upload* Excel, karena apabila ada kolom yang kosong maka akan memunculkan pesan *error* sesuai data yang tidak ada pada saat melakukan POST. HTTP response pada validasi ini adalah 400 yang berarti *bad request*.

B. Validasi data *error* dan tidak ditemukan

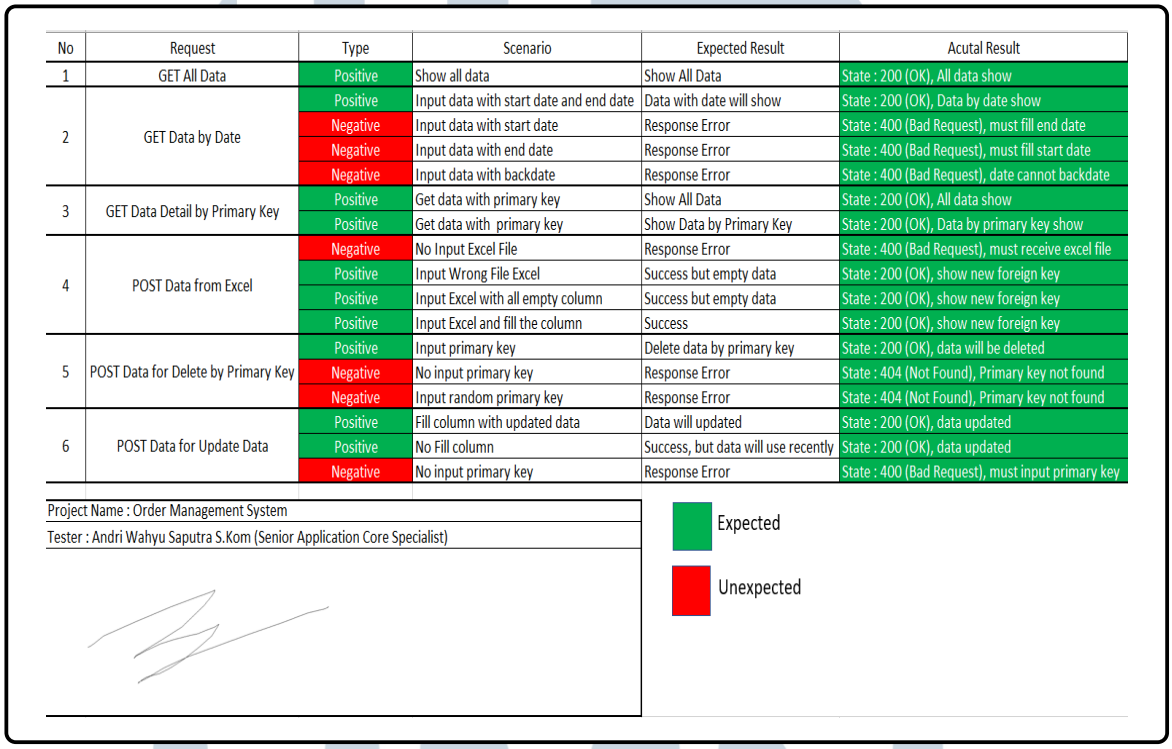
```
1 DATABASE_ERROR         = "Gagal masukkan data kedalam database"
2 DATE_NOTFOUND          = "Data yang dicari tidak ditemukan"
3 DATE_WRONG             = "Format tanggal salah"
4 DATE_ERROR             = "Data berdasarkan tanggal salah"
5 DELETE_FAILED          = "Data tidak berhasil dihapus"
6 DELETE_NOTFOUND        = "Data tidak berhasil dihapus dikarenakan
     tidak ditemukan"
7 UPDATE_FAILED          = "Gagal memperbarui data"
```

Kode 3.16: Validasi API terhadap data error dan tidak ditemukan

Validasi ini digunakan apabila ada data yang dicari tidak ditemukan dalam *database*, tidak dapat memasukkan data kedalam *database*, format tanggal salah.

Validasi ini banyak digunakan terutama pada *method* GET untuk memanggil data berdasarkan parameter yang diinginkan.

3.3.3 *Testing* API



No	Request	Type	Scenario	Expected Result	Acutal Result
1	GET All Data	Positive	Show all data	Show All Data	State : 200 (OK), All data show
2	GET Data by Date	Positive	Input data with start date and end date	Data with date will show	State : 200 (OK), Data by date show
		Negative	Input data with start date	Response Error	State : 400 (Bad Request), must fill end date
		Negative	Input data with end date	Response Error	State : 400 (Bad Request), must fill start date
		Negative	Input data with backdate	Response Error	State : 400 (Bad Request), date cannot backdate
3	GET Data Detail by Primary Key	Positive	Get data with primary key	Show All Data	State : 200 (OK), All data show
		Positive	Get data with primary key	Show Data by Primary Key	State : 200 (OK), Data by primary key show
4	POST Data from Excel	Negative	No Input Excel File	Response Error	State : 400 (Bad Request), must receive excel file
		Positive	Input Wrong File Excel	Success but empty data	State : 200 (OK), show new foreign key
		Positive	Input Excel with all empty column	Success but empty data	State : 200 (OK), show new foreign key
		Positive	Input Excel and fill the column	Success	State : 200 (OK), show new foreign key
5	POST Data for Delete by Primary Key	Positive	Input primary key	Delete data by primary key	State : 200 (OK), data will be deleted
		Negative	No input primary key	Response Error	State : 404 (Not Found), Primary key not found
		Negative	Input random primary key	Response Error	State : 404 (Not Found), Primary key not found
6	POST Data for Update Data	Positive	Fill column with updated data	Data will updated	State : 200 (OK), data updated
		Positive	No Fill column	Success, but data will use recently	State : 200 (OK), data updated
		Negative	No input primary key	Response Error	State : 400 (Bad Request), must input primary key

Project Name : Order Management System

Tester : Andri Wahyu Saputra S.Kom (Senior Application Core Specialist)

Expected

Unexpected

Gambar 3.8. Testing API oleh Senior Application Core Specialist

Testing API yang diberikan dari *Senior Application Core Specialist* dengan melakukan beberapa tes terhadap API yang telah dibuat. Berdasarkan hasil dari testing tersebut menunjukkan bahwa semua tes yang sudah dilakukan sesuai dengan ekspetasi dari tes tersebut, dan dapat dipastikan bahwa API dapat berjalan sesuai dengan permintaan yang diberikan.

3.4 Kendala dan Solusi yang Ditemukan

3.4.1 Kendala

Kendala yang ditemukan dalam proses kerja magang pada B-Log adalah sebagai berikut :

- Terjadi *conflict* pada *file* config.yml dalam melakukan *merge request* pada Gitlab sehingga server menjadi *error* yang membuat API tidak dapat terpanggil oleh *module* OMS

3.4.2 Solusi

- Melakukan *revert* terhadap data yang sudah di *push* pada *branch development* dan melakukan *backup* terhadap *file* config.yml. Hal ini dilakukan untuk mengembalikan keadaan sebelum terjadinya *conflict*.
- Melakukan *clone* ulang dari repositori untuk mendapatkan versi terbaru dari *branch development* yang bersih dari *conflict*. Proses ini memastikan bahwa tidak ada data yang bertabrakan atau korup.
- Mengidentifikasi bagian dari *file* config.yml yang menyebabkan *conflict* dan melakukan penyesuaian secara manual. Ini melibatkan penghapusan atau penggabungan bagian yang konflik sehingga sesuai dengan kebutuhan semua pihak yang terlibat.
- Melakukan *merge request* baru dengan config.yml yang sudah disesuaikan. Sebelum melakukan *merge*, lakukan pengujian lokal untuk memastikan tidak ada *error* yang muncul.
- Menyediakan dokumentasi yang jelas mengenai struktur dan aturan perubahan pada config.yml. Dokumentasi ini dapat membantu tim dalam memahami dan mengikuti prosedur yang benar saat melakukan perubahan.