

BAB III

PELAKSANAAN KERJA

3.1 Kedudukan dan Koordinasi

Selama menjalani kegiatan magang di PT Phillip Sekuritas Indonesia, penulis ditempatkan secara strategis pada posisi sebagai *Machine Learning Engineer Intern*. Posisi ini berada di bawah naungan Divisi Teknologi dan Pengembangan Sistem, sebuah unit krusial yang bertanggung jawab atas inovasi digital dan pemeliharaan infrastruktur teknologi perusahaan. Penempatan ini membawa penulis masuk ke dalam lingkungan kerja profesional yang dinamis, di mana efisiensi dan adaptabilitas menjadi prioritas utama. Untuk mendukung ritme kerja tersebut, perusahaan mengadopsi metodologi pengembangan perangkat lunak yang sistematis dan kolaboratif, yaitu *Scrum Framework*, yang merupakan bagian dari pendekatan *Agile*.



Gambar 3. 1 Alur Proses Scrum

Penerapan metode Scrum di divisi ini memastikan bahwa setiap proyek, termasuk inisiasi dan pengembangan model *Machine Learning*, tidak dikerjakan secara isolatif, melainkan melalui tahapan yang terstruktur, transparan, dan terukur. Sebagaimana diilustrasikan pada Gambar 3.1, siklus kerja dimulai dari pemecahan

Product Backlog menjadi tugas-tugas yang lebih kecil dalam *Sprint Backlog*. Dalam konteks pekerjaan penulis, hal ini melibatkan penerjemahan kebutuhan bisnis menjadi tugas teknis, seperti pengumpulan data, *preprocessing*, pelatihan model, hingga evaluasi performa algoritma.

Siklus kerja utama dijalankan dalam periode waktu yang telah ditentukan atau dikenal sebagai *Sprint*. Selama periode *Sprint* ini, penulis terlibat aktif dalam serangkaian seremoni Scrum yang bertujuan untuk menjaga keselarasan tim:

1. ***Sprint Planning***: Tahap awal di mana penulis berkoordinasi dengan tim untuk menentukan target pengerjaan model dan estimasi waktu yang dibutuhkan
2. ***Daily Stand-up Meeting***: Pertemuan harian singkat untuk melaporkan progres pelatihan model, mendiskusikan hambatan teknis (*blockers*), dan menyinkronkan langkah dengan anggota tim lain.
3. ***Sprint Review***: Sesi demonstrasi hasil kerja (*product increment*) kepada pemangku kepentingan (*stakeholders*) untuk mendapatkan umpan balik langsung mengenai akurasi atau relevansi model yang dikembangkan
4. ***Sprint Retrospective***: Sesi evaluasi internal tim untuk mengidentifikasi hal-hal yang sudah berjalan baik dan aspek yang perlu diperbaiki pada *Sprint* berikutnya

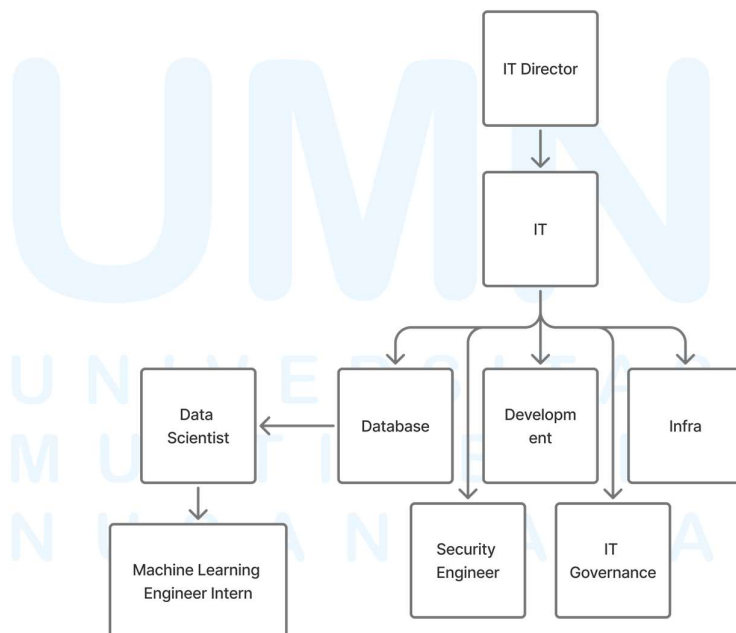
Melalui partisipasi aktif dalam siklus ini, penulis mempelajari pola kerja industri yang sangat menekankan pada komunikasi terbuka, transparansi progres, dan prinsip *continuous improvement*.

3.1.1 Kedudukan

Kedudukan penulis di PT Phillip Sekuritas Indonesia adalah sebagai *Machine Learning Engineer Intern* yang ditempatkan pada divisi *Information Technology* (IT), khususnya dalam tim *Database*. Dalam kedudukan ini, penulis memiliki peran sebagai pendukung dalam kegiatan pengembangan solusi berbasis data, mulai dari proses pengumpulan, pembersihan, dan pengolahan data, hingga tahap analisis dan pemodelan menggunakan metode

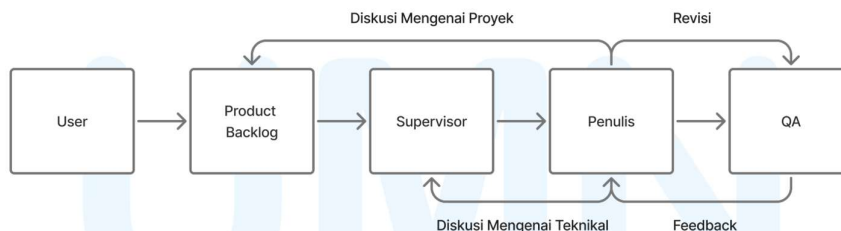
machine learning. Aktivitas tersebut bertujuan untuk membantu perusahaan dalam memperoleh *insight* yang relevan serta meningkatkan efisiensi proses bisnis, seperti analisis data pasar, prediksi tren, dan otomasi proses internal yang berbasis teknologi.

Sebagai seorang *intern*, penulis melaksanakan tugas dan tanggung jawab di bawah bimbingan langsung *Data Scientist*, serta menjalin koordinasi dengan tim IT dan unit kerja terkait lainnya. Kolaborasi ini dilakukan untuk memastikan bahwa pengembangan dan implementasi model *machine learning* dapat berjalan sesuai dengan kebutuhan sistem perusahaan, standar teknis yang berlaku, serta selaras dengan tujuan bisnis PT Phillip Sekuritas Indonesia. Melalui kedudukan tersebut, penulis juga memperoleh pengalaman praktis dalam penerapan ilmu *machine learning* di lingkungan industri keuangan secara profesional dan terstruktur. Berikut contoh bagan struktur organisasi divisi tempat penulis ditempatkan di PT Phillip Sekuritas Indonesia:



3.1.2 Koordinasi

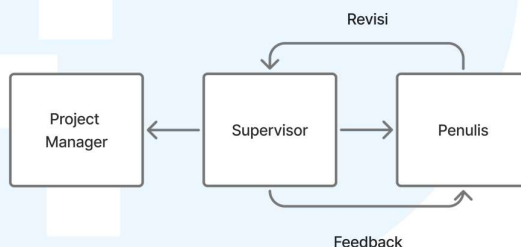
Pada proyek pengembangan website, penulis melakukan koordinasi dengan tim *Project Manager* (PM) dan tim *Quality Assurance* (QA) dengan mengacu pada metode kerja *Scrum*. Proses koordinasi diawali dengan penyusunan dan pembahasan *product backlog*, yang berisi daftar kebutuhan sistem, fitur, serta perbaikan yang akan dikembangkan. *Backlog* tersebut disusun berdasarkan kebutuhan pengguna dan arahan dari PM, kemudian diprioritaskan untuk menentukan pekerjaan yang akan dikerjakan pada setiap *sprint*.



Selama pelaksanaan *sprint*, penulis berkoordinasi secara rutin dengan PM untuk melaporkan perkembangan pekerjaan, mendiskusikan kendala teknis, serta memastikan bahwa pengembangan website berjalan sesuai dengan tujuan dan jadwal yang telah ditetapkan. Setelah tahap pengembangan selesai, penulis melakukan koordinasi dengan tim QA untuk proses pengujian sistem. Pada tahap ini, tim QA melakukan pengujian fungsionalitas dan

kualitas website, sementara penulis menindaklanjuti hasil pengujian dengan melakukan perbaikan atau penyesuaian sesuai dengan temuan yang diberikan.

Sementara itu, pada proyek analisis data nasabah, penulis melakukan koordinasi dengan tim *Database* dan *Data scientist*. Koordinasi difokuskan pada penentuan kebutuhan analisis, pemahaman karakteristik data nasabah, serta diskusi terkait hasil analisis dan model *machine learning* yang dikembangkan. Melalui koordinasi ini, penulis memastikan bahwa hasil analisis yang dihasilkan dapat memberikan *insight* yang relevan dan mendukung proses pengambilan keputusan bisnis perusahaan.



Gambar 3. 4 Bagan Alur Koordinasi (Machine Learning)

Secara keseluruhan, alur koordinasi yang dilakukan oleh penulis bertujuan untuk menjaga kelancaran proses kerja, meningkatkan efektivitas kolaborasi antar tim, serta meminimalkan kesalahan komunikasi. Selain itu, koordinasi juga melibatkan pembimbing lapangan sebagai pihak yang memberikan arahan, evaluasi, dan persetujuan terhadap hasil pekerjaan yang telah dilakukan oleh penulis.

3.2 Tugas yang Dilakukan

Selama menjalani kegiatan magang, peserta magang diberikan berbagai tanggung jawab yang disampaikan melalui sistem backlog perusahaan. Setiap tugas yang diterima berkaitan dengan proses pengembangan aplikasi berbasis web,

terutama pada pengembangan *Single Page Application (SPA)* yang merupakan pendekatan modern dalam membangun antarmuka web yang dinamis, responsif, dan efisien. Dalam hal ini, peserta magang diharapkan memahami konsep dasar arsitektur front-end, struktur komponen, serta mekanisme integrasi dengan layanan *Application Programming Interface (API)* yang digunakan perusahaan.

Selain bekerja pada bagian pengembangan aplikasi, peserta magang juga memiliki peran penting dalam pekerjaan yang melibatkan *machine learning (ML)* dan analisis data. Tugas-tugas tersebut mencakup proses pengumpulan data, pembersihan data, eksplorasi data, hingga pembuatan model seperti *clustering* untuk mengelompokkan data nasabah berdasarkan pola tertentu, serta *sentiment analysis* untuk memetakan opini pengguna terhadap layanan perusahaan. Proses ini membutuhkan pemahaman terhadap berbagai algoritma dan teknik statistik agar hasil analisis dapat memberikan informasi yang akurat dan relevan.

Dalam pelaksanaan tugasnya, peserta magang mengikuti alur kerja (*workflow*) yang ditetapkan perusahaan, dimulai dari analisis kebutuhan yang diajukan oleh *user*, perencanaan pengerjaan, pengembangan fitur, proses *testing*, hingga penyusunan dokumentasi teknis. Setiap progres pekerjaan dilaporkan secara berkala melalui *meeting* rutin agar dapat dipantau oleh supervisor dan tim terkait. Peserta magang juga diharapkan mampu bekerja secara kolaboratif, berdiskusi, serta memberikan solusi ketika menghadapi kendala teknis.

Tabel 3. 1 Detail Pekerjaan yang Dilakukan

No.	Minggu	Proyek	Keterangan
1	1-2	Membuat Clustering Data Nasabah	Proses clustering nasabah dilakukan dengan tujuan untuk mengelompokkan nasabah ke dalam beberapa cluster berdasarkan karakteristik dan pola data yang dimiliki. Melalui proses ini, diharapkan dapat diketahui pembagian cluster nasabah yang memiliki kesamaan perilaku atau atribut tertentu, sehingga hasil

No.	Minggu	Proyek	Keterangan
			pengelompokan tersebut dapat digunakan sebagai dasar dalam analisis lebih lanjut serta mendukung pengambilan keputusan yang lebih tepat
2	3-4	Membuat Dashboard Hasil Clustering Data Nasabah	Menyusun dashboard berdasarkan hasil analisis clustering yang telah diperoleh dari proses FM Analisis, sehingga informasi yang dihasilkan dapat disajikan secara visual dan mudah dipahami
3	5-6	Membuat Sentiment Analisis Pada Aplikasi POEMS ID	Proses analisis sentimen dilakukan dengan tujuan untuk mengelompokkan sentimen nasabah ke dalam beberapa kategori, sehingga dapat diketahui kecenderungan persepsi nasabah terhadap layanan atau produk yang dianalisis.
4	7	Membuat Dashboard Hasil Sentiment Analisis pada Aplikasi Poems ID	Menyusun dashboard yang bersumber dari hasil analisis sentimen dengan tujuan menyajikan informasi secara visual agar lebih mudah dipahami dan mendukung proses analisis
5	8	Membuat Website Foreign Transaction	Mengembangkan fitur baru untuk menampilkan data transaksi luar negeri (foreign transaction)
6	9-10	Membuat Website Phillip Coins Expired	Melakukan perubahan UI pada history coins
7	11	Membuat Website Update Account (UA) Margin	Mengembangkan fitur baru untuk pengkiniaan data UA Margin
8	12	Membuat Website Open Account (OA) Margin	Melakukan pembaruan antarmuka pengguna (User Interface) pada OA Margin serta menambahkan field baru guna meningkatkan fungsionalitas dan kemudahan penggunaan

3.3 Uraian Pelaksanaan Kerja

Selama menjalani kegiatan magang, peserta magang diberikan berbagai tanggung jawab yang disampaikan melalui sistem backlog perusahaan. Setiap tugas yang diterima berkaitan dengan proses pengembangan aplikasi berbasis web, terutama pada pengembangan *Single Page Application (SPA)* yang merupakan pendekatan modern dalam membangun antarmuka web yang dinamis, responsif, dan efisien. Dalam hal ini, peserta magang diharapkan memahami konsep dasar arsitektur front-end, struktur komponen, serta mekanisme integrasi dengan layanan *Application Programming Interface (API)* yang digunakan perusahaan.

Selain bekerja pada bagian pengembangan aplikasi, peserta magang juga memiliki peran penting dalam pekerjaan yang melibatkan *machine learning (ML)* dan analisis data. Tugas-tugas tersebut mencakup proses pengumpulan data, pembersihan data, eksplorasi data, hingga pembuatan model seperti *clustering* untuk mengelompokkan data nasabah berdasarkan pola tertentu, serta *sentiment analysis* untuk memetakan opini pengguna terhadap layanan perusahaan. Proses ini membutuhkan pemahaman terhadap berbagai algoritma dan teknik statistik agar hasil analisis dapat memberikan informasi yang akurat dan relevan.

Dalam pelaksanaan tugasnya, peserta magang mengikuti alur kerja (*workflow*) yang ditetapkan perusahaan, dimulai dari analisis kebutuhan yang diajukan oleh *user*, perencanaan pengerjaan, pengembangan fitur, proses *testing*, hingga penyusunan dokumentasi teknis. Setiap progres pekerjaan dilaporkan secara berkala melalui *meeting* rutin agar dapat dipantau oleh supervisor dan tim terkait. Peserta magang juga diharapkan mampu bekerja secara kolaboratif, berdiskusi, serta memberikan solusi ketika menghadapi kendala teknis

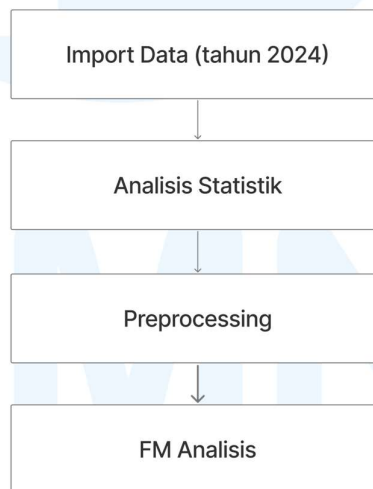
3.3.1 Proses Pelaksanaan

3.3.1.1. Membuat Clustering Data Nasabah

Pada proyek ini dilakukan proses pembentukan klasifikasi nasabah sebagai dasar dalam melakukan analisis nasabah secara lebih terstruktur dan mendalam. Klasifikasi nasabah ini bertujuan untuk mengelompokkan nasabah ke dalam beberapa kategori tertentu berdasarkan karakteristik,

perilaku, serta pola transaksi yang relevan. Dengan adanya pengelompokan ini, perusahaan dapat lebih mudah memahami kebutuhan, preferensi, dan tingkat keterlibatan masing-masing kelompok nasabah, sehingga analisis yang dilakukan menjadi lebih terarah dan informatif.

Untuk mendukung proses tersebut, proyek ini juga membangun sebuah pipeline analisis data yang terstruktur dan terintegrasi. Pipeline ini dirancang untuk mengotomatisasi seluruh tahapan pengolahan data nasabah, mulai dari pengumpulan data, pembersihan dan transformasi data, hingga proses analisis dan penyajian hasil. Dengan adanya pipeline, setiap tahapan analisis dapat dijalankan secara konsisten dan berulang, serta meminimalkan kesalahan yang dapat terjadi akibat proses manual. Pipeline ini juga memudahkan pengembangan dan pemeliharaan sistem analisis nasabah di masa mendatang.



Gambar 3. 5 Pipeline FM Clustering

Berdasarkan Gambar 3.5, dapat dilihat sebuah *pipeline* atau alur kerja yang digunakan dalam proyek *clustering*. Pipeline ini menggambarkan tahapan-tahapan yang saling berkaitan dan dilakukan secara berurutan untuk memastikan bahwa data yang dianalisis memiliki kualitas yang baik serta menghasilkan output analisis yang akurat dan relevan. Setiap tahapan

dalam pipeline memiliki peran penting dalam mendukung keberhasilan proses *clustering* yang dilakukan.

Tahap pertama dalam pipeline ini adalah import data. Pada tahap ini, data dikumpulkan dari sumber utama, yaitu sebuah basis data (*database*) yang menyimpan informasi mentah yang dibutuhkan dalam penelitian. Data tersebut kemudian diekspor ke dalam format file Excel. Penggunaan format Excel bertujuan untuk memudahkan proses pembacaan, pemeriksaan awal, serta pengolahan data menggunakan berbagai perangkat lunak analisis. Pada tahap ini juga dilakukan pengecekan awal terhadap struktur data, seperti jumlah atribut, jenis data, dan kelengkapan data, untuk memastikan bahwa data dapat diproses lebih lanjut.

Setelah proses import data selesai, pipeline berlanjut ke tahap analisis statistik. Tahap ini bertujuan untuk memahami karakteristik dasar dari data yang digunakan. Analisis statistik dilakukan dengan menghitung nilai-nilai statistik deskriptif seperti nilai minimum, maksimum, rata-rata (*mean*), median, standar deviasi, serta distribusi data pada setiap atribut. Melalui analisis ini, peneliti dapat memperoleh gambaran umum mengenai pola data, mendeteksi adanya anomali atau *outlier*, serta memahami kecenderungan data yang dapat memengaruhi hasil *clustering*. Tahap ini sangat penting sebagai dasar pengambilan keputusan pada proses selanjutnya.

Tahap berikutnya adalah proses *preprocessing* data, yang merupakan salah satu tahapan paling krusial dalam keseluruhan pipeline pengolahan data. Pada tahap ini, data yang telah melalui analisis statistik awal akan dibersihkan, diseleksi, dan disiapkan agar sesuai dengan kebutuhan metode *clustering* yang akan digunakan. Proses *preprocessing* bertujuan untuk meningkatkan kualitas data sehingga hasil pengelompokan yang dihasilkan menjadi lebih akurat dan relevan. Adapun tahapan *preprocessing* yang dilakukan meliputi penanganan data dalam jumlah besar dengan menerapkan transformasi logaritmik (*log transformation*) untuk

mengurangi skala data yang terlalu besar dan menstabilkan varians data. Penerapan proses tersebut ditunjukkan pada gambar di bawah ini.



Gambar 3. 6 Implementasi *Log Transform*

Tahap terakhir dalam pipeline ini adalah FM Analysis. Pada tahap ini, data yang telah melalui proses preprocessing digunakan untuk melakukan analisis lebih lanjut berdasarkan konsep FM (Frequency dan Monetary). FM Analysis bertujuan untuk mengelompokkan data berdasarkan frekuensi dan nilai moneter tertentu, sehingga dapat mengidentifikasi pola-pola yang tersembunyi dalam data. Hasil dari FM Analysis ini kemudian digunakan sebagai dasar dalam proses *clustering* untuk membentuk kelompok-kelompok data yang memiliki karakteristik serupa. Dengan demikian, tahapan ini berperan penting dalam menghasilkan insight yang bermanfaat dan sesuai dengan tujuan penelitian yang telah ditetapkan.

3.3.1.2. Membuat Dashboard Hasil Clustering Data Nasabah

Setelah proses analisis FM (*Frequency dan Monetary*) selesai dilakukan, tahap selanjutnya dalam proyek ini adalah pengembangan sebuah dashboard visualisasi. Dashboard ini dirancang untuk menyajikan gambaran yang komprehensif dan mudah dipahami mengenai hasil clustering data nasabah. Melalui dashboard tersebut, pengguna dapat mengamati pola pengelompokan nasabah berdasarkan karakteristik perilaku transaksi, sehingga memudahkan dalam mengidentifikasi segmen nasabah yang memiliki nilai, frekuensi, dan kontribusi yang berbeda. Selain itu, dashboard ini juga berfungsi sebagai alat bantu analisis dan pengambilan keputusan, karena informasi yang disajikan dapat digunakan oleh pihak terkait untuk merancang strategi pemasaran, peningkatan layanan, serta

pengelolaan hubungan dengan nasabah secara lebih efektif dan tepat sasaran.

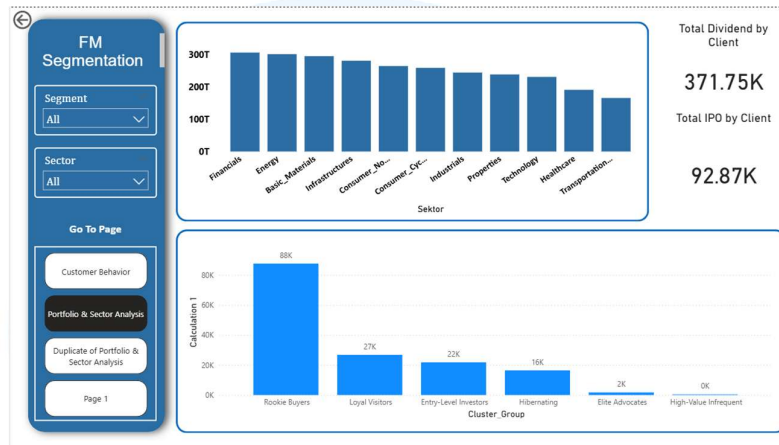
Pembuatan *dashboard* ini dilakukan dengan menggunakan *Microsoft Power BI* sebagai alat utama dalam proses visualisasi data. *Power BI* dipilih karena kemampuannya dalam mengolah data secara interaktif, menampilkan informasi dalam bentuk visual yang informatif, serta memudahkan pengguna dalam melakukan eksplorasi data. Dengan memanfaatkan berbagai fitur yang tersedia, seperti grafik interaktif, *filter*, dan *drill-down*, *dashboard* yang dihasilkan mampu menyajikan hasil analisis secara lebih jelas, terstruktur, dan mudah dipahami. Selain itu, penggunaan *Power BI* juga mendukung proses pembaruan data secara efisien, sehingga *dashboard* dapat digunakan sebagai alat bantu analisis dan pengambilan keputusan yang efektif dan berkelanjutan.



Gambar 3. 7 Dashboard Customer Behavior (*clustering*)

Pembuatan dashboard ini dirancang dengan dua tampilan utama yang memiliki fokus analisis yang berbeda. Tampilan pertama adalah Customer Behavior, yang menyajikan visualisasi mengenai perilaku nasabah berdasarkan data transaksi dan hasil pengelompokan (*clustering*) yang telah dilakukan. Melalui tampilan ini, pengguna dapat memahami pola aktivitas

nasabah, seperti frekuensi transaksi dan nilai kontribusi, sehingga memberikan gambaran yang lebih mendalam mengenai karakteristik masing-masing segmen nasabah.



Gambar 3. 8 Contoh Dashboard Portofolio & Sector (data dummy)

Tampilan kedua adalah Portfolio & Sector, yang berfungsi untuk menampilkan distribusi portofolio nasabah berdasarkan sektor usaha atau kategori tertentu. Pada tampilan ini, pengguna dapat melihat komposisi dan kontribusi setiap sektor terhadap keseluruhan portofolio, sehingga membantu dalam menganalisis potensi, risiko, serta peluang pengembangan bisnis pada masing-masing sektor. Kedua tampilan dashboard tersebut saling melengkapi dan dirancang untuk mendukung proses analisis data serta pengambilan keputusan secara lebih terstruktur dan informatif.

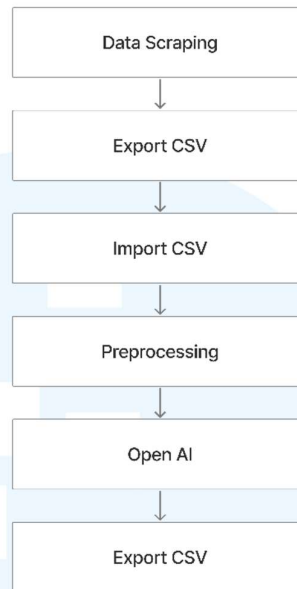
3.3.1.3. Membuat Sentiment Analisis Pada Aplikasi POEMS ID

Pada proyek ini, dilakukan proses analisis sentimen (*sentiment analysis*) terhadap aplikasi POEMS ID yang tersedia pada platform Android dan iOS. Analisis sentimen ini bertujuan untuk mengidentifikasi serta memahami tingkat kepuasan pengguna (*customer satisfaction*) terhadap layanan investasi yang disediakan oleh aplikasi POEMS ID berdasarkan ulasan (*review*) yang diberikan oleh pengguna pada masing-masing platform.

Data ulasan dikumpulkan dari Google Play Store untuk pengguna Android dan App Store untuk pengguna iOS. Ulasan tersebut kemudian diproses melalui beberapa tahapan, mulai dari pengumpulan data (*data collection*), pembersihan teks (*text preprocessing*), hingga analisis sentimen menggunakan pendekatan pemrosesan bahasa alami (*Natural Language Processing*). Setiap ulasan dianalisis untuk menentukan kategori sentimen, yaitu positif, negatif, atau netral, yang mencerminkan persepsi pengguna terhadap kualitas layanan aplikasi.

Melalui analisis sentimen ini, diharapkan dapat diperoleh gambaran yang lebih komprehensif mengenai pengalaman pengguna (*user experience*) dalam menggunakan aplikasi POEMS ID, termasuk kelebihan, kekurangan, serta aspek-aspek layanan yang sering menjadi perhatian pengguna. Hasil dari analisis ini dapat digunakan sebagai bahan evaluasi bagi pengembang aplikasi dalam meningkatkan kualitas layanan, memperbaiki fitur yang bermasalah, serta mempertahankan aspek positif yang telah diapresiasi oleh pengguna.

Pada proyek ini dilakukan perancangan dan pembangunan sebuah pipeline menggunakan python untuk proses analisis sentimen (*sentiment analysis*) yang terstruktur dan sistematis. Pipeline ini dirancang untuk mengintegrasikan seluruh tahapan analisis sentimen secara berurutan, mulai dari proses pengumpulan data ulasan pengguna, pembersihan dan prapemrosesan teks (*text preprocessing*), hingga tahap analisis sentimen dan penyajian hasil. Dengan adanya pipeline ini, setiap proses dapat dijalankan secara otomatis, konsisten, dan mudah direplikasi, sehingga meningkatkan efisiensi serta akurasi dalam melakukan analisis sentimen terhadap data ulasan aplikasi.



Gambar 3. 9 Pipeline Sentiment Analysis

Pipeline analisis sentimen pada aplikasi POEMS ID dimulai dengan tahap pengambilan data (*data scraping*), yaitu proses mengumpulkan ulasan pengguna dari Google Play Store untuk platform Android dan App Store untuk platform iOS. Data ulasan yang diperoleh pada tahap ini mencakup informasi seperti isi ulasan, rating, tanggal, dan sumber platform. Setelah proses pengambilan data selesai, data mentah tersebut disimpan dalam format CSV (*Comma-Separated Values*) melalui tahap *export CSV*, sehingga data dapat terdokumentasi dengan baik dan digunakan kembali pada tahap selanjutnya.

Tahap berikutnya adalah proses import CSV, yaitu tahapan di mana data ulasan yang telah disimpan sebelumnya dimuat kembali ke dalam sistem untuk selanjutnya dilakukan pemrosesan lebih lanjut. Setelah data berhasil dimuat, pipeline pemrosesan dilanjutkan ke tahap pra-pemrosesan (*preprocessing*). Tahap ini bertujuan untuk membersihkan serta menormalkan teks ulasan agar data siap digunakan pada proses analisis berikutnya. Proses pra-pemrosesan yang dilakukan meliputi penyaringan pola teks menggunakan regular expression (*regex*) untuk menghilangkan

karakter yang tidak diperlukan, penghapusan kata-kata umum (stopword removal) menggunakan daftar stopwords Bahasa Indonesia, proses word tokenization untuk memisahkan teks menjadi unit kata, serta proses stemming menggunakan stemmer factory guna mengubah kata ke bentuk dasarnya. Alur proses pra-pemrosesan tersebut ditunjukkan pada gambar di bawah ini.



```
1 import re
2 import emoji
3 from nltk.tokenize import word_tokenize
4 from nltk.corpus import stopwords
5 from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
6
7 pattern = r"[\'!?\*,<>/()]+\"
8
9 stopwords_list = set(stopwords.words(\"indonesian\"))
10 stopwords_list.update([
11     \"tp\", \"utk\", \"yg\", \"aja\", \"klo\", \"trs\", \"gk\", \"ga\", \"gak\", \"bgt\"
12 ])
13
14 stemmer = StemmerFactory().create_stemmer()
15
16 def clean_text(text):
17     text = text.lower()
18     text = re.sub(pattern, \"\", text)
19     text = emoji.demojize(text, delimiters=(\" \", \" \"))
20     tokens = word_tokenize(text)
21     tokens = [t for t in tokens if t not in stopwords_list]
22     tokens = [stemmer.stem(t) for t in tokens]
23     return \" \".join(tokens)
```

Gambar 3. 10 *Preprocessing* data

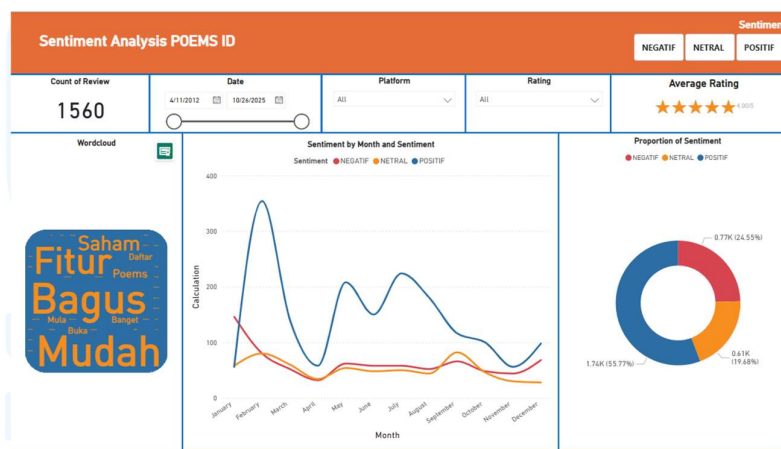
Pada tahap ini dilakukan berbagai proses seperti pengubahan huruf menjadi huruf kecil (*case folding*), penghapusan tanda baca dan simbol, tokenisasi, penghapusan kata umum (*stopwords*), serta stemming untuk memperoleh bentuk kata dasar. Tahap preprocessing ini sangat penting untuk meningkatkan kualitas data teks sebelum dilakukan analisis sentimen.

Selanjutnya, data teks yang telah diproses dikirim ke OpenAI API pada tahap *Open AI* untuk dilakukan analisis sentimen. Pada tahap ini, setiap ulasan dianalisis menggunakan model bahasa berbasis *Artificial*

Intelligence untuk menentukan kategori sentimen, yaitu positif, negatif, atau netral, serta untuk memahami konteks dan emosi yang terkandung dalam ulasan pengguna. Hasil analisis sentimen yang diperoleh kemudian disimpan kembali dalam bentuk CSV melalui tahap *export CSV* sebagai output akhir dari pipeline. File hasil ini berisi informasi sentimen yang dapat digunakan untuk analisis lebih lanjut, visualisasi data, maupun sebagai bahan evaluasi tingkat kepuasan pengguna terhadap layanan investasi yang disediakan oleh aplikasi POEMS ID

3.3.1.4. Membuat Dashboard Hasil Sentiment Analisis pada Aplikasi Poems ID

Pada proyek ini merupakan kelanjutan dari proyek analisis sentimen (*sentiment analysis*) yang telah dilakukan sebelumnya. Setelah proses analisis sentimen terhadap ulasan pengguna berhasil diselesaikan, tahap selanjutnya pada proyek ini adalah melakukan pengolahan dan penyajian hasil analisis tersebut dalam bentuk dashboard visual. Dashboard ini dibangun menggunakan Power BI dengan tujuan untuk menyajikan hasil analisis sentimen secara informatif, interaktif, dan mudah dipahami oleh pengguna.



Gambar 3. 11 Contoh *Dashboard Sentiment Analysis (data dummy)*

Gambar 3.11 menunjukkan dashboard analisis sentimen (*sentiment analysis dashboard*) yang dirancang untuk menyajikan hasil analisis

sentimen terhadap ulasan pengguna aplikasi secara visual dan terstruktur. Dashboard ini menampilkan berbagai informasi penting, di antaranya distribusi sentimen positif, negatif, dan netral, yang memberikan gambaran umum mengenai persepsi pengguna terhadap aplikasi. Selain itu, dashboard juga menyajikan tren sentimen berdasarkan periode waktu tertentu, sehingga memungkinkan analisis perubahan tingkat kepuasan pengguna dari waktu ke waktu.

Selain distribusi dan tren sentimen, dashboard ini juga menampilkan aspek-aspek utama yang sering dibahas oleh pengguna dalam ulasan aplikasi, seperti fitur, performa, kemudahan penggunaan, serta kualitas layanan. Seluruh informasi tersebut disajikan dalam bentuk berbagai elemen visual, seperti grafik batang, diagram lingkaran, dan tabel ringkasan, dengan memanfaatkan fitur visualisasi yang tersedia pada Power BI. Penyajian data dalam bentuk visual ini memudahkan pengguna untuk melakukan eksplorasi data (*data exploration*) secara lebih mendalam dan interaktif.

Melalui dashboard analisis sentimen ini, diharapkan pihak terkait dapat memperoleh gambaran yang lebih komprehensif mengenai tingkat kepuasan pengguna terhadap aplikasi, sekaligus mengidentifikasi area layanan yang perlu dipertahankan maupun ditingkatkan. Informasi yang dihasilkan dari dashboard ini dapat digunakan sebagai dasar dalam pengambilan keputusan strategis, khususnya dalam upaya peningkatan kualitas layanan dan pengembangan fitur aplikasi di masa mendatang.

3.3.1.5. Membuat Website Foreign Transaction

Pada pekerjaan ini, penulis bertanggung jawab dalam mengembangkan website Foreign Transaction yang berfungsi untuk menampilkan informasi transaksi pihak asing pada setiap kode saham yang tersedia di dalam sistem. Website ini dirancang menggunakan konsep one-page display, sehingga seluruh data dapat diakses dalam satu halaman tanpa perlu melakukan perpindahan menu. Dalam proses pengembangannya,

penulis membuat antarmuka pengguna pada sisi client (frontend) yang menampilkan tabel data transaksi secara terstruktur, informatif, dan responsif. Penulis juga memastikan tampilan dapat berjalan baik di berbagai ukuran layar serta menyediakan elemen interaktif seperti fitur pencarian dan pengurutan data agar informasi yang disajikan mudah dipahami oleh pengguna.

Selain mengembangkan sisi frontend, penulis juga mengerjakan bagian backend yang berfungsi mengambil data transaksi asing dari database perusahaan agar dapat diolah dan dikirimkan ke client melalui Application Programming Interface (API). Pada bagian ini, penulis membuat logika pemanggilan data, mengatur format respon yang dikirimkan ke tampilan, serta melakukan testing untuk memastikan data yang muncul pada halaman sesuai dengan data aktual dalam sistem. Dengan adanya pengembangan kedua sisi ini, website Foreign Transaction dapat berfungsi secara optimal dan memberikan pengalaman akses data yang cepat, akurat, dan efisien bagi pengguna.

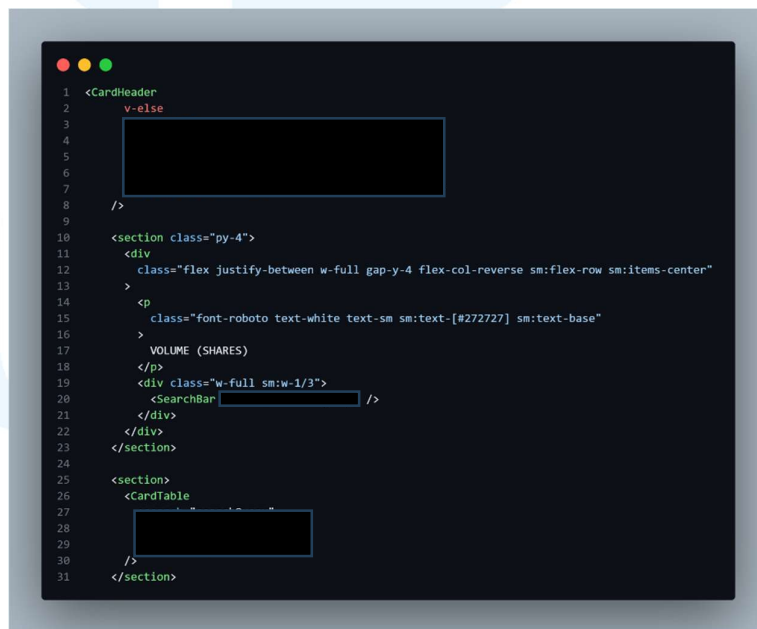
1. Frontend

Pada sisi *frontend*, proyek ini menggunakan *framework* sumber terbuka (*open-source*) yaitu *Vue*. Pemilihan *Vue* dilakukan karena *framework* ini dikenal memiliki proses *bundling* yang sangat cepat, ukuran berkas yang ringan, serta kemudahan dalam pengelolaan komponen. Selain itu, *Vue* menawarkan struktur pengembangan yang sederhana namun fleksibel, sehingga memudahkan penulis dalam membangun antarmuka yang responsif, interaktif, dan mudah dikembangkan lebih lanjut sesuai kebutuhan proyek.

Langkah pertama yang dilakukan dalam pembuatan proyek ini adalah melakukan instalasi *framework* *Vue* sebagai fondasi utama untuk membangun struktur aplikasi. Setelah *Vue* berhasil dipasang, tahap berikutnya adalah menginstal *Tailwind CSS* yang digunakan

sebagai alat untuk melakukan styling. Tailwind dipilih karena mengusung konsep utility-based, sehingga pengembang dapat melakukan penataan tampilan tanpa perlu menulis berkas CSS secara manual. Pendekatan ini membuat proses pengembangan tampilan menjadi lebih cepat, konsisten, dan mudah untuk disesuaikan dengan kebutuhan desain proyek.

Dalam sisi *frontend* dibagi menjadi tiga komponen yaitu *header*, *search bar*, dan *tables*. Ketiga komponen tersebut merupakan memiliki fungsi yang berbeda beda, *header* berfungsi untuk menampilkan data *summary* dari *foreign buy*, *foreign sell*, dan juga *net*. *Search bar* merupakan sebuah komponen yang berfungsi untuk melakukan filter berdasarkan dari *code* saham. Lalu untuk *tables* merupakan sebuah komponen untuk menampilkan data *foreign buy*, *foreign sell*, dan juga *net* berdasarkan dari saham.



```
1 <CardHeader
2   v-else
3   <div>
4     <div>
5       <div>
6         <div>
7           <div>
8             <div>
9               <div>
10                <div>
11                  <div>
12                    <div>
13                      <div>
14                        <div>
15                          <div>
16                            <div>
17                              <div>
18                                <div>
19                                  <div>
20                                    <div>
21                                      <div>
22                                        <div>
23                                          <div>
24                                            <div>
25                                              <div>
26                                                <div>
27                                                  <div>
28                                                    <div>
29                                                      <div>
30                                                        <div>
31                                                          <div>
```

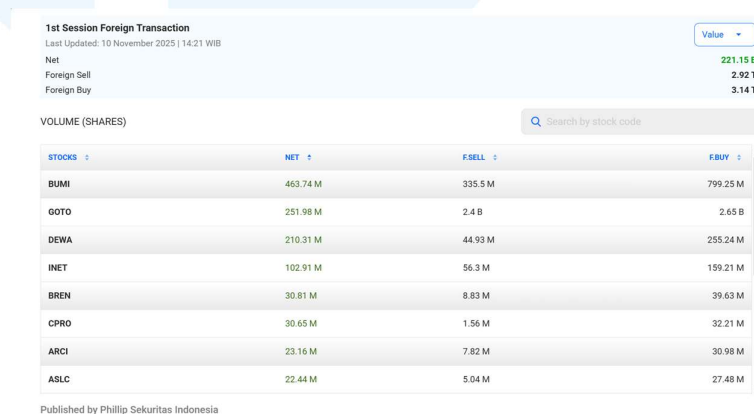
Gambar 3. 12 Cuplikan Code main.vue

Gambar 3.12 menunjukkan cuplikan *code* yang menggambarkan proses penggabungan tiga *component* ke dalam satu berkas.

Penggunaan *component* memiliki keunggulan utama, yaitu memisahkan setiap fungsi ke dalam bagian-bagian yang lebih kecil dan terstruktur sehingga alur kerja aplikasi menjadi lebih mudah dipahami, dikelola, serta dibaca oleh pengembang.

Selain itu, pada berkas yang sama penulis juga membuat beberapa fungsi untuk melakukan pemanggilan data dari *API*, yaitu *fetchStocks* dan *fetchSummary*. Masing-masing fungsi tersebut memiliki peran tersendiri dalam mengambil data yang diperlukan dari *API* yang telah disiapkan sebelumnya, sehingga proses pengolahan dan penampilan data pada halaman dapat berjalan dengan lebih terstruktur dan efisien.

Dengan demikian, sistem mampu menampilkan data pada sisi klien secara optimal, baik pada perangkat desktop maupun perangkat mobile, sebagaimana ditunjukkan pada Gambar 3.13 dan 3.14



1st Session Foreign Transaction
Last Updated: 10 November 2025 | 14:21 WIB

Value ▾

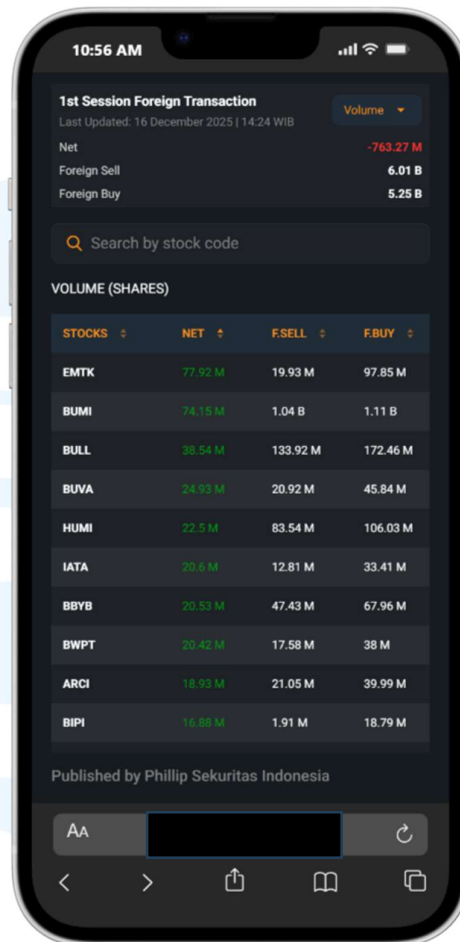
Net 221.15 B
Foreign Sell 2.92 T
Foreign Buy 3.14 T

VOLUME (SHARES) 🔍 Search by stock code

STOCKS ▾	NET ▴	F.SELL ▴	F.BUY ▴
BUMI	463.74 M	335.5 M	799.25 M
GOTO	251.98 M	2.4 B	2.65 B
DEWA	210.31 M	44.93 M	255.24 M
INET	102.91 M	56.3 M	159.21 M
BREN	30.81 M	8.83 M	39.63 M
CPRO	30.65 M	1.56 M	32.21 M
ARCI	23.16 M	7.82 M	30.98 M
ASLC	22.44 M	5.04 M	27.48 M

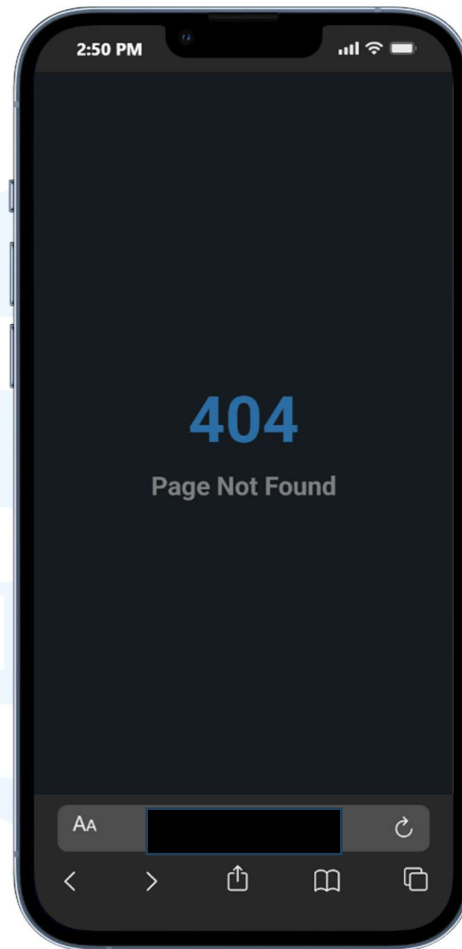
Published by Phillip Sekuritas Indonesia

Gambar 3. 13 Tampilan Client Fitur Foreign Transaction (desktop)



Gambar 3. 14 Tampilan Client Fitur Foreign Transaction (Mobile)

Tidak hanya itu, pembuatan *routes* khusus untuk menangani *invalid link* juga sangat penting guna memberikan penanganan yang tepat ketika pengguna mengakses tautan yang tidak valid, sehingga sistem dapat menampilkan informasi yang jelas, meningkatkan pengalaman pengguna, serta mencegah terjadinya kesalahan atau kebingungan dalam penggunaan aplikasi.



Gambar 3. 15 Tampilan Client Fitur Foreign Transaction Invalid link

2. Backend

Backend merupakan bagian inti dari sistem yang berfungsi sebagai penghubung antara *client* dan database. Pada lapisan ini, backend bertanggung jawab untuk menerima setiap *request* yang dikirimkan oleh *client*, kemudian memproses permintaan tersebut berdasarkan logika bisnis yang telah dirancang. Selanjutnya, backend melakukan komunikasi dengan database untuk menjalankan berbagai operasi data, seperti mengambil, menyimpan, memperbarui, maupun menghapus data sesuai kebutuhan sistem. Hasil dari pemrosesan tersebut kemudian dikirimkan kembali kepada *client* dalam bentuk *response*,

sehingga alur pertukaran data dapat berjalan secara aman, terstruktur, dan efisien.

Pada sisi backend, proyek ini dikembangkan menggunakan teknologi berbasis *JavaScript* dengan memanfaatkan *Node.js* sebagai *runtime environment*. Penggunaan *Node.js* memungkinkan aplikasi berjalan dengan performa yang optimal karena mendukung pemrosesan *asynchronous*, sehingga mampu menangani banyak permintaan secara bersamaan tanpa mengorbankan kecepatan. Selain itu, teknologi ini sangat sesuai untuk mengelola komunikasi antara server dan database, pengembangan serta pengelolaan *API*, dan pemrosesan data secara *real-time*, sehingga backend dapat mendukung kebutuhan sistem secara menyeluruh dan berkelanjutan.

Dalam proses pembuatan backend menggunakan *Node.js*, langkah pertama yang dilakukan adalah melakukan instalasi dan inisialisasi proyek dengan menjalankan perintah *npm i -y*. Perintah ini berfungsi untuk membuat dan menyiapkan berkas *package.json*, yang digunakan sebagai konfigurasi utama proyek untuk menyimpan informasi aplikasi, dependensi, serta pengaturan lainnya. Setelah berkas *package.json* berhasil dibuat, tahap selanjutnya adalah melakukan inisialisasi struktur proyek dan menambahkan konfigurasi yang diperlukan agar aplikasi dapat berjalan dengan baik. Pada tahap ini, dilakukan penulisan kode untuk membuat dan menjalankan *server* menggunakan *Node.js*, sehingga backend siap menerima dan memproses *request* dari *client*. Proses inisialisasi dan pembuatan *server* tersebut ditunjukkan pada gambar di bawah ini, yang menggambarkan tahapan awal dalam pengembangan backend.



Gambar 3. 16 Pembuatan Server proyek *Foreign Transaction*

Setelah proses pembuatan *server* selesai dilakukan, tahap selanjutnya yang tidak kalah penting adalah pembuatan *routes*. *Routes* berfungsi sebagai pengatur jalur komunikasi antara *client* dan *server*, di mana setiap *endpoint* yang dibuat akan menentukan bagaimana *request* dari *client* diproses dan bagaimana *response* dikirimkan kembali. Dengan adanya *routes* yang terstruktur dengan baik, backend dapat mengelola berbagai fitur aplikasi secara terpisah dan terorganisasi, seperti pengelolaan data pengguna, autentikasi, maupun operasi lainnya. Oleh karena itu, pembuatan *routes* menjadi komponen penting dalam memastikan alur kerja backend berjalan secara sistematis, mudah dikembangkan, dan mudah dipelihara di masa mendatang

3.3.1.6. Membuat Website Phillip Coins Expired

Pada proyek ini, penulis melakukan berbagai pengembangan dan peningkatan (*improvement*) baik pada sisi antarmuka pengguna (*UI*) maupun pada sisi *server* untuk mengimplementasikan fitur *expired coins*. Peningkatan pada sisi *UI* bertujuan untuk memberikan tampilan yang lebih menarik, responsif, dan mudah digunakan oleh pengguna, sehingga pengalaman penggunaan aplikasi menjadi lebih optimal. Sementara itu,

pada sisi *server*, dilakukan penyesuaian dan pengembangan logika sistem agar proses pengelolaan masa berlaku (*expired*) koin dapat berjalan secara otomatis dan sesuai dengan aturan yang telah ditentukan.

Dalam pengembangan antarmuka pengguna, proyek ini menggunakan teknologi *Vue.js* sebagai *framework* utama untuk membangun tampilan website yang dinamis dan interaktif. Selain itu, *Bootstrap* digunakan sebagai *CSS framework* untuk membantu dalam pengaturan tata letak, desain, serta memastikan tampilan website bersifat responsif dan konsisten di berbagai ukuran layar. Dengan kombinasi teknologi tersebut, website Phillip Coins Expired diharapkan mampu memberikan performa yang baik serta tampilan yang profesional dalam mendukung kebutuhan sistem secara keseluruhan.

1. Frontend

Pada sisi *frontend*, bagian ini merupakan komponen yang ditampilkan secara langsung kepada *client* atau pengguna. *Frontend* berperan dalam menyajikan antarmuka pengguna serta mengatur alur akses tampilan aplikasi, di mana terdapat mekanisme *guards* yang digunakan untuk membatasi akses ke halaman tertentu. Dengan adanya *guards*, pengguna diwajibkan untuk melakukan *login* terlebih dahulu sebelum dapat mengakses fitur atau halaman tertentu, serta memastikan bahwa parameter-parameter yang dibutuhkan terkait proyek ini telah dikirimkan dan tervalidasi dengan baik. Mekanisme ini bertujuan untuk menjaga keamanan, konsistensi data, dan kelancaran alur penggunaan aplikasi.

Selain itu, penerapan *component* dalam pengembangan aplikasi merupakan hal yang sangat penting dan tidak dapat diabaikan, karena berperan besar dalam menjaga kualitas dan keteraturan *code*. Dengan membagi aplikasi ke dalam beberapa *component* yang memiliki tanggung jawab masing-masing, struktur *code* menjadi lebih modular,

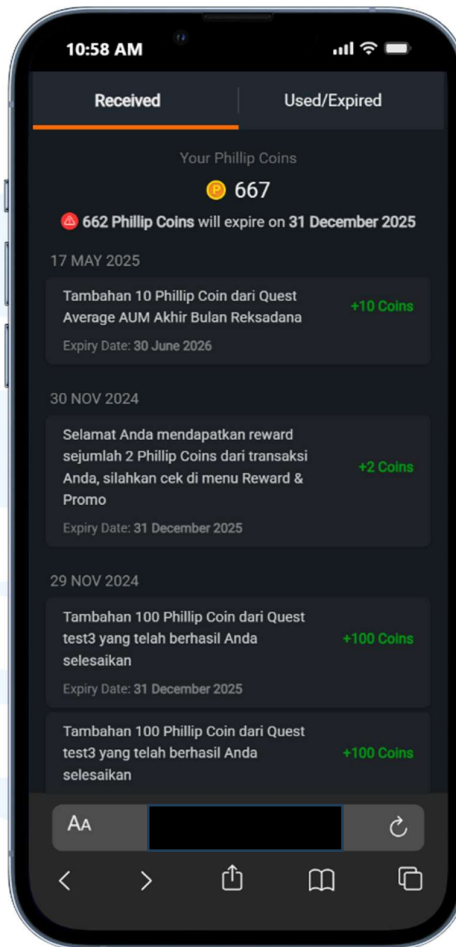
rapi, dan mudah dipahami. Pendekatan ini memudahkan pengembang dalam melakukan *maintenance*, perbaikan *bug*, serta penambahan fitur baru tanpa harus mengubah keseluruhan *codebase*.

Lebih lanjut, penggunaan *component* juga meningkatkan tingkat *reusability*, di mana satu *component* dapat digunakan kembali pada beberapa bagian aplikasi yang berbeda. Hal ini tidak hanya mempercepat proses pengembangan, tetapi juga mengurangi duplikasi *code* dan potensi kesalahan. Dengan demikian, implementasi *component* menjadi salah satu praktik terbaik (*best practice*) dalam pengembangan aplikasi modern untuk memastikan sistem dapat dikembangkan secara berkelanjutan dan mudah dikelola dalam jangka panjang. Berikut gambar dibawah merupakan contoh mengimplementaasi penggunaan component pada vue



Gambar 3. 17 Pembuatan *component* pada Vue

Setelah menyelesaikan proses pengerjaan pada sisi *frontend*, berikut ini merupakan tampilan antarmuka yang dihasilkan pada proyek *Phillip Coins*.



Gambar 3. 18 Tampilan Pada View Mobile

2. Backend

Pada sisi *backend project* ini, bagian ini merupakan komponen sistem yang tidak ditampilkan atau diperlihatkan secara langsung kepada pengguna (*client*). *Backend* berperan sebagai pusat pengolahan data dan logika aplikasi, di mana setiap permintaan (*request*) yang dikirimkan oleh *client* akan diterima, diproses, dan divalidasi terlebih dahulu sebelum diteruskan ke *database*. Selain itu, *backend* juga bertanggung jawab dalam mengelola keamanan, autentikasi, serta memastikan data yang diambil atau disimpan ke dalam *database* berjalan dengan benar dan konsisten.

Proses pengambilan data dilakukan dengan menghubungkan sistem ke dalam *database* serta menerima *stored procedure (SP)* yang telah dibuat sebelumnya. Data yang diperoleh dari *SP* tersebut kemudian diolah menggunakan *library* yang telah tersedia dan sesuai dengan kebutuhan sistem. Setelah proses pengolahan awal selesai, selanjutnya dibuat sebuah fungsi khusus yang bertugas untuk memproses dan mengelola data tersebut, sehingga data dapat diterima, disesuaikan, dan digunakan secara optimal oleh sistem atau komponen lain yang membutuhkan.

Dalam proses pengambilan data, sistem dilengkapi dengan mekanisme pengamanan berupa *guard* yang berfungsi untuk memeriksa *session* dari *user*, guna memastikan apakah pengguna tersebut telah melakukan proses *login* atau belum. Mekanisme ini bertujuan untuk membatasi akses data hanya kepada pengguna yang memiliki hak akses yang sah, sehingga keamanan data tetap terjaga dan mencegah terjadinya akses tidak sah ke dalam sistem. Selain itu, penerapan *guard* ini juga membantu sistem dalam mengelola autentikasi serta menjaga integritas data yang diambil dari *database*. Berikut adalah contoh dari penggunaan *middleware* yang akan di tunjukan pada gambar dibawah.



Gambar 3. 19 Pembuatan *Routes* pada BE

3.3.1.7. Membuat Website Update Account (UA) Margin

Pada proyek ini dilakukan proses *improvement UI* pada fitur *UA Margin* dengan menambahkan beberapa *field* baru yang diperlukan untuk mendukung kebutuhan sistem. Penambahan *field* tersebut bertujuan untuk melengkapi informasi yang dibutuhkan oleh pengguna serta meningkatkan

kemudahan dalam proses pengisian data. Dalam konteks fungsionalitas, fitur *UA Margin* merupakan bagian dari mekanisme pembuatan *request* pengajuan *margin*, sehingga perubahan pada sisi antarmuka pengguna ini diharapkan dapat memperlancar alur proses, meningkatkan akurasi data yang dikirim, serta mengurangi potensi kesalahan dalam pengajuan *margin*.

Sebelum memulai proses development pada proyek ini, dilakukan perubahan pada alur proses kerja dengan menerapkan metode Scrum secara lebih terstruktur, salah satunya dengan menambahkan tahapan sprint dalam perencanaan dan pelaksanaan pekerjaan. Penambahan sprint ini bertujuan untuk membagi pekerjaan ke dalam periode waktu yang lebih terorganisir, sehingga setiap tahap pengembangan dapat dilakukan secara bertahap, terukur, dan mudah untuk dievaluasi. Dengan adanya penerapan sprint pada Scrum, diharapkan proses pengembangan proyek dapat berjalan lebih efektif, meningkatkan kolaborasi antar anggota tim, serta memudahkan dalam melakukan pemantauan progres dan penyesuaian terhadap kebutuhan proyek.

Proses tersebut diawali dengan pelaksanaan *meeting* yang bertujuan untuk membahas proyek *UA Margin* dan *OA Margin*. Dalam *meeting* ini dilakukan pembahasan secara menyeluruh terkait tujuan dari pengubahan atau pengembangan yang akan dilakukan, ruang lingkup pekerjaan, serta penentuan *deadline* pengerjaan proyek. Selain itu, pertemuan ini juga digunakan untuk menyelaraskan pemahaman antar anggota tim, mendiskusikan kebutuhan teknis maupun non-teknis, serta mengidentifikasi potensi kendala yang mungkin muncul selama proses pengerjaan proyek. Dengan adanya *meeting* ini, diharapkan seluruh pihak yang terlibat memiliki pemahaman yang sama sehingga proses pengembangan dapat berjalan secara terarah dan sesuai dengan rencana yang telah ditetapkan.

Setelah proses *meeting* selesai dilaksanakan, pengerjaan proyek dapat dilanjutkan dengan memasuki tahap *development*. Pada tahap ini, setiap anggota tim mulai mengerjakan tugas dan tanggung jawab yang telah

disepakati sebelumnya. Selain itu, untuk memastikan perkembangan pekerjaan tetap terpantau dengan baik, dilakukan kegiatan *daily stand up Scrum* secara rutin sebagai sarana untuk melakukan *update progress*. Melalui *daily stand up Scrum* ini, setiap anggota tim menyampaikan perkembangan pekerjaan, kendala yang dihadapi, serta rencana pekerjaan selanjutnya, sehingga koordinasi antar tim dapat terjaga dan proses pengembangan proyek dapat berjalan secara efektif dan sesuai dengan target yang telah ditentukan.

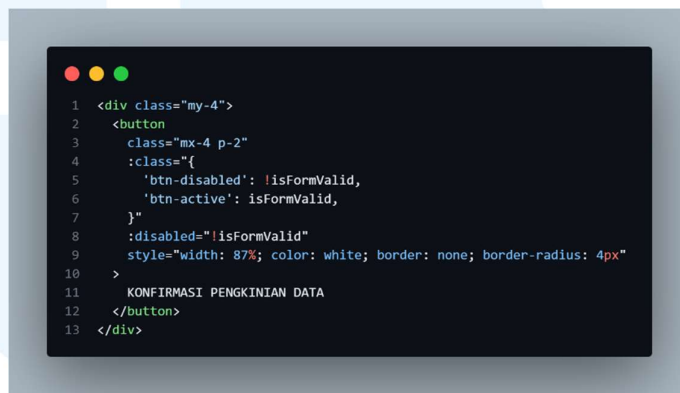
1. Frontend

Pada sisi *frontend*, proyek ini menggunakan teknologi *Vue.js* sebagai *framework* utama dalam pengembangan antarmuka pengguna. Penggunaan *Vue.js* bertujuan untuk membangun tampilan aplikasi yang dinamis, responsif, dan mudah dalam pengelolaan komponen. Selain itu, proyek ini juga memanfaatkan *Bootstrap* sebagai *library CSS* untuk membantu proses perancangan tampilan agar lebih konsisten, responsif, dan sesuai dengan berbagai ukuran layar. Dengan kombinasi *Vue.js* dan *Bootstrap*, diharapkan antarmuka aplikasi dapat memberikan pengalaman pengguna yang lebih baik, mudah digunakan, serta memiliki struktur tampilan yang rapi dan terstandarisasi.

Dalam pengembangannya, proyek ini menerapkan pendekatan *mobile first*, di mana desain dan tata letak antarmuka terlebih dahulu dioptimalkan untuk perangkat mobile sebelum disesuaikan untuk tampilan pada layar yang lebih besar, seperti tablet dan desktop. Pendekatan *mobile first* ini bertujuan untuk memastikan aplikasi dapat digunakan secara optimal pada perangkat mobile, meningkatkan kenyamanan pengguna, serta menjaga konsistensi tampilan dan fungsionalitas pada berbagai jenis perangkat.

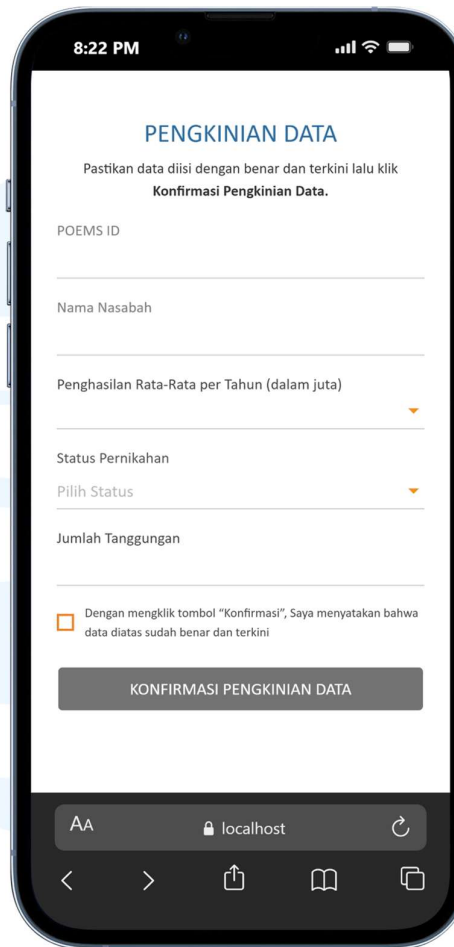
Dari sisi *client*, sistem dilengkapi dengan mekanisme validasi sebagai bentuk penjagaan terhadap data yang diinputkan oleh pengguna. Validasi tersebut mencakup pembatasan penggunaan

karakter (*character*) pada pengisian nama, serta pengaturan format dan jumlah input pada pengisian nomor KTP agar sesuai dengan ketentuan yang berlaku. Selain itu, pengguna diwajibkan untuk mengisi seluruh data yang diperlukan hingga lengkap sebelum proses dapat dilanjutkan atau diproses oleh sistem. Penerapan validasi ini bertujuan untuk meminimalkan kesalahan penginputan, meningkatkan akurasi data, serta memastikan data yang dikirimkan telah memenuhi standar yang ditetapkan.



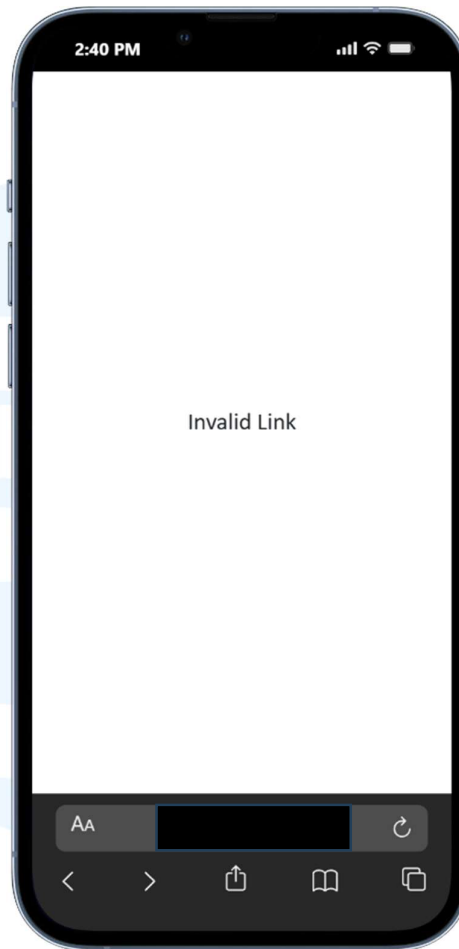
Gambar 3. 20 Implementasi validasi

Setelah mengimplementasikan mekanisme *guard*, berikut ini merupakan tampilan antarmuka dari fitur *UA Margin* yang dihasilkan.



Gambar 3. 21 Tampilan UA Margin

Tidak hanya itu, penerapan sistem penjagaan *routes* juga memiliki peranan yang sangat penting untuk memastikan bahwa setiap akses yang dilakukan oleh pengguna telah melalui proses otorisasi yang sesuai. Dengan adanya mekanisme ini, sistem dapat mencegah pihak yang tidak berwenang mengakses halaman atau fitur tertentu, sehingga keamanan, kestabilan, dan keandalan aplikasi dapat terjaga dengan baik. Berikut merupakan tampilan yang muncul ketika *client* mengakses *route* yang tidak sesuai atau tidak tersedia.



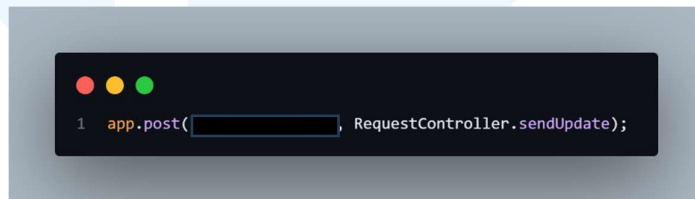
Gambar 3. 22 Routes Validate

2. Backend

Pada sisi *backend*, proyek ini menggunakan *Node.js* sebagai platform utama dalam pengembangan layanan sistem. Penggunaan *Node.js* memungkinkan pengolahan permintaan (*request*) dari *client* dapat dilakukan secara efisien dan bersifat asinkron, sehingga mampu menangani proses komunikasi data dengan lebih cepat dan stabil. Selain itu, *Node.js* juga mendukung penggunaan berbagai *library* dan *framework* yang membantu dalam pengembangan *backend*, pengelolaan *API*, serta integrasi dengan *database*. Dengan pemanfaatan *Node.js* sebagai *backend*, sistem

diharapkan dapat berjalan secara optimal, mudah dikembangkan, dan mampu mendukung kebutuhan aplikasi secara keseluruhan.

Sama seperti proyek sebelumnya, proses pengambilan data pada proyek ini dilakukan dengan memanfaatkan hasil dari pembuatan *stored procedure* (*SP*) yang telah disiapkan sebelumnya. *Stored procedure* tersebut digunakan sebagai mekanisme utama dalam mengambil dan mengolah data dari *database*, sehingga proses akses data dapat berjalan lebih terstruktur, efisien, dan aman. Dengan menggunakan *SP*, logika pengambilan data dapat dipusatkan di sisi *database*, serta memudahkan proses pemeliharaan dan pengembangan sistem apabila di kemudian hari terjadi perubahan kebutuhan atau struktur data.



Gambar 3. 23 Pembuatan *Route* pada BE

3.3.1.8. Membuat Website Open Account (OA) Margin

Pada proyek *OA*, dilakukan proses *improvement UI* pada fitur *Margin* dengan menambahkan beberapa *field* baru yang diperlukan untuk mendukung kebutuhan sistem. Penambahan *field* ini bertujuan untuk melengkapi data yang harus diinput oleh pengguna, meningkatkan kejelasan informasi, serta memastikan proses pengajuan *Margin* dapat berjalan sesuai dengan alur dan ketentuan yang berlaku. Selain itu, pembaruan pada sisi antarmuka ini juga diharapkan dapat meningkatkan kenyamanan pengguna dalam melakukan pengisian data, mengurangi potensi kesalahan input, serta mendukung integrasi data yang lebih baik dengan proses di sisi *backend*.

Proyek *OA Margin* mengikuti alur *sprint planning* yang sama seperti pada proyek *UA Margin*. Dalam tahap *sprint planning* ini, dilakukan

perencanaan pekerjaan secara terstruktur dengan menentukan ruang lingkup pengembangan, pembagian tugas kepada setiap anggota tim, serta penetapan prioritas dan target penyelesaian pada setiap *sprint*. Penerapan alur *sprint planning* ini bertujuan untuk memastikan proses pengembangan proyek berjalan secara sistematis, terukur, dan selaras dengan kebutuhan bisnis serta tenggat waktu yang telah ditetapkan.

Pada proyek *OA Margin*, fitur yang dikembangkan merupakan sebuah mekanisme utama yang digunakan untuk melakukan *request margin*. Fitur ini dirancang untuk memfasilitasi pengguna dalam mengajukan permohonan *margin* secara sistematis dan terstruktur, mulai dari proses pengisian data, pengiriman permintaan, hingga pemrosesan lebih lanjut oleh sistem. Dengan adanya fitur *request margin* ini, diharapkan alur pengajuan dapat berjalan lebih efisien, akurat, serta sesuai dengan kebutuhan operasional dan ketentuan yang telah ditetapkan dalam sistem.

1. Frontend

Pada sisi *frontend*, proyek ini menggunakan teknologi *Vue.js* sebagaimana yang telah diterapkan pada proyek sebelumnya, serta memanfaatkan *Bootstrap* sebagai pendukung dalam perancangan tampilan antarmuka pengguna. Penggunaan kombinasi *Vue.js* dan *Bootstrap* ini bertujuan untuk menghasilkan antarmuka yang dinamis, responsif, dan konsisten, sehingga dapat meningkatkan kenyamanan serta kemudahan pengguna dalam mengakses fitur-fitur yang tersedia.

Pada proses pembuatan *frontend*, perubahan *UI* memiliki tantangan tersendiri yang perlu diperhatikan dengan baik. Beberapa tantangan tersebut antara lain adalah pembuatan *step button* yang berfungsi untuk menampilkan alur proses secara bertahap, sehingga pengguna dapat memahami setiap langkah yang harus dilalui. Selain itu, sistem juga dilengkapi dengan fitur pemilihan bahasa, yaitu

Bahasa Indonesia dan Bahasa Inggris, yang membutuhkan penyesuaian pada teks, tata letak, serta konsistensi tampilan. Tantangan-tantangan ini menuntut perancangan antarmuka yang matang agar tetap mudah digunakan, informatif, serta mampu memberikan pengalaman pengguna yang optimal pada berbagai kebutuhan pengguna.

Sama seperti proyek sebelumnya, fitur *OA Margin* pada proyek ini diimplementasikan dengan fokus utama pada penggunaan perangkat *mobile*. Seluruh perancangan dan pengembangan antarmuka disesuaikan secara khusus agar dapat digunakan secara optimal pada layar *mobile*, baik dari segi tata letak, ukuran komponen, maupun kemudahan navigasi. Pendekatan ini bertujuan untuk memastikan kenyamanan pengguna saat mengakses fitur *OA Margin* melalui perangkat *mobile*, serta mendukung kebutuhan operasional yang lebih fleksibel dan efisien.





Gambar 3. 24 Tampilan OA Margin

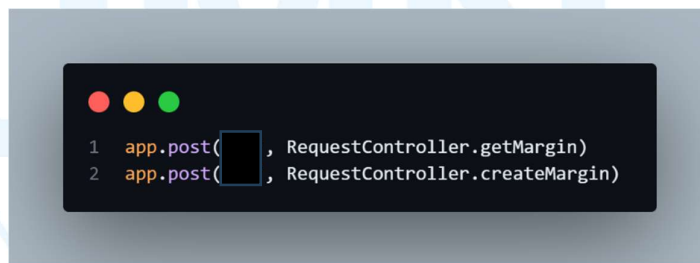
2. Backend

Pada sisi *backend*, proyek ini menggunakan *Node.js* sebagai platform pengembangan, sama seperti yang telah diterapkan pada proyek sebelumnya. Penggunaan *Node.js* memungkinkan sistem untuk menangani proses komunikasi antara *client* dan *server* secara efisien, terutama dalam menerima dan memproses setiap *request* yang masuk. Selain itu, *Node.js* juga mendukung pengembangan layanan *backend* yang bersifat asinkron, sehingga dapat meningkatkan performa aplikasi, memudahkan integrasi dengan *database*, serta mendukung pengelolaan *API* secara lebih terstruktur dan mudah dikembangkan di masa mendatang.

Dalam proses menerima dan mengirimkan data, proyek ini menggunakan *stored procedure (SP)* sebagai mekanisme utama dalam pengolahan data. Penggunaan *SP* memungkinkan proses pengambilan, penyimpanan, serta pengiriman data ke *database* dilakukan secara terstruktur dan efisien. Selain itu, penerapan *stored procedure* juga membantu meningkatkan keamanan data, karena logika pengolahan data dipusatkan pada sisi *database*, serta memudahkan pemeliharaan dan pengembangan sistem apabila di kemudian hari diperlukan perubahan atau penyesuaian kebutuhan.

Berikut ini merupakan contoh tampilan *API* yang digunakan pada proyek *Node.js*. *API* ini berfungsi sebagai penghubung antara sisi *frontend* dan *backend*, di mana setiap permintaan (*request*) dan respons (*response*) data diproses melalui layanan yang telah dirancang sesuai dengan kebutuhan sistem.

Berikut ini merupakan contoh tampilan *API* yang digunakan pada proyek *Node.js*. *API* ini berfungsi sebagai penghubung antara sisi *frontend* dan *backend*, di mana setiap permintaan (*request*) dan respons (*response*) data diproses melalui layanan yang telah dirancang sesuai dengan kebutuhan sistem.



Gambar 3. 25 Pembuatan *Routes* pada BE

3.3.2 Kendala yang Ditemukan

Selama pelaksanaan kegiatan praktik magang, penulis menemukan beberapa kendala yang dialami dalam proses pelaksanaan tugas dan

pekerjaan. Adapun kendala-kendala tersebut akan dijelaskan secara rinci pada bagian berikut.

1. Batasan Koneksi Internet

Salah satu kendala yang dihadapi selama pelaksanaan kegiatan magang adalah keterbatasan koneksi internet, khususnya pada saat mengakses *database* (DB). Koneksi yang tidak stabil maupun terbatas menyebabkan proses pengambilan data berjalan lebih lambat dan kurang efisien. Kondisi tersebut berdampak langsung pada durasi pengerjaan analisis data serta proses pengembangan *backend* untuk kebutuhan *fetching* data. Oleh karena itu, diperlukan penyesuaian jadwal kerja serta penerapan strategi alternatif dalam proses pengolahan data agar seluruh tugas dan pekerjaan tetap dapat diselesaikan secara optimal.

2. Keterbatasan Akses Data

Keterbatasan akses data menjadi salah satu kendala yang cukup signifikan selama pelaksanaan kegiatan magang. Kendala ini menjadi tantangan tersendiri, terutama ketika melakukan proses *development* dan *testing* dalam pembuatan maupun pengembangan suatu fitur. Akses data yang tidak dapat diperoleh secara langsung atau harus melalui prosedur tertentu menyebabkan proses pengujian fitur tidak dapat dilakukan secara maksimal. Hal tersebut berdampak pada efektivitas pengembangan sistem, sehingga diperlukan penyesuaian alur kerja serta koordinasi dengan pihak terkait agar proses *development* dan *testing* tetap dapat berjalan dengan baik.

3. Permasalahan Versi *Environment*

Permasalahan perbedaan versi *environment* atau lingkungan pengembangan juga menjadi kendala dalam pelaksanaan pekerjaan. Perbedaan versi perangkat lunak, pustaka (*library*), atau sistem yang digunakan menyebabkan beberapa program tidak dapat dijalankan secara optimal. Oleh karena itu, diperlukan penyesuaian konfigurasi

serta pembaruan atau penyesuaian versi agar proses pengerjaan dapat berjalan dengan lancar.

3.3.3 Solusi atas Kendala yang Ditemukan

Selama pelaksanaan kegiatan praktik magang, penulis menemukan beberapa kendala yang dihadapi dalam proses pelaksanaan tugas. Namun demikian, kendala-kendala tersebut dapat diatasi melalui langkah-langkah pencegahan dan solusi yang akan dijelaskan secara rinci pada bagian berikut.

1. Solusi terhadap Batasan Koneksi Internet

Untuk mengatasi keterbatasan koneksi internet dalam mengakses *database* (DB), solusi yang diterapkan adalah dengan melakukan penjadwalan ulang proses pengambilan data pada waktu koneksi relatif lebih stabil. Selain itu, data yang telah berhasil diambil disimpan secara lokal atau dalam bentuk *cache* sehingga dapat digunakan kembali tanpa harus melakukan akses berulang ke DB. Pada proses pengembangan *backend*, pengujian juga dilakukan menggunakan data sampel atau *mock data* agar proses *fetching* data tetap dapat diuji meskipun koneksi internet terbatas. Dengan strategi tersebut, proses analisis data dan pengembangan sistem tetap dapat berjalan secara lebih efisien.

2. Solusi terhadap Keterbatasan Akses Data

Dalam menghadapi keterbatasan akses data, langkah yang dilakukan adalah dengan melakukan koordinasi secara intensif dengan pembimbing dan tim terkait untuk memperoleh kejelasan prosedur perizinan akses data. Selama menunggu akses data resmi, proses *development* dan *testing* fitur tetap dilakukan dengan memanfaatkan data dummy atau data simulasi yang memiliki struktur serupa dengan data asli.

Selain itu, diterapkan pula proses ETL (*Extract, Transform, Load*) sebagai bagian dari pengelolaan data. Pada tahap *extract*, data dummy dikumpulkan dari berbagai sumber simulasi yang relevan. Selanjutnya,

pada tahap *transform*, data tersebut dibersihkan, divalidasi, dan disesuaikan format serta strukturnya agar menyerupai karakteristik data asli. Terakhir, pada tahap *load*, data hasil transformasi dimuat ke dalam sistem atau basis data pengujian untuk mendukung proses pengembangan dan pengujian fitur.

Pendekatan ini memungkinkan proses pengembangan sistem tetap berjalan tanpa harus menunggu akses data sepenuhnya, sekaligus memastikan kesiapan alur pengolahan data ketika data resmi telah tersedia, sehingga waktu pengerjaan dapat dimanfaatkan secara lebih optimal dan efektif.

3. Solusi terhadap Permasalahan Versi *Environment*

Untuk mengatasi permasalahan perbedaan versi *environment*, solusi yang diterapkan adalah dengan membangun sebuah *pipeline* pengembangan yang terstandarisasi dan dilengkapi dengan berkas *requirements library*. Seluruh pustaka (*library*) yang digunakan dalam proyek didokumentasikan secara jelas dalam sebuah berkas *requirements* (seperti *requirements.txt*), sehingga setiap lingkungan pengembangan dapat menggunakan versi pustaka yang sama. Selain itu, proses instalasi dan konfigurasi *environment* dilakukan secara terstruktur melalui *pipeline* tersebut, sehingga meminimalkan perbedaan konfigurasi antar sistem. Dengan adanya *pipeline* dan daftar *requirements library* yang jelas, proses pengembangan, pengujian, dan implementasi aplikasi dapat berjalan lebih konsisten, stabil, dan mudah direplikasi di lingkungan lain