

# BAB 1

## PENDAHULUAN

### 1.1 Latar Belakang

Sektor perbankan adalah bagian penting dalam mengembangkan ekonomi modern yang memainkan fungsi yang sangat penting dalam memajukan ekonomi suatu negara, sehingga perbankan perlu secara konsisten melakukan inovasi dengan meningkatkan kualitas dan kuantitasnya (Riyyasy et al., 2023). Perkembangan teknologi digital menuntut perbankan untuk meninggalkan metode pemasaran konvensional yang bersifat satu arah dan beralih ke pendekatan yang lebih personal dan berbasis data (Fauziah et al., 2025).

Tim penjualan sering menemui kesulitan dalam mengutamakan prospek karena metode penawaran yang manual tanpa bantuan sistem analitik. Proses komunikasi dalam aktivitas penjualan membutuhkan pengeluaran yang tinggi, baik dari aspek waktu maupun materi. Untuk itu, perbankan perlu memusatkan fokus pada pengembangan prospek yang paling cocok dan menunjukkan tingkat ketertarikan yang tinggi demi mempertahankan tingkat pengembalian investasi yang menguntungkan (Hamim, 2022).

Meningkatnya kompleksitas dan daya saing pasar, bersama dengan ledakan data konsumen yang tersedia, telah membuat perbankan wajib mengadopsi teknik yang lebih canggih dan efisien untuk menghasilkan prospek (Agboola et al., 2022). Salah satu pendekatan yang saat ini banyak digunakan adalah implementasi sistem rekomendasi yang berbasis kecerdasan buatan untuk menyesuaikan tawaran produk perbankan, termasuk deposito berjangka (Hiryanto, 2025). Sebagai solusi atas permasalahan tersebut, pendekatan berbasis data melalui *predictive lead scoring* hadir untuk mengidentifikasi peluang konversi nasabah secara akurat dengan memanfaatkan data historis (Panguluri, 2025).

Penelitian ini bertujuan untuk merancang dan membangun *dashboard* dengan menggabungkan hasil prediksi skor prioritas secara langsung menggunakan algoritma *random forest*. *dashboard* ini dikembangkan dengan menggunakan stack React.js, Express.js, dan Supabase sebagai pengelola basis data. Dengan rancang

bangun ini, diharapkan LeadsUp mampu berfungsi sebagai alat bantu yang efektif untuk sektor perbankan dalam mengidentifikasi target nasabah potensial secara tepat dan efisien.

## 1.2 Perumusan Masalah

Berdasarkan latar belakang, rumusan masalah dalam penelitian ini adalah bagaimana merancang dan membangun *dashboard* LeadsUp berbasis *predictive lead scoring* untuk penentuan target nasabah?

## 1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah :

1. *Dashboard* ini dikembangkan berbasis web menggunakan stack React.js, Node.js, dan Express.js, dengan platform Supabase sebagai manajemen basis data.
2. Pengembangan difokuskan pada perancangan dan pembangunan *dashboard* LeadsUp dengan integrasi hasil *predictive lead scoring* (*high, medium, low*) dari model *machine learning* eksternal melalui API sebagai fitur utama. Penelitian ini dititikberatkan pada penyajian hasil prediksi tersebut ke dalam antarmuka *dashboard* untuk klasifikasi prospek nasabah, sehingga tidak membahas proses pelatihan (*training*) atau optimasi algoritma model *machine learning* secara mendalam.
3. *Dashboard* ini dirancang untuk dua jenis pengguna yaitu admin dan *sales*.

## 1.4 Tujuan dan Manfaat

### 1.4.1 Tujuan Penelitian

Tujuan pada penelitian ini adalah merancang dan membangun *dashboard* LeadsUp berbasis *predictive lead scoring* yang dapat memprediksi prioritas calon nasabah, sehingga mampu menjadi alat bantu bagi tim *sales* dalam menentukan target nasabah potensial.

#### 1.4.2 Manfaat Penelitian

1. Bagi bank penelitian ini diharapkan dapat membantu tim *sales* dalam menentukan target nasabah secara lebih akurat berdasarkan skor prediksi yang divisualisasikan langsung oleh *dashboard*.
2. Bagi mahasiswa penelitian ini bermanfaat sebagai sarana untuk menguji dan menerapkan keahlian pengembangan *dashboard*. Selain itu, penelitian ini disusun untuk memenuhi salah satu syarat akademik guna memperoleh gelar Sarjana Komputer pada Program Studi Informatika di Universitas Sahid Surakarta.
3. Bagi universitas hasil penelitian ini diharapkan dapat menjadi referensi tambahan bagi civitas akademika untuk penelitian selanjutnya, khususnya yang berkaitan dengan pengembangan *dashboard* yang terintegrasi dengan teknologi *machine learning*.

### 1.5 Metodologi Penelitian

Dalam merancang dan membangun *dashboard* LeadsUp ini melalui tiga tahapan utama yang saling berkesinambungan. Alur pelaksanaan dimulai dari langkah awal di lapangan, yaitu memetakan masalah nyata yang dihadapi dalam proses penentuan target nasabah perbankan. Setelah akar permasalahannya ditemukan, langkah berikutnya adalah menyusun fondasi data yang kuat serta mengkaji berbagai literatur teori yang relevan. Seluruh data dan teori yang telah terkumpul kemudian dijadikan sebagai landasan utama dalam proses perancangan, pengembangan, hingga pengujian sistem ini agar solusi yang dihasilkan benar-benar tepat sasaran.

#### 1.5.1 Identifikasi Masalah

Penelitian ini dimulai dari adanya ketidakefektifan dalam *outbound call* perbankan yang masih melakukan seleksi nasabah secara acak. Situasi ini mengakibatkan rendahnya angka konversi dan tingginya biaya operasional karena tim penjualan menghabiskan waktu pada nasabah yang tidak memiliki prospek. Nilai strategis yang ditawarkan oleh penelitian ini adalah penerapan *predictive lead scoring* menggunakan algoritma *Random forest* untuk memprediksi potensi

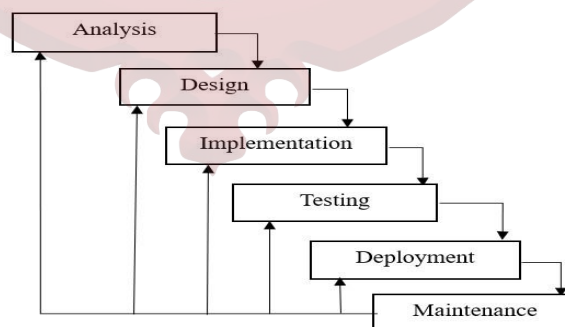
nasabah. Solusi ini mengubah proses pengambilan keputusan yang dulu bersifat subjektif menjadi berbasis data.

### 1.5.2 Pengumpulan Data dan Literatur

Tahap ini dilakukan untuk menghimpun dataset serta referensi yang diperlukan yang mendukung keberlanjutan penelitian. Dataset yang digunakan dalam penelitian ini berasal dari kampanye pemasaran langsung yang dilakukan oleh sebuah lembaga perbankan di Portugal yaitu *Bank Marketing Dataset* yang diperoleh dari *UCI Machine learning Repository*. Data tersebut berperan sebagai basis pelatihan model untuk menghasilkan skor prediksi prioritas. Setelah dataset diperoleh, dilakukan studi literatur dilakukan dengan mengkaji penelitian terdahulu mengenai algoritma *machine learning* seperti *random forest* serta mempelajari arsitektur teknologi pendukung seperti React.js, Express.js, dan manajemen basis data pada supabase guna menjamin kesesuaian pengembangan sistem.

### 1.5.3 Pengembangan Sistem

Metode yang digunakan dalam penelitian ini menggunakan model *Waterfall*. *Waterfall* adalah model SDLC (*System Development Life Cycle*) yang sering dipakai untuk membuat sistem informasi atau perangkat lunak. Metode ini cocok digunakan karena *Dashboard* yang dibuat tidak terlalu besar dan ruang lingkupnya terbatas. Pendekatan *waterfall* yang sistematis dan bertahap akan mempermudah proses pengembangan (Armanda et al., 2023). Metode *waterfall* dapat dilihat pada Gambar 1.1



Gambar 1. 1 Metode Waterfall

Sumber : (Sutikno, 2025)

Adapun tahapan dalam model *waterfall* pada penelitian ini adalah sebagai berikut:

1. *Analysis* (Analisis Kebutuhan Sistem)

Pada tahap ini, dilakukan identifikasi mendalam mengenai kebutuhan sistem baik dari sisi fungsional maupun non-fungsional. Kebutuhan fungsional difokuskan pada kemampuan *dashboard* dalam menyajikan skor prediksi nasabah, pengelolaan data *sales*, serta penyediaan fitur whatsapp *direct* untuk mempercepat komunikasi. Sementara itu, kebutuhan non-fungsional mencakup penentuan arsitektur teknologi serta pemilihan layanan *cloud* untuk memastikan sistem dapat beroperasi dengan stabil dan responsif.

2. *Design* (Desain Sistem)

Tahap ini bertujuan untuk menerjemahkan kebutuhan yang telah dianalisis ke dalam bentuk rancangan *dashboard*. Perancangan dilakukan menggunakan *Unified Modeling Language* (UML) seperti *usecase diagram*, *activity diagram*, *sequence diagram*, *class diagram*, dan *deployment diagram*. Selain itu, dilakukan perancangan UI menggunakan *figma* untuk membuat *wireframe* guna memastikan tampilan *dashboard* telah sesuai dengan kebutuhan pengguna.

3. *Implementation* (Implementasi)

Tahap implementasi merupakan realisasi dari rancangan desain ke dalam baris kode program. Pengembangan dilakukan pada sisi *frontend* menggunakan pustaka *React.js* untuk membangun komponen antarmuka yang dinamis, serta pada sisi *backend* menggunakan *Node.js* dan *Express.js* untuk membangun layanan REST API.

4. *Testing* (Pengujian)

Tahap ini bertujuan memastikan bahwa semua komponen sistem memenuhi spesifikasi dan standar kualitas. Proses verifikasi dilakukan dengan tiga metode pengujian yang berbeda. Pertama, pengujian fungsionalitas antarmuka menggunakan *Black Box Testing*. Kedua, pengujian integrasi data dan *endpoint* API dengan *Postman*. Ketiga, pengujian performa model data prediktif dengan algoritma *Random forest*.

## 5. *Deployment*

Tahap ini merupakan proses pengunggahan aplikasi ke lingkungan produksi agar *dashboard* dapat diakses secara publik melalui jaringan internet. Layanan vercel digunakan untuk melakukan hosting pada sisi frontend, sedangkan sisi backend dan *machine learning* di deploy melalui platform railway.

## 6. *Maintenance* (Pemeliharaan)

Tahap terakhir dalam metodologi ini adalah pemeliharaan sistem guna memastikan operasional *dashboard* berjalan tanpa kendala setelah diimplementasikan. Proses ini mencakup pemantauan berkala terhadap performa API, perbaikan bug yang mungkin muncul saat sistem digunakan, serta penyesuaian fitur apabila terdapat perkembangan kebutuhan dari sisi pengguna.

### 1.6 Sistematika Penulisan

Penulisan laporan penelitian ini disusun dengan sistematika yang terstruktur ke dalam lima bab utama, dengan rincian sebagai berikut :

#### **BAB I PENDAHULUAN**

Bab ini akan menguraikan latar belakang masalah terkait efisiensi akuisisi nasabah deposito, perumusan masalah, batasan masalah pada *dashboard* LeadsUp, tujuan dan manfaat penelitian, metodologi penelitian yang digunakan serta sistematika penulisan laporan.

#### **BAB II LANDASAN TEORI**

Bab ini akan memaparkan teori-teori dasar yang menjadi landasan penelitian, meliputi penelitian terdahulu, kerangka berfikir, konsep *predictive lead scoring*, cara kerja algoritma *random forest*, serta penjelasan teknis mengenai stack teknologi yang digunakan seperti React.js, Node.js, Supabase, dan konsep integrasi sistem melalui RESTful API.

#### **BAB III ANALISIS DAN PERANCANGAN SISTEM**

Bab ini akan menguraikan tahap analisis kebutuhan fungsional dan non fungsional sistem, desain arsitektur sistem, perancangan antarmuka *dashboard*, serta perancangan model *predictive lead scoring* dengan algoritma *random forest*.

#### **BAB IV IMPLEMENTASI SISTEM DAN ANALISIS HASIL**

Bab ini akan memaparkan proses implementasi seluruh komponen sistem, termasuk pembangunan model *machine learning*, pengembangan *frontend* dan *backend*, integrasi sistem, serta hasil pengujian dan evaluasi model dan sistem secara keseluruhan.

#### **BAB V SIMPULAN DAN SARAN**

Bab ini akan menyajikan simpulan akhir dari seluruh rangkaian penelitian mengenai efektivitas sistem LeadsUp, menjawab rumusan masalah dan tujuan penelitian, serta memberikan saran dan rekomendasi untuk pengembangan sistem lebih lanjut.



## BAB II

### LANDASAN TEORI

#### 2.1 Tinjauan Pustaka

Penelitian yang berjudul "*Smart Sales: Amplifying the Power of Predictive lead scoring in B2B Sales*" bertujuan untuk mengembangkan model penilaian prospek prediktif (*predictive lead scoring model*) yang lebih objektif dan berbasis data dengan mengintegrasikan perspektif serta niat beli prospek (*leads*) di setiap fase proses penjualan B2B. Penelitian ini menerapkan metode *clustering-then-classification*. Algoritma yang digunakan meliputi *K-Prototypes* untuk pengelompokan, *Gradient Boosting Trees* (GBT) untuk klasifikasi, serta teknik SMOTE untuk menangani ketidakseimbangan data (*class imbalance*). Hasilnya, pendekatan ini terbukti mampu mengoptimalkan kualitas daftar prospek, menurunkan biaya kesalahan prediksi, dan memberikan panduan strategis bagi tim penjualan untuk memprioritaskan prospek berdasarkan skala perusahaan serta kesiapan membeli (Wu et al., 2024).

Penelitian yang berjudul "*AI-Powered Predictive lead scoring: Transforming Sales Prioritization through Data-Driven Insights*" bertujuan untuk mengeksplorasi transformasi proses penjualan melalui penilaian prospek prediktif (*predictive lead scoring*) berbasis AI, yang memungkinkan organisasi memprioritaskan prospek secara ilmiah berdasarkan kemungkinan terbesar mereka untuk melakukan konversi menjadi pelanggan. Penelitian ini mengeksplorasi penggunaan algoritma *Random Forest*, *XGBoost*, *Logistic Regression*, dan *Neural Networks*. Implementasi teknologi ini terbukti secara signifikan meningkatkan akurasi kualifikasi prospek, memperpendek siklus penjualan, memangkas waktu operasional untuk prospek bernilai rendah, meningkatkan nilai rata-rata kesepakatan, serta menurunkan biaya akuisisi pelanggan secara efisien (Panguluri, 2025).

Penelitian yang berjudul "*AI-Driven Lead Scoring in Salesforce: Using Machine learning Models to Prioritize High-Value Leads and Optimize Conversion Rates*" menganalisis otomatisasi penilaian prospek *real-time* di platform Salesforce. Algoritma yang digunakan dalam penelitian ini meliputi Regresi

Logistik (*Logistic Regression*), *Random forest*, *XGBoost*, *LightGBM*, dan Jaringan Saraf (*Neural Networks*), penelitian ini membuktikan bahwa sistem berbasis AI lebih unggul dibandingkan sistem berbasis aturan (*rules-based*) yang kaku. Manfaat utamanya meliputi peningkatan efisiensi keterlibatan, kenaikan tingkat konversi, serta terciptanya keselarasan (*alignment*) yang lebih harmonis antara tim pemasaran dan tim penjualan (Tarra & Mittapelly, 2024).

Penelitian yang berjudul "*Predicting the Success of Bank Telemarketing for Selling Long-term Deposits: An Application of Machine learning Algorithms*" Fokus utamanya adalah memprediksi keberhasilan kampanye telemarketing perbankan di Portugal menggunakan lima algoritma: *Logistic Regression*, *Decision Tree*, *Random Forest*, *Support Vector Machine* (SVM), dan *Gaussian Naive Bayes*. Dataset yang bersumber dari Repositori Pembelajaran Mesin UCI (*UCI Machine learning Repository*). Dataset ini berisi data kampanye pemasaran langsung (panggilan telepon) dari sebuah institusi perbankan di Portugal. Penelitian ini menyimpulkan bahwa *Logistic Regression* memberikan performa terbaik dan sangat direkomendasikan untuk memprediksi minat nasabah terhadap produk deposito jangka panjang (Borugadda et al., 2021).

Sejalan dengan penelitian sebelumnya, penelitian yang berjudul "*Prediksi Nasabah Potensial Deposito Berjangka Menggunakan Regresi Logistik dan Support Vector Machine*" berupaya mengidentifikasi nasabah potensial guna mengoptimalkan strategi pemasaran langsung. Dengan membandingkan algoritma *Logistic Regression* dan *Support Vector Machine* (SVM) pada *dataset* yang sama (Bank Marketing Dataset), hasil penelitian menunjukkan bahwa *Logistic Regression* sedikit lebih unggul dalam meminimalkan kesalahan prediksi. Selain itu, penggunaan teknik *Cross-Validation* terbukti membuat model jauh lebih stabil dan memiliki kemampuan generalisasi data yang lebih baik. (Saputra et al., 2025).

Kelima penelitian terdahulu di atas memiliki fokus pada pembuktian efektivitas algoritma *machine learning* dan AI dalam meningkatkan akurasi penilaian potensi prospek (*predictive lead scoring*) demi mengoptimalkan konversi penjualan. Namun, seluruh penelitian tersebut masih terbatas pada tahap eksperimen model komparatif di dalam lingkungan simulasi atau sangat bergantung pada platform

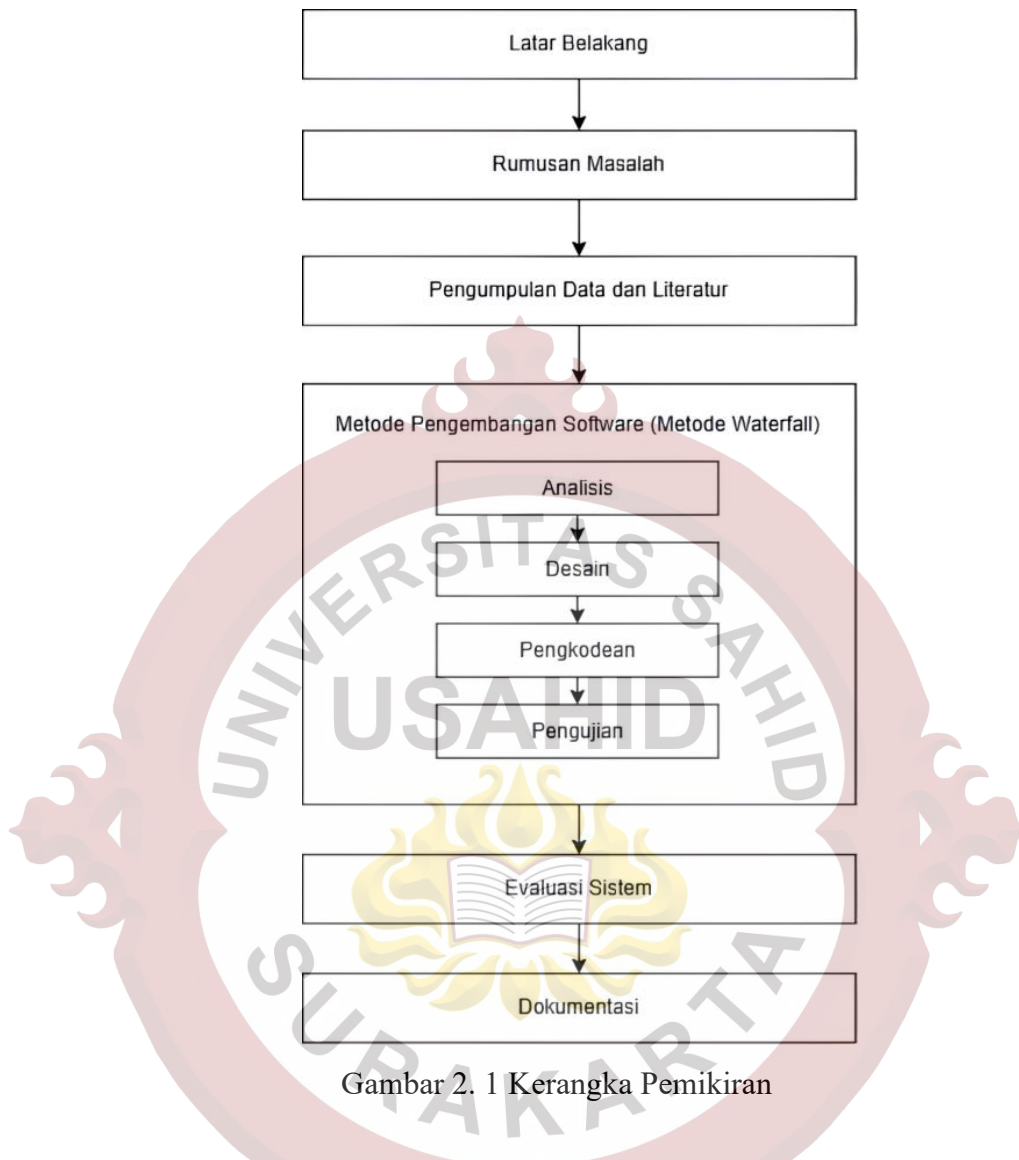
CRM pihak ketiga berskala besar yang berbayar dan kaku, tanpa menyediakan rancang bangun web mandiri yang menyediakan fitur tindakan langsung tim lapangan.

Oleh karena itu, penelitian ini hadir untuk mengisi celah tersebut melalui perancangan dan pembangunan *dashboard* LeadsUp berbasis *predictive lead scoring* yang mengandalkan model *machine learning* untuk mengklasifikasikan tingkat prioritas nasabah, dengan arsitektur teknologi modern berbasis React.js, Express.js, dan Supabase.

Sistem ini memiliki kebaruan dalam mengintegrasikan hasil prediksi skor prioritas secara langsung ke dalam *dashboard*, sehingga memungkinkan tim *sales* melakukan *filtering* nasabah berdasarkan kategori prioritas. Melalui fitur ini, tim *sales* dapat menentukan target nasabah potensial untuk segera dihubungi menggunakan fitur whatsapp *direct* yang terintegrasi di dalam sistem.

## 2.2 Kerangka Pemikiran

Kerangka pemikiran dalam penelitian ini merupakan pola pikir yang mendasari seluruh alur penelitian, mulai dari identifikasi masalah, pemilihan metode, hingga implementasi sistem untuk mencapai tujuan yang diharapkan. Kerangka pemikiran ini menggambarkan tahapan sistematis dalam merancang dan membangun *dashboard* LeadsUp berbasis *predictive lead scoring* untuk membantu penentuan target nasabah. Kerangka pemikiran dalam dilihat pada Gambar 2.1



Gambar 2. 1 Kerangka Pemikiran

## 2.3 Landasan Teori

### 2.3.1 Rancang bangun

Menurut (Talia et al., 2024) Pengertian Rancang bangun adalah serangkaian intruksi untuk menerjemahkan hasil analisis ke dalam bahasa pemrograman, baik dalam menciptakan sistem yang baru atau memperbaiki sistem yang lama. Sejalan dengan itu (parjito, Oktavia, 2022) menegaskan bahwa rancang bangun merupakan suatu proses yang meliputi perancangan dan penerapan sistem, baik untuk membangun sistem yang benar-benar baru, maupun untuk mengganti atau menyempurnakan sistem yang sudah ada, baik secara menyeluruh maupun hanya

sebagian. Proses ini diharapkan dapat memberikan jalan keluar bagi permasalahan yang sedang dihadapi (parjito, Oktavia, 2022).

Dalam penelitian ini, konsep rancang bangun diterapkan untuk mentransformasi hasil analisis kebutuhan pemasaran perbankan konvensional ke dalam sebuah *dashboard* berbasis web yang fungsional. Proses rancang bangun pada *dashboard* LeadsUp ini mencakup tahap perancangan visual antarmuka menggunakan figma hingga tahap implementasi kode program menggunakan *stack* teknologi React.js, Express.js, dan Supabase. Rancang bangun ini bertujuan untuk menyempurnakan sistem penentuan target nasabah yang sebelumnya bersifat acak dan subjektif menjadi sistem yang terstruktur, otomatis, dan berbasis data.

### 2.3.2 *Dashboard*

Menurut (Mardi et al., 2023) Pengertian *Dashboard* adalah tampilan visual informasi penting yang hanya berisi data yang diperlukan untuk mencapai satu atau lebih tujuan tertentu. Tampilan visual memiliki arti bahwa penyajian data harus diatur dengan baik agar mata manusia dapat menerima informasi dengan cepat dan otak manusia mampu memahami maksudnya dengan benar (Geni et al., 2023).

Dalam penelitian ini, konsep *dashboard* diterapkan secara spesifik ke dalam *dashboard* LeadsUp untuk menyajikan ringkasan data pemasaran perbankan secara terpusat, ringkas, dan mudah dipahami. Penyajian visual yang terstruktur ini bertujuan agar *admin* dan tim *sales* dapat langsung menangkap status kerja serta menentukan target nasabah potensial secara cepat dan akurat tanpa harus menganalisis data mentah yang rumit.

### 2.3.3 Predictive Leads Scoring

Menurut (Panguluri, 2025) pengertian *predictive lead scoring* adalah metodologi yang memanfaatkan *machine learning* untuk menganalisis data untuk memberikan nilai numerik kepada prospek berdasarkan kemungkinan mereka untuk berkonversi menjadi pelanggan.

Dalam penelitian ini, metodologi *predictive lead scoring* diterapkan ke dalam *dashboard* LeadsUp untuk mengotomatiskan proses kualifikasi target nasabah. Proses ini diawali dengan mekanisme unggah data oleh *admin* menggunakan fitur Import csv ke dalam basis data supabase, yang kemudian secara

otomatis diklasifikasikan oleh sistem menjadi label prioritas kerja (*High, Medium, Low*) pada tabel distribusi. Penerapan metodologi ini bertujuan untuk menggantikan penentuan target nasabah konvensional yang subjektif menjadi berbasis data ilmiah, sehingga tim *sales* dapat mengeksekusi prioritas tertinggi terlebih dahulu demi meningkatkan efisiensi waktu dan konversi penjualan.

#### 2.3.4 *Machine learning*

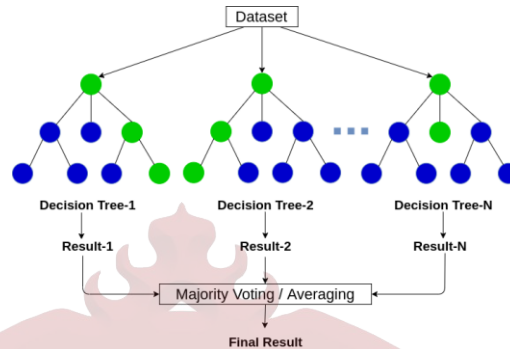
Menurut (Chyan et al., 2024) pengertian *machine learning* merupakan bagian dari kecerdasan buatan yang menginstruksikan komputer untuk mempelajari informasi dari data dan mengeluarkan prediksi atau tindakan berdasarkan interpretasi terhadap data tersebut. *Machine learning* memiliki cabang yang memanfaatkan algoritma jaringan saraf tiruan untuk beradaptasi dengan berbagai data dalam jumlah besar. Algoritma ini memungkinkan mesin untuk mengidentifikasi pola dalam data yang tidak terstruktur atau data dengan fitur yang tidak dapat ditentukan secara langsung. *Machine learning* memerlukan data yang valid sebagai sumber belajar (selama proses training) sebelum diuji untuk mendapatkan hasil yang optimal (Pandia et al., 2024).

Dalam penelitian ini, konsep *machine learning* diterapkan sebagai otak kecerdasan *dashboard* LeadsUp untuk melakukan pengenalan pola historis nasabah perbankan. Sebelum sistem dapat digunakan untuk memprediksi secara *real-time*, proses *training* (pembelajaran model) dilakukan terlebih dahulu menggunakan dataset kampanye pemasaran bank yang valid untuk melatih algoritma *random forest*. Setelah model mendapatkan hasil evaluasi parameter dan akurasi yang optimal melalui proses pengujian (*testing*), barulah formula model prediktif tersebut diintegrasikan ke dalam sistem utama agar siap mengklasifikasikan setiap data prospek baru yang masuk secara *real time*.

#### 2.3.5 *Random forest*

*Random forest* adalah *ensemble learning approach* yang populer karena adaptabilitas dan akurasinya, mampu menangani data kategorikal dan tahan terhadap *feature scaling* (Tan et al., 2024). *Random forest* adalah metode klasifikasi yang terdiri dari beberapa kumpulan pohon keputusan (*decision tree*) yang menerapkan prinsip *majority vote* (suara terbanyak) untuk menentukan suatu

keputusan akhir dalam klasifikasi (Ramadhan, 2025). Jalur kerja algoritma *random forest* ditunjukkan pada Gambar 2.2.



Gambar 2. 2 Struktur Skema Kerja *Random forest*

Sumber : (Feby, 2023)

Secara matematis, rumus prediksi akhir *random forest* untuk klasifikasi dapat dinyatakan pada persamaan 2.1 (Modami et al., 2025):

$$\hat{y} = \text{mode}(h_1(x), h_2(x), \dots, h_n(x)) \quad (2.1)$$

Keterangan :

$h_i(x)$  : Hasil prediksi dari pohon ke- $i$  terhadap data input  $x$

$n$  : Jumlah pohon yang digunakan dalam model.

$\hat{y}$  : Nilai prediksi akhir yang ditentukan dari kelas yang paling banyak muncul (*majority voting*)

### 2.3.6 Confusion Matrix

Metode yang digunakan untuk evaluasi adalah *confusion matrix* dengan menggunakan data testing (Amaliah & Nusrang, 2022). *Confusion matrix* merupakan alat evaluasi dalam klasifikasi yang terdiri dari empat elemen utama untuk menggambarkan kinerja model (Riyadi et al., 2026). *Confusion matrix* mengandung informasi berupa perbandingan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi sebenarnya. *Confusion matrix* dilakukan untuk mengetahui sejauh mana model bekerja sesuai dengan apa yang kita inginkan (Ramadhan, 2025). *Confusion matrix* dapat dinyatakan pada Tabel 2.1

Tabel 2. 1 Confusion Matrix

| Aktual      | Prediksi    |             |
|-------------|-------------|-------------|
|             | Negatif (1) | Positif (1) |
| Negatif (0) | TN          | FP          |
| Positif (1) | FN          | TP          |

Penjelasan dari TN, FN, FP dan TP sebagai berikut :

- 3 *True Negative (TN)* : Data negatif yang diprediksi benar sebagai negatif.
- 4 *False Positive (FP)* : Data negatif yang salah diprediksi sebagai positif.
- 5 *False Negative (FN)* : Data positif yang salah diprediksi sebagai negatif.
- 6 *True Positive (TP)* : Data positif yang diprediksi benar sebagai positif.

Berdasarkan *confusion matrix* tersebut, berikut rumus perhitungan untuk menghitung akurasi model disajikan pada Persamaan 2.2 (Azmi & Voutama, 2024):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \cdot 100\% \quad (2.2)$$

### 2.3.7 *Preprocessing Data*

*Preprocessing* data dalam pemodelan prediktif merupakan tahapan metodologis mendasar yang bertujuan untuk menyiapkan *dataset* mentah agar siap digunakan dalam analisis statistik dan aplikasi *machine learning*. *Preprocessing* data digambarkan sebagai serangkaian langkah sistematis yang dirancang untuk mengubah data mentah menjadi bentuk terstruktur yang dapat dianalisis, serta sesuai dengan asumsi dan kebutuhan operasional algoritma prediktif. Prosedur ini umumnya mencakup pembersihan data (*data cleaning*), normalisasi, transformasi, penyandian variabel kategorikal (*encoding of categorical variables*), dan pengoreksian struktur *dataset* menjadi representasi fitur yang terorganisir. Oleh karena itu, *preprocessing* data berfungsi sebagai mekanisme penting yang meningkatkan interpretabilitas dan keandalan *dataset* sebelum proses estimasi model (Miah & Islam, 2024).

### 2.3.8 *Feature Encoding*

*Feature encoding* merupakan salah satu tahapan penting dalam *preprocessing* data pada *machine learning* yang bertujuan untuk mengubah data kategorik menjadi representasi numerik. Sebagian besar algoritma *machine learning* hanya dapat memproses data dalam bentuk angka, sehingga data kategorik

seperti jenis kelamin, status pendidikan, atau kategori produk tidak dapat langsung digunakan dalam proses pemodelan. Oleh karena itu, *feature encoding* diperlukan agar informasi yang terkandung dalam data kategorik dapat dipertahankan dan dimanfaatkan secara optimal oleh model (Jupron, 2026).

Salah satu teknik *feature encoding* adalah *one-hot encoding*. *One-hot encoding* merupakan metode yang paling sederhana untuk mengklasterkan data bertipe campuran (Fadilah & Wijayanto, 2023). *One-Hot Encoding* digunakan untuk merepresentasikan variabel kategorikal sebagai vektor biner. Setiap kategori atau label dalam metode encoding ini diubah menjadi vektor biner dengan panjang yang sama dengan jumlah total kategori yang berbeda dalam variabel tersebut. Semua nilai vektor adalah nol, kecuali indeks yang sesuai dengan kategori, yang ditandai dengan angka 1 (Herdian et al., 2024).

#### 2.3.9 Standard Scaler

*Standard Scaler* atau yang lebih umum dikenal sebagai teknik standardisasi, mentransformasikan *dataset* agar memiliki nilai rata-rata (*mean*) sebesar nol dengan standar deviasi sebesar satu (Thakker & H. Buch, 2024). Nilai hasil transformasi tersebut, atau yang sering disebut sebagai *z-score*, dihitung menggunakan persamaan 2.3 (Fadilah & Wijayanto, 2023).

$$z = \frac{x - \mu}{\sigma} \quad (2.3)$$

Keterangan :

$z$  = Nilai hasil standardisasi (*z-score*)

$x$  = Nilai fitur asli

$\mu$  = Nilai rata-rata (*mean*)

$\sigma$  = Standar deviasi

Persamaan ini memastikan bahwa nilai rata-rata dari data hasil transformasi akan bernilai 0, dan standar deviasinya bernilai 1.

#### 2.3.10 *Splitting* Data

*Splitting* data merupakan sebuah praktik yang dianggap sangat penting dan mutlak diperlukan untuk menghilangkan atau mengurangi bias terhadap data pelatihan dalam model *Machine Learning*. Proses ini dilakukan untuk mencegah

algoritma *machine learning* mengalami *overfitting* yaitu suatu kondisi di mana model berkinerja buruk saat diuji dengan data uji yang sebenarnya. *data splitting* adalah pemisahan data menjadi beberapa subhimpunan secara terpisah yang masing-masing digunakan untuk pelatihan dan evaluasi model. Pemisahan ini secara umum membagi data menjadi set pelatihan, pengujian, atau validasi (Muraina & Olaniyi, 2022).

#### 2.3.11 Ambang Batas Skor (*Threshold*)

Ambang batas skor (*threshold*) dalam *lead scoring* merupakan acuan kuantitatif yang digunakan untuk mengkategorikan tingkat kesiapan prospek (*leads*) berdasarkan akumulasi bobot nilai kriteria demografis dan perilaku. Penetapan *threshold* ini berfungsi sebagai dasar pengambilan keputusan bagi tim sales dan marketing dalam menentukan prioritas serta strategi *follow-up* yang tepat (Pusparini, 2026).

#### 2.3.12 Deposito Berjangka

Berdasarkan Undang-Undang Republik Indonesia Nomor 10 Tahun 1998 tentang Perbankan, deposito berjangka didefinisikan sebagai simpanan yang penarikannya hanya dapat dilakukan pada waktu tertentu berdasarkan perjanjian nasabah penyimpan dengan bank (Pratiwi et al., 2022). Artinya, jika klien kontributor menyimpan uang untuk jangka waktu 90 hari, maka uang tersebut harus dicairkan setelah jangka waktu tertentu berakhir dan sering disebut dengan tanggal jatuh tempo (Hasugian et al., 2024).

Dalam penelitian ini, deposito berjangka merupakan produk instrumen perbankan utama yang menjadi objek target penawaran dalam *dashboard* LeadsUp. *Dashboard* ini dirancang untuk menganalisis data historis untuk mengenali profil nasabah yang memiliki kecenderungan finansial dan minat paling tinggi terhadap produk deposito berjangka. Melalui visualisasi hasil prediksi ini, tim *sales* dapat menghemat energi operasional dengan mengabaikan nasabah bernilai rendah dan berfokus menawarkan produk deposito berjangka hanya kepada target nasabah yang masuk dalam kategori prioritas tinggi.

#### 2.3.13 React.js

React.js adalah sebuah library JavaScript yang bersifat *open source* untuk membangun antarmuka pengguna (*User Interface*) yang interaktif dan dinamis. React.js memungkinkan pengembang untuk membuat komponen-komponen UI yang dapat digunakan kembali secara modular, sehingga mempermudah pengembangan aplikasi web berskala besar maupun kompleks. Keunggulan utama React.js terletak pada proses rendering halaman web yang efisien dan responsif terhadap input pengguna, karena React hanya memperbarui bagian-bagian tertentu dari halaman yang benar-benar berubah tanpa perlu melakukan refresh ulang pada keseluruhan halaman (Murti, 2022).

Dalam penelitian ini, React.js digunakan sebagai fondasi utama dalam membangun antarmuka (*frontend*) *dashboard* LeadsUp. Sifatnya yang modular diterapkan untuk menyusun komponen-komponen sistem seperti kartu indikator statistik (*metric cards*), grafik analitik, dan tabel data nasabah secara terpisah agar mudah dikembangkan. Selain itu, kemampuan React.js dalam memperbarui data secara parsial (tanpa memuat ulang seluruh halaman) dimanfaatkan untuk memastikan bahwa ketika tim *sales* melakukan *filtering* data atau memperbarui status kontak nasabah, antarmuka aplikasi tetap berjalan dengan ringan, responsif, dan interaktif di lapangan.

#### 2.3.14 Tailwind CSS

Tailwind CSS merupakan framework css yang menggunakan pendekatan *utility-first* untuk membangun antarmuka pengguna secara cepat tanpa harus meninggalkan kode HTML (Santoso, 2025). Tailwind CSS dilengkapi dengan fitur PurgeCSS yang secara otomatis menghapus kelas CSS yang tidak digunakan dalam produksi. Hal ini menghasilkan ukuran file CSS yang sangat kecil, sehingga mempercepat waktu pemuatan halaman pada *dashboard* (Deo Eka Putra et al., 2025).

Dalam penelitian ini, Tailwind CSS digunakan sebagai alat utama untuk merancang tata letak (*layout*) dan estetika visual pada antarmuka *dashboard* LeadsUp. Pendekatan *utility-first* dari *framework* ini diterapkan untuk mempercepat proses kustomisasi komponen UI, seperti penataan warna indikator

pada kartu statistik (*metric cards*) serta pewarnaan label kategori pada kolom prioritas (*High, Medium, Low*).

#### 2.3.15 Express.js

Express.js adalah framework web berbahasa JavaScript yang digunakan untuk membangun *backend* secara efisien. Express menyediakan metode untuk permintaan HTTP seperti GET, POST, dan PUT, serta mendukung routing melalui templat URL, penentuan mesin templat, dan metode respons. Framework ini juga dilengkapi middleware untuk menambah fungsi tambahan seperti manajemen cookie dan sesi (Kholisho & Amri, 2025).

Dalam penelitian ini, Express.js digunakan sebagai fondasi utama dalam membangun arsitektur *backend* atau *Application Programming Interface* (API) pada *dashboard* LeadsUp. Metode HTTP yang disediakan oleh Express.js diterapkan secara spesifik untuk menangani lalu lintas data operasional.

#### 2.3.16 Node.js

Node.js adalah perangkat lunak yang dibuat untuk mengembangkan aplikasi web menggunakan JavaScript. Node.js memungkinkan JavaScript berperan sebagai bahasa pemrograman sisi server, setara dengan PHP, Ruby, dan Perl (Fahmi, 2024). Dalam penelitian ini, Node.js diterapkan sebagai lingkungan eksekusi (*runtime environment*) di sisi *server* untuk menjalankan seluruh arsitektur *backend dashboard* LeadsUp. Dengan memanfaatkan Node.js, seluruh kode backend yang dibangun menggunakan *framework* Express.js dapat dieksekusi secara non-blokir (*non-blocking I/O*). Karakteristik ini dimanfaatkan secara spesifik untuk memastikan bahwa *server* tetap responsif dan memiliki performa tinggi ketika menangani proses data secara bersamaan.

#### 2.3.17 Supabase

Supabase didefinisikan sebagai sebuah platform yang menyediakan kumpulan alat (*collection of tools*) untuk membangun aplikasi. Platform ini menawarkan berbagai layanan *backend* seperti *autentikasi, database, penyimpanan file, API* yang dihasilkan secara otomatis, serta *edge functions*. Supabase didirikan pada tahun 2020 dan dipasarkan sebagai alternatif sumber terbuka (*open source*)

dari Firebase. Setiap proyek yang dibuat di Supabase sudah termasuk *database* postgres penuh yang bersifat relasional (Zewdie Ayezabu, 2022).

Dalam penelitian ini, Supabase diterapkan sebagai ekosistem *Cloud Database* dan sistem keamanan utama pada *dashboard* LeadsUp. Fitur *database* Postgres relasional dari Supabase dimanfaatkan secara terpusat untuk menyimpan dan mengorganisasi seluruh manajemen data sistem, yang meliputi data kredensial pengguna, data profil prospek nasabah, hingga riwayat aktivitas log operasional *dashboard*.

#### 2.3.18 RESTful API

RESTful API (*Representational State Transfer Application Programming Interface*) adalah gaya arsitektur untuk membangun layanan web yang mengikuti prinsip REST (Golmohammadi et al., 2023). Arsitektur ini memungkinkan komunikasi antar sistem berbeda melalui protokol HTTP dengan karakteristik stateless, penggunaan metode standar (GET, POST, PUT, DELETE), serta format data JSON (Guntara & Azkarin, 2023).

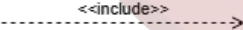
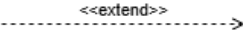
RESTful API menjadi standar industri untuk backend aplikasi modern. Pengujiannya menghadapi tantangan seperti ketergantungan pada dokumentasi OpenAPI, *autentikasi endpoint*, serta *oracle problem* dalam menentukan kebenaran respons (Ehsan et al., 2022). Dalam penelitian ini, arsitektur RESTful API diterapkan sebagai jembatan komunikasi data berbasis format JSON antara antarmuka *frontend* React.js dan *server backend* Express.js pada *dashboard* LeadsUp.

#### 2.3.19 UML

*Unified Modeling Language* (UML) adalah sebuah alat yang digunakan dalam pengembangan sistem berbasis objek atau sistem yang berkelanjutan. Dengan UML, proses perancangan sistem menjadi lebih mudah dipahami karena disajikan secara visual dan merupakan bahasa standar dalam pembuatan blueprint perangkat lunak. Beberapa diagram yang digunakan dalam perancangan berorientasi objek berbasis UML meliputi *usecase diagram*, *activity diagram*, *sequence diagram*, *class diagram*, dan *deployment diagram* (Pangestu & Voutama, 2024).

### 1. *Usecase Diagram*

*Usecase diagram* merupakan *tools* untuk mengembangkan sistem dengan menjelaskan hubungan antara *actor* yang terlibat dalam sistem. Diagram ini biasanya digunakan untuk mengkomunikasikan fungsi *high-level* dari sistem dan ruang lingkup sistem yang menggambarkan beberapa ekspektasi (harapan) dari sistem oleh praktisi di luar system (Wayahdi & Ruziq, 2023). Dapat dikatakan, *usecase diagram* merupakan sebuah pendekatan yang memfasilitasi pengembangan berpusatkan kegunaan (Hidayati et al., 2023). Simbol use case bisa dilihat pada Gambar 2.3.

| Simbol                                                                              | Nama        | Keterangan                                                                                                        |
|-------------------------------------------------------------------------------------|-------------|-------------------------------------------------------------------------------------------------------------------|
|   | Aktor       | Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan usecase                             |
|  | Use Case    | Abstraksi dan interaksi antara sistem dan aktor                                                                   |
|  | Association | Abstarksi dari penghubung antara aktor dengan use case                                                            |
|  | Include     | Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya                        |
|  | Extend      | Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi |

Gambar 2. 3 Simbol Use Case Diagram

### 2. *Activity Diagram*

*Activity diagram* menunjukkan aktivitas sistem dalam bentuk kumpulan aksi-aksi, bagaimana masing-masing aksi tersebut dimulai, keputusan yang mungkin terjadi hingga berakhirnya aksi (Ramadani, 2025). Suatu aktivitas dapat dilakukan dengan satu atau lebih *usecase*. Aktivitas menggambarkan proses yang sedang

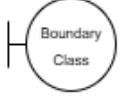
berjalan (Elis & Voutama, 2022). Simbol activity diagram bisa dilihat pada Gambar 2.4.

| Simbol                                                                              | Nama                     | Keterangan                                                                                   |
|-------------------------------------------------------------------------------------|--------------------------|----------------------------------------------------------------------------------------------|
|    | Status awal              | Sebuah diagram memiliki status awal                                                          |
|    | Aktivitas                | Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja                |
|    | Percabangan/<br>Decision | Percabangan dimana ada pilihan aktivitas yang lebih dari satu                                |
|    | Penggabungan/Join        | Penggabungan dimana yang mana lebih dari satu aktivitas lalu digabungkan jadi satu.          |
|   | Status akhir             | Status akhir yang dilakukan sistem sebuah diagram aktivitas memiliki sebuah status akhir     |
|  | Swimlane                 | Swimlane memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi |

Gambar 2. 4 Simbol Activity Diagram

### 3. Sequence Diagram

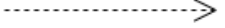
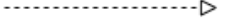
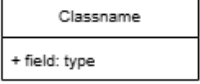
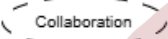
Diagram sekuen menggambarkan kelakuan objek pada *usecase* dengan pendekatan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek (Syaqila et al., 2024). Oleh karena itu untuk menggambar diagram *sequence* maka harus diketahui objek-objek yang terlibat dalam sebuah *usecase* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu (Hendri & Anna, 2024). Simbol *sequence diagram* bisa dilihat pada Gambar 2.5.

| Simbol                                                                             | Nama           | Keterangan                                                               |
|------------------------------------------------------------------------------------|----------------|--------------------------------------------------------------------------|
|   | Entity Class   | Gambaran sistem sebagai landasan dalam menyusun basis data               |
|   | Boundary Class | Menangani komunikasi antar lingkungan sistem                             |
|   | Control Class  | Bertanggung jawab terhadap kelas-kelas terhadap objek yang berisi logika |
|   | Recursive      | Pesan untuk dirinya                                                      |
|   | Activation     | Mewakili proses durasi aktivitas sebuah operasi                          |
|  | Life Line      | Komponen yang digambarkan garis putus-putus dengan objek                 |

Gambar 2. 5 Simbol Sequence Diagram

#### 4. Class Diagram

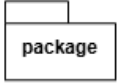
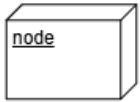
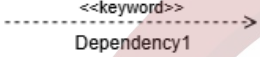
*Class diagram* merupakan diagram yang menunjukkan *class* sistem dan hubungannya secara logic (Revita, 2023). *Class diagram* merupakan inti dari proses pemodelan objek. *Tool* yang membantu dalam visualisasi struktur kelas dari suatu sistem. Hal ini disebabkan karena *class* adalah deskripsi kelompok obyek-obyek dengan *property*, perilaku (operasi) dan relasi yang sama. Disamping itu class diagram bisa memberikan pandangan global atas sebuah sistem (Novianto & Purwanto, 2022). Simbol *class diagram* bisa dilihat pada Gambar 2.6.

| Simbol                                                                            | Nama           | Keterangan                                                                                                          |
|-----------------------------------------------------------------------------------|----------------|---------------------------------------------------------------------------------------------------------------------|
|  | Generalization | Relasi antar kelas dengan makna generalisasi spesialisasi                                                           |
|  | Association    | Relasi antar kelas dengan arti umum, asosiasi biasanya juga disertai dengan multiplicity                            |
|  | Dependency     | Relasi antar kelas dengan makna kebergantungan dengan kelas                                                         |
|  | Realization    | Operasi yang benar-benar dilakukan oleh suatu objek                                                                 |
|  | Class          | Hubungan dari objek-objek yang berbagi atribut serta operasi yang sama                                              |
|  | Collaboration  | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor |

Gambar 2. 6 Simbol Class Diagram

### 5. *Deployment Diagram*

*Deployment diagram* menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi (Irianto et al., 2021). *Deployment diagram* digunakan untuk memvisualisasikan hubungan antara Software dan Hardware. Secara spesifik *deployment diagram* dapat membuat *physical model* tentang bagaimana komponen perangkat lunak (artefak) digunakan pada komponen perangkat keras, yang dikenal sebagai node (Pulungan & Purwanto, 2021). Simbol *deployment diagram* bisa dilihat pada Gambar 2.7.

| Simbol                                                                              | Nama       | Keterangan                                                                                                                                                                                                 |
|-------------------------------------------------------------------------------------|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|    | Package    | Package : Sebuah bungkusan dari satu atau lebih komponen                                                                                                                                                   |
|    | Node       | Biasanya mencakup pada perangkat keras, perangkat lunak yang tidak dibuat sendiri, jika didalam node disertakan komponen untuk mengkonsistenkan rancangan maka komponen yang telah didefinisikan komponen. |
|    | Dependency | Ketergantungan antar node, arah panah mengarah pada node yang dipakai                                                                                                                                      |
|    | Link       | Relasi antar node                                                                                                                                                                                          |
|  | Component  | sebuah component melambangkan sebuah entitas software dalam sistem                                                                                                                                         |

Gambar 2. 7 Simbol Deployment Diagram

### 2.3.20 Black Box Testing

Metode *Black Box Testing* adalah suatu cara untuk mengidentifikasi kesalahan dalam aplikasi, seperti kesalahan pada fungsi sistem aplikasi, serta adanya menu aplikasi yang tidak tersedia (Ichsanudin & Yusuf, 2022). *Black Box Testing*, juga disebut sebagai pengujian perilaku, menekankan pada kebutuhan fungsional suatu perangkat lunak. Dalam hal ini, pengujian kotak hitam memberikan para insinyur perangkat lunak serangkaian situasi input yang secara sempurna memenuhi keseluruhan kebutuhan fungsional aplikasi (Raihan et al., 2023).

Dalam penelitian ini, metode *Black Box Testing* diterapkan untuk menguji seluruh aspek fungsionalitas antarmuka pada *dashboard* LeadsUp. Pengujian dilakukan dengan mensimulasikan berbagai skenario aksi pengguna tanpa

memeriksa struktur kode internal aplikasi. Melalui pengujian ini, sistem dipastikan bebas dari galat (*error*) fungsional dan siap digunakan oleh tim *sales* maupun *admin* secara optimal.

#### 2.3.21 Postman

Postman merupakan aplikasi klien API untuk pengembang yang mempermudah pembuatan, distribusi, pengujian, dan dokumentasi API (Kore et al., 2022). Postman digunakan untuk memastikan bahwa setiap *endpoint* API beroperasi sebagaimana mestinya dan memberikan respons yang diharapkan pada tahap pengembangan. Platform ini bertugas mengirimkan permintaan HTTP ke server Express.js untuk menguji *endpoint* API, serta memeriksa data respons yang diberikan oleh server (Sri et al., 2024).

Dalam penelitian ini, postman digunakan sebagai perangkat lunak utama pada tahap pengembangan untuk melakukan pengujian *backend* pada *dashboard* LeadsUp. Sebelum antarmuka *frontend* dihubungkan, postman dimanfaatkan untuk mengirimkan permintaan HTTP (*request simulation*) ke berbagai *endpoint* fungsional. Pengujian melalui postman ini bertujuan untuk memvalidasi bahwa struktur respons data yang dikembalikan oleh *server* sudah akurat, terenkripsi dengan token keamanan, serta menghasilkan format data JSON yang tepat dan siap digunakan oleh *frontend*.

#### 2.3.22 Diagram Alir

Diagram alir atau flowchart adalah representasi visual berupa diagram dari sebuah algoritma. Diagram ini terdiri atas berbagai bentuk grafik dua dimensi (*node*) yang saling terhubung oleh garis penghubung. Setiap bentuk grafik pada diagram alir merepresentasikan operasi tertentu dan biasanya terdapat teks di dalamnya. Dengan menggunakan diagram alir, proses-proses yang terjadi dalam suatu sistem/algoritma dapat divisualisasikan dengan bentuk-bentuk yang lebih mudah dipahami (Suryawan et al., 2023). Simbol diagram alir bisa dilihat pada Gambar 2.8.

| Simbol                                                                              | Nama                  | Keterangan                                            |
|-------------------------------------------------------------------------------------|-----------------------|-------------------------------------------------------|
|    | Garis Alir            | Menunjukkan arah aliran algoritma                     |
|    | Terminator            | Permulaan atau akhir program                          |
|    | Proses                | menyatakan kegiatan yang akan terjadi di diagram alir |
|   | Input/<br>output data | Proses input/output data, parameter, informasi        |
|  | Titik<br>Keputusan    | Proses atau langkah dimana perlu adanya keputusan     |

Gambar 2. 8 Simbol Diagram Alir