BAB I
PENDAHULUAN

1.1 Latar Belakang

Penyajian hasil analisis memerlukan media yang efektif untuk menyajikan informasi secara ringkas dan mudah dipahami. *Learning Analytics Dashboard* memungkinkan data yang kompleks disajikan dalam bentuk visual seperti grafik, indikator performa, serta ringkasan statistik yang memudahkan pengguna dalam membaca pola dan tren pembelajaran. Studi terbaru menunjukkan bahwa penggunaan *dashboard* dalam konteks pendidikan mampu membantu dalam mengidentifikasi pola belajar siswa serta mendukung proses pengambilan keputusan berbasis data (Hakim, 2025).

Setiap siswa memiliki karakteristik aktivitas pembelajaran dan hasil belajar yang berbeda-beda. Perbedaan tersebut dapat terlihat dari tingkat penyelesaian modul, frekuensi pengerjaan kuis, jumlah pengulangan kuis, tingkat keberhasilan tugas, serta capaian nilai yang diperoleh selama proses pembelajaran. Oleh karena itu, diperlukan suatu mekanisme yang mampu mengelompokkan siswa berdasarkan pola aktivitas dan hasil belajar yang dimiliki sehingga karakteristik masing-masing kelompok dapat diidentifikasi dengan lebih mudah. Melalui proses klasifikasi, siswa dapat dikelompokkan ke dalam beberapa tipe pembelajaran berdasarkan karakteristik aktivitas pembelajaran dan hasil belajar yang dimiliki. Informasi tersebut dapat membantu pendidik atau pengelola pembelajaran dalam melakukan monitoring dan menentukan strategi pendampingan yang lebih tepat sesuai karakteristik masing-masing siswa.

Data aktivitas pembelajaran dan hasil belajar siswa dapat dimanfaatkan untuk memperoleh informasi mengenai karakteristik pembelajaran setiap siswa. Namun, informasi tersebut tidak hanya perlu disajikan dalam bentuk visual, tetapi juga dianalisis sehingga mampu memberikan gambaran mengenai karakteristik belajar siswa. Oleh karena itu, penelitian ini mengintegrasikan visualisasi data melalui dashboard dengan proses klasifikasi tipe pembelajaran siswa menggunakan algoritma *Light Gradient Boosting Machine (LightGBM)*. Sehingga sistem yang

dikembangkan tidak hanya menyajikan informasi pembelajaran, tetapi juga menghasilkan informasi yang dapat mendukung proses monitoring dan pengambilan keputusan.

Perkembangan teknologi *machine learning* turut memberikan kontribusi dalam meningkatkan kemampuan analisis data pembelajaran. Pada penelitian ini, digunakan algoritma *Light Gradient Boosting Machine (LightGBM)* karena memiliki kemampuan yang baik dalam menangani data tabular, efisien dalam proses pelatihan, serta mampu menghasilkan performa klasifikasi yang tinggi.

Beberapa penelitian terbaru juga menunjukkan bahwa algoritma *LightGBM* memiliki performa yang baik dalam berbagai kasus klasifikasi dan prediksi, khususnya pada data dengan jumlah besar dan kompleks, sehingga banyak digunakan dalam berbagai bidang, termasuk analisis data pendidikan (Maulita dan Wahid, 2024).

Berdasarkan permasalahan tersebut, penelitian ini bertujuan untuk merancang dan membangun *Dashboard* Klasifikasi Tipe Pembelajaran Siswa Berbasis Web menggunakan *Machine learning*. Sistem yang dikembangkan diharapkan mampu mengolah data aktivitas pembelajaran dan hasil belajar siswa untuk menghasilkan klasifikasi tipe pembelajaran secara otomatis. Selain itu, sistem juga dapat menyajikan informasi dalam bentuk *dashboard* yang interaktif sehingga memudahkan proses monitoring, analisis, dan pengambilan keputusan berdasarkan data yang tersedia.

1.2 Perumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka rumusan masalah dalam penelitian ini adalah: “Bagaimana merancang dan membangun sistem *dashboard* berbasis web yang mampu mengklasifikasikan tipe pembelajaran siswa menggunakan *machine learning* serta menyajikan informasi aktivitas dan hasil belajar?”

1.3 Batasan Masalah

Agar penelitian ini memiliki ruang lingkup yang jelas dan tidak menyimpang dari tujuan penelitian, maka batasan masalah yang diterapkan adalah

1. Sistem yang dirancang dan dibangun berupa dashboard berbasis web yang digunakan untuk menampilkan dan memonitor aktivitas belajar siswa, meliputi durasi belajar per hari, durasi belajar per minggu, jumlah modul yang dimulai, tingkat penyelesaian modul, jumlah percobaan kuis, rasio pengulangan kuis, jumlah pengumpulan tugas, serta hasil belajar siswa yang meliputi rata-rata nilai kuis dan tingkat keberhasilan pengumpulan tugas.
2. Sistem difokuskan pada penyajian data dalam bentuk visualisasi *dashboard* serta hasil klasifikasi tipe pembelajaran siswa, tanpa mencakup fitur autentikasi pengguna atau manajemen hak akses yang kompleks sehingga seluruh pengguna yang memiliki tautan *(URL)* aplikasi dapat mengakses sistem.
3. *Model machine learning* yang digunakan bertujuan untuk mengklasifikasikan tipe pembelajaran siswa ke dalam empat kategori, yaitu *The Overachiever, The All Rounder, The Steady Climber, dan Low-Performance Learner*.
4. Dataset yang digunakan dalam penelitian merupakan dataset simulasi (*data dummy*) yang dibuat berdasarkan karakteristik data pembelajaran siswa, sehingga tidak menggunakan data asli peserta program Dicoding.
5. Penelitian ini hanya menggunakan algoritma *Light Gradient Boosting Machine (LightGBM)* sebagai model klasifikasi dan tidak melakukan perbandingan performa dengan algoritma *machine learning* lainnya.
6. Pengujian sistem dibatasi pada pengujian fungsional menggunakan *Black Box Testing*, pengujian *API* menggunakan *Postman*, serta evaluasi performa model *machine learning* menggunakan *metrik accuracy, confusion matrix, dan classification report.*

1.4 Tujuan Dan Manfaat

a. Tujuan Penelitian

Tujuan dari penelitian ini adalah untuk merancang dan membangun sebuah sistem *dashboard* klasifikasi tipe pembelajaran siswa berbasis web dengan integrasi

machine learning yang dapat membantu proses analisis aktivitas dan hasil belajar secara lebih efektif. Sistem ini dirancang untuk mengolah data aktivitas belajar siswa dan mengklasifikasikan tipe pembelajaran berdasarkan pola perilaku serta hasil belajar yang dimiliki oleh masing-masing siswa

Melalui implementasi metode *machine learning* menggunakan algoritma *Light Gradient Boosting Machine* (*LightGBM)*, sistem diharapkan mampu menghasilkan klasifikasi tipe pembelajaran siswa secara akurat. Hasil klasifikasi tersebut kemudian disajikan dalam bentuk *dashboard* berbasis web yang informatif dan mudah dipahami.

Selain itu, penelitian ini juga bertujuan mengintegrasikan komponen *frontend, backend, basis data*, dan *model machine learning* ke dalam satu sistem yang mampu melakukan pengolahan data, klasifikasi, serta visualisasi hasil secara terintegrasi.

b. Manfaat

Manfaat yang diharapkan dari penelitian ini adalah :

1. Bagi Pengguna :

Penelitian ini diharapkan dapat membantu pengguna, khususnya admin atau pengelola pembelajaran di sekolah, dalam memonitor aktivitas dan hasil belajar siswa secara lebih mudah melalui *dashboard* berbasis web yang menyajikan visualisasi data dan hasil klasifikasi tipe pembelajaran secara informatif.

2. Bagi Mahasiswa :

Penelitian ini diharapkan dapat menjadi sarana penerapan ilmu yang telah dipelajari selama perkuliahan, khususnya dalam bidang pengembangan sistem berbasis web, basis data, dan *machine learning*. Selain itu, penelitian ini juga dapat menjadi referensi bagi mahasiswa lain yang ingin mengembangkan penelitian serupa di bidang *learning analytics* dan sistem *dashboard* pendidikan.

3. Bagi Universitas :

Penelitian ini diharapkan dapat memberikan kontribusi dalam pengembangan penelitian dan inovasi di lingkungan universitas, khususnya pada bidang teknologi informasi dan sistem cerdas dalam pendidikan. Selain itu, hasil

penelitian ini dapat menjadi tambahan referensi akademik bagi pengembangan penelitian selanjutnya.

1.5 Metodologi Penelitian

Metode pengumpulan data yang digunakan pada Tugas Akhir ini sebagai berikut:

A. Pengumpulan Data

1) Observasi

Tahap observasi dilakukan untuk memahami permasalahan yang terjadi dalam proses pembelajaran berbasis digital, khususnya terkait pemanfaatan data aktivitas pembelajaran dan hasil belajar siswa. Observasi merupakan salah satu teknik pengumpulan data yang dilakukan melalui proses pengamatan terhadap objek atau fenomena yang diteliti guna memperoleh informasi yang relevan dengan kebutuhan penelitian (Yuliana dan Ernawati, 2022).

Pada tahap ini dilakukan observasi terhadap dataset aktivitas pembelajaran dan hasil belajar siswa yang diperoleh dari pihak use *case owner*, yaitu Dicoding. Observasi dilakukan untuk memahami karakteristik data, atribut yang tersedia, serta informasi yang terkandung dalam dataset sebagai dasar dalam proses analisis kebutuhan dan perancangan sistem. Melalui observasi ini diperoleh gambaran mengenai kebutuhan sistem yang mampu mengolah dan menyajikan data pembelajaran secara terstruktur serta memberikan insight yang dapat membantu dalam proses monitoring dan evaluasi pembelajaran.

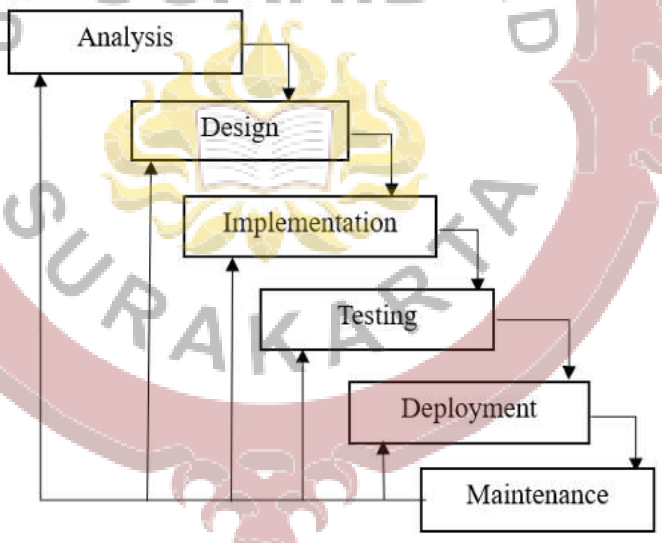
2) Studi literatur

Studi literatur dilakukan dengan mengkaji berbagai sumber ilmiah seperti jurnal, buku, serta referensi terpercaya yang berkaitan dengan topik penelitian. Studi literatur merupakan metode yang digunakan untuk mengumpulkan, menganalisis, dan mensintesis pengetahuan yang telah ada guna membangun landasan teoritis serta mengidentifikasi kesenjangan penelitian yang masih dapat dikembangkan. Tujuan dari studi literatur adalah untuk memperoleh landasan teori yang kuat, memahami perkembangan penelitian terdahulu, serta mengidentifikasi metode dan pendekatan terbaik dalam pengembangan sistem informasi (Koivu et al., 2021).

Pada penelitian ini, studi literatur difokuskan pada teori-teori terkait *Learning Analytics Dashboard, machine learning*, algoritma *Light Gradient Boosting Machine (LightGBM)*, pengembangan aplikasi web menggunakan *Next.js*, dan *PostgreSQL*.

B. Metode Pengembangan Aplikasi

Metode yang digunakan untuk pengembangan sistem ini adalah dengan metode SDLC, yaitu *Waterfall* model. *Waterfall* merupakan salah satu model *Software Development Life Cycle (SDLC)* di mana proses pengembangan perangkat lunak diawali dengan tahap spesifikasi, pengembangan, validasi, dan evolusi. Model ini kemudian membagi proses tersebut menjadi tahapan yang tersusun secara linier dimulai dari spesifikasi kebutuhan, perancangan perangkat lunak, implementasi, pengujian, dan tahap lainnya (Nurseptaji dkk., 2021). Model *Waterfall* ditampilkan pada Gambar 1.1



Gambar 1. 1 Metode Waterfall
(Nurseptaji dkk., 2021)

1. *Analysis*

Pada tahap ini dilakukan identifikasi kebutuhan sistem berdasarkan hasil observasi dan studi literatur yang telah dilakukan. Analisis kebutuhan difokuskan pada kebutuhan *dashboard* untuk menampilkan informasi aktivitas pembelajaran dan hasil belajar siswa serta kebutuhan sistem dalam melakukan klasifikasi tipe pembelajaran menggunakan *machine learning*. Hasil analisis digunakan sebagai

dasar dalam menentukan fitur-fitur yang akan dikembangkan, seperti pengelolaan data siswa, visualisasi *dashboard*, detail informasi siswa, hasil klasifikasi, serta pengiriman email peringatan kepada siswa yang termasuk kategori *Low-Performance Learner*.

2. Design

Tahap ini bertujuan untuk menerjemahkan kebutuhan yang telah dianalisis ke dalam bentuk rancangan sistem. Perancangan dilakukan menggunakan *Unified Modeling Language (UML)*, yang meliputi *Use case Diagram*, *Activity diagram*, *Sequence Diagram,* dan *Deployment Diagram*. Selain itu, dilakukan pula perancangan basis data, arsitektur sistem, serta rancangan antarmuka pengguna (*user interface*) untuk *dashboard* klasifikasi tipe pembelajaran siswa.

3. Implementation

Tahap implementasi dilakukan dengan membangun sistem *dashboard* berbasis web sesuai dengan desain yang telah dirancang pada tahap sebelumnya. Pengembangan *frontend* dilakukan untuk menampilkan *dashboard* serta hasil klasifikasi tipe pembelajaran siswa secara interaktif dan mudah dipahami. Sementara itu, *backend* dikembangkan untuk mengelola proses pengolahan data, menyediakan layanan API, serta menghubungkan sistem dengan basis data dan model *machine learning*.

Basis data digunakan untuk menyimpan data aktivitas belajar siswa, data hasil pengolahan, serta hasil klasifikasi yang dihasilkan oleh model. Pada tahap ini, sistem juga mengelola proses pengambilan dan penyimpanan data secara terstruktur agar dapat digunakan secara efisien dalam proses analisis.

Model *machine learning* dibangun menggunakan algoritma *Light Gradient Boosting Machine (LightGBM)* untuk menganalisis pola aktivitas dan hasil belajar siswa. Model ini dilatih menggunakan dataset simulasi (*data dummy*) yang telah melalui proses preprocessing, sehingga mampu menghasilkan klasifikasi tipe pembelajaran siswa secara akurat.

Selanjutnya, model yang telah dibangun diintegrasikan ke dalam sistem berbasis web melalui layanan *backend*. Proses ini memungkinkan sistem untuk

melakukan klasifikasi secara otomatis terhadap data siswa dan menampilkan hasilnya dalam bentuk visualisasi *dashboard* yang informatif.

4. *Testing*

Pada tahap ini dilakukan pengujian terhadap sistem yang telah dibangun untuk memastikan bahwa sistem berjalan sesuai dengan kebutuhan yang telah ditentukan. Metode pengujian yang digunakan adalah *blackbox testing*, yaitu metode pengujian yang berfokus pada fungsi sistem tanpa melihat struktur kode program di dalamnya.

Pengujian dilakukan dengan cara menguji setiap fitur utama pada *dashboard*, seperti proses upload dan pengolahan data, penampilan visualisasi aktivitas dan hasil belajar siswa, pengelolaan data *dashboard*, serta hasil klasifikasi tipe pembelajaran yang dihasilkan oleh sistem. Pengujian ini bertujuan untuk memastikan bahwa setiap fitur dapat berjalan sesuai dengan fungsi yang telah dirancang.

Selain menggunakan *blackbox testing*, pengujian juga dilakukan menggunakan *Postman* untuk menguji *endpoint API* yang digunakan pada sistem. Pengujian *API* dilakukan untuk memastikan proses komunikasi data antara *frontend*, *backend*, database, dan modul *machine learning* dapat berjalan dengan baik. Melalui *Postman*, proses *request* dan *response* data diuji untuk memastikan data yang dikirim maupun diterima sesuai dengan kebutuhan sistem.

Pengujian juga dilakukan terhadap model *machine learning* yang digunakan untuk klasifikasi tipe pembelajaran siswa. Pengujian model dilakukan menggunakan metrik *accuracy* untuk mengetahui tingkat ketepatan model dalam melakukan klasifikasi berdasarkan data yang digunakan. Nilai akurasi diperoleh dengan membandingkan jumlah prediksi yang benar terhadap keseluruhan data uji sehingga dapat diketahui performa model dalam menghasilkan klasifikasi tipe pembelajaran siswa.

Tahap pengujian ini bertujuan untuk memastikan bahwa seluruh fungsi *dashboard*, *integrasi API*, *database*, serta model *machine learning* dapat berjalan secara optimal dan menghasilkan output yang sesuai dengan kebutuhan analisis aktivitas dan hasil belajar siswa.

5. Deployment

Tahap *deployment* dilakukan dengan mengimplementasikan sistem ke lingkungan produksi menggunakan *Vercel* sebagai platform *hosting frontend* dan *backend.* Pada tahap ini dilakukan konfigurasi *environment variables*, koneksi ke database *PostgreSQL* melalui *Supabase*, serta integrasi model *machine learning* sehingga sistem dapat diakses melalui *web browser* dan seluruh fungsionalitasnya berjalan dengan baik.

6. Pemeliharaan Sistem (Maintenance)

Tahap terakhir adalah pemeliharaan sistem agar dapat terus digunakan secara optimal. Pemeliharaan mencakup perbaikan bug, peningkatan performa aplikasi, pemantauan integrasi sistem, serta pengembangan fitur baru sesuai kebutuhan pengguna. Selain itu, pemeliharaan juga dilakukan untuk memastikan *dashboard*, basis data, dan layanan pendukung tetap berjalan dengan stabil dan aman.

1.6 Sistematika Penulisan

Untuk memudahkan pemahaman dan penyusunan laporan Tugas Akhir ini, maka sistematika penulisan disusun sebagai berikut:

BAB I PENDAHULUAN

Bab ini berisi latar belakang masalah yang menjadi dasar penelitian, perumusan masalah, batasan masalah, tujuan dan manfaat penelitian, metodologi penelitian yang digunakan, serta sistematika penulisan laporan Tugas Akhir.

BAB II LANDASAN TEORI

Bab ini membahas teori dan konsep yang mendukung penelitian, meliputi konsep *dashboard*, *learning analytics*, klasifikasi dalam *machine learning*, serta konsep perancangan dan pembangunan sistem berbasis web. Selain itu, pada bab ini juga disajikan tinjauan pustaka yang relevan sebagai dasar penelitian.

BAB III ANALISIS DAN PERANCANGAN SISTEM

Bab ini menguraikan hasil analisis kebutuhan sistem serta perancangan *dashboard* yang dibangun. Pembahasan meliputi analisis sistem yang berjalan, analisis sistem yang diusulkan, perancangan arsitektur sistem, perancangan alur data, perancangan basis data, serta desain antarmuka pengguna (*user interface*).

BAB IV IMPLEMENTASI SISTEM DAN ANALISIS HASIL

Bab ini menjelaskan proses pembangunan sistem *dashboard* berbasis web serta integrasi model *machine learning* ke dalam sistem. Selain itu, dibahas hasil pengujian sistem menggunakan metode *blackbox testing* dan *Postman Collection* untuk memastikan bahwa sistem berjalan sesuai dengan kebutuhan yang telah ditentukan.

BAB V SIMPULAN DAN SARAN

Bab ini memuat kesimpulan yang diperoleh berdasarkan hasil penelitian yang telah dilakukan, serta saran-saran yang dapat digunakan sebagai bahan pengembangan dan penelitian selanjutnya.

UNIVERSITAS SAHID
USAHID
SURAKARTA

BAB II
LANDASAN TEORI

2.1 Tinjauan Pustaka

Penelitian terkait pemanfaatan *learning analytics* dan *dashboard* dalam pendidikan telah banyak dilakukan dalam beberapa tahun terakhir. Salah satu penelitian oleh Paulsen dan Lindsay (2024) mengkaji perkembangan learning analytics *dashboard* melalui tinjauan sistematis terhadap berbagai penelitian sebelumnya. Hasil kajian menunjukkan bahwa pengembangan *learning analytics dashboard* tidak lagi hanya berfokus pada penyajian data analitik, tetapi semakin diarahkan untuk mendukung proses pembelajaran melalui penyediaan informasi yang relevan, umpan balik, serta dukungan terhadap proses refleksi belajar siswa (Paulsen dan Lindsay, 2024).

Selanjutnya, penelitian yang dilakukan oleh Cabral et al., (2025) melakukan tinjauan sistematis terhadap penerapan *Artificial Intelligence* pada *Learning Analytics Dashboard*. Hasil kajian menunjukkan bahwa integrasi AI dalam dashboard pembelajaran mampu mendukung analisis aktivitas dan hasil belajar siswa, serta memberikan informasi yang membantu proses pemantauan pembelajaran dan mendukung *self-regulated learning* (Cabral et al., 2025).

Pada perkembangan yang lebih lanjut, penelitian oleh Masiello et al., (2024) menjelaskan bahwa *Learning Analytics Dashboard* mulai berkembang dengan mengintegrasikan teknik *predictive analytics* dan *machine learning* untuk memperluas kemampuan analisis data pembelajaran. Integrasi tersebut memungkinkan dashboard tidak hanya menyajikan informasi secara deskriptif, tetapi juga mendukung analisis prediktif terhadap kondisi belajar siswa sehingga menghasilkan insight yang lebih bermakna bagi proses pembelajaran (Masiello et al., 2024).

Penelitian lain oleh Ardianti dkk (2024) memanfaatkan metode *ensemble machine learning* untuk memprediksi kinerja siswa berdasarkan data log *Learning Management System* (LMS). Hasil penelitian menunjukkan bahwa model *machine*

learning mampu mengidentifikasi pola aktivitas belajar siswa dan menghasilkan prediksi kinerja yang dapat dimanfaatkan untuk mendukung proses monitoring serta evaluasi pembelajaran (Ardianti dkk., 2024).

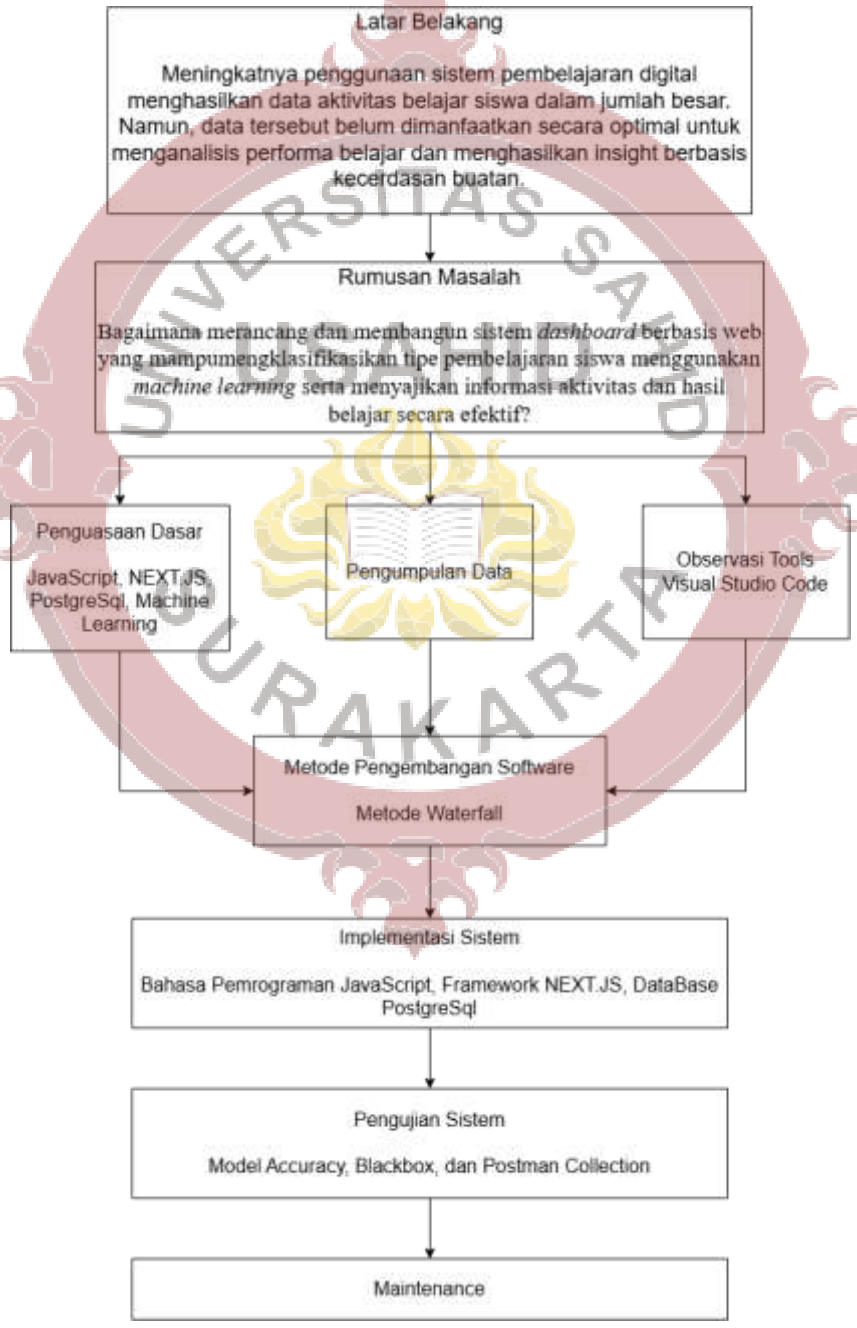
Penggunaan algoritma *Light Gradient Boosting Machine (LightGBM*) juga telah diterapkan dalam penelitian di bidang pendidikan. Penelitian oleh (Ayulani et al., 2023) membandingkan beberapa metode *tree-bsased ensemble*, yaitu *Random Forest, XGBoost,* dan *LightGBM*, untuk memprediksi performa akademik siswa berdasarkan data aktivitas pada *Learning Management System* (LMS). Hasil penelitian menunjukkan bahwa model *LightGBM* menghasilkan performa prediksi yang lebih baik dibandingkan metode lainnya pada skenario tertentu, sehingga menunjukkan potensinya sebagai algoritma yang efektif dalam analisis data pembelajaran.

Selain pada sisi analitik, pengembangan sistem berbasis web modern juga didukung oleh penggunaan *framework NEXT.JS*. Penelitian oleh Maharani (2025) menunjukkan bahwa website yang dikembangkan mampu memenuhi kebutuhan perusahaan dengan menyediakan platform digital yang mudah diakses, responsif pada berbagai perangkat, serta memiliki performa yang optimal. Selain itu, penggunaan *Next.js* dan *Tailwind CSS* memungkinkan proses pengembangan website dilakukan secara efisien dan memberikan pengalaman pengguna *(user experience)* yang lebih baik (Maharani, 2025).

Berdasarkan beberapa penelitian terdahulu tersebut, dapat disimpulkan bahwa penggunaan *learning analytics dashboard*, *algoritma machine learning* seperti *LightGBM*, serta teknologi web modern seperti *NEXT.JS* memiliki potensi besar dalam meningkatkan kualitas analisis pembelajaran dan performa sistem. Namun, masih terdapat keterbatasan dalam pengembangan sistem yang mengintegrasikan klasifikasi tipe pembelajaran siswa dengan visualisasi *dashboard* berbasis web dalam satu platform yang terintegrasi. Oleh karena itu, penelitian ini bertujuan untuk mengembangkan sistem *dashboard* yang tidak hanya menampilkan data, tetapi juga mampu mengklasifikasikan tipe pembelajaran siswa secara otomatis menggunakan pendekatan *machine learning*.

2.2 Kerangka Pemikiran

Kerangka pemikiran ini memberikan gambaran mengenai tahapan-tahapan yang dilakukan dalam penelitian terkait perancangan dan pembangunan sistem *dashboard* berbasis web dengan integrasi *machine learning* untuk klasifikasi tipe pembelajaran siswa. Kerangka pemikiran penelitian ini dapat dilihat pada Gambar 2.1.



Gambar 2. 1 Kerangka Pemikiran

2.3 Landasan Teori

2.3.1. Rancang Bangun Sistem Informasi

Rancang bangun merupakan proses perencanaan, perancangan, dan pembangunan suatu sistem yang dilakukan secara sistematis untuk menghasilkan solusi sesuai dengan kebutuhan pengguna. Proses ini mencakup penerjemahan hasil analisis kebutuhan ke dalam bentuk desain sistem yang kemudian diimplementasikan menjadi sebuah aplikasi atau sistem informasi yang dapat berfungsi sesuai tujuan pengembangannya. Sehingga rancang bangun tidak hanya berfokus pada pembuatan sistem, tetapi juga memastikan bahwa sistem yang dikembangkan mampu menyelesaikan permasalahan secara efektif dan sesuai dengan kebutuhan pengguna (Surahmat dkk., 2022).

Penerapan konsep rancang bangun sistem informasi dapat dilihat pada penelitian yang dilakukan oleh Prasetiawan dan Sumarlinda (2024) yang mengembangkan sistem informasi kepegawaian berbasis website melalui tahapan analisis kebutuhan, perancangan sistem, implementasi, dan pengujian. Hasil penelitian menunjukkan bahwa pendekatan tersebut mampu menghasilkan sistem yang sesuai dengan kebutuhan pengguna serta mendukung pengelolaan data secara lebih efektif (Prasetiawan dan Sumarlinda, 2024).

Pada penelitian ini, konsep rancang bangun digunakan dalam pengembangan sistem *dashboard* berbasis web yang mengintegrasikan teknologi *machine learning* untuk klasifikasi tipe pembelajaran siswa. Proses rancang bangun dilakukan secara sistematis mulai dari analisis kebutuhan, perancangan sistem menggunakan UML, implementasi menggunakan teknologi web, hingga tahap pengujian untuk memastikan sistem berjalan dengan baik.

2.3.2. Machine learning

Machine learning merupakan salah satu cabang dari kecerdasan buatan (*Artificial Intelligence*) yang memungkinkan sistem komputer untuk belajar dari data dan meningkatkan kinerjanya secara otomatis tanpa perlu diprogram secara eksplisit. *Machine learning* bekerja dengan memanfaatkan algoritma untuk menemukan pola dalam data, sehingga sistem dapat melakukan prediksi,

klasifikasi, maupun pengambilan keputusan berdasarkan data tersebut (Kharis dan Zili, 2022).

Beberapa tahun terakhir, *Machine learning* telah banyak diterapkan dalam berbagai bidang, termasuk pendidikan. *Machine learning* digunakan untuk menganalisis data aktivitas belajar siswa guna memahami pola perilaku belajar serta memprediksi performa akademik. Penelitian oleh Kharis dan Zili, (2022) menunjukkan bahwa penerapan *Machine learning* dalam analisis data pembelajaran mampu membantu dalam mengidentifikasi pola belajar siswa serta mendukung pengambilan keputusan berbasis data (Kharis dan Zili, 2022).

Penggunaan *Machine learning* dalam sistem ini bertujuan untuk mengklasifikasikan tipe pembelajaran siswa berdasarkan data aktivitas dan hasil belajar. Sehingga adanya model *Machine learning*, sistem tidak hanya menampilkan data secara deskriptif, tetapi juga mampu memberikan insight berupa hasil klasifikasi yang dapat membantu dalam memahami karakteristik belajar siswa secara lebih mendalam.

2.3.3. Data Mining

Data Mining merupakan proses pengolahan data yang menggabungkan teknik statistik, matematika, kecerdasan buatan (*Artificial Intelligence*), dan *machine learning* untuk menggali informasi yang saling berkaitan dan bermanfaat dari data dalam jumlah besar. Informasi yang dihasilkan melalui proses Data Mining dapat dimanfaatkan sebagai dasar dalam mendukung pengambilan keputusan (*Decision Support System*) maupun proses prediksi (Srirahayu dan Pribadie, 2023).

Data Mining digunakan untuk memperoleh pengetahuan atau informasi baru dengan cara mengidentifikasi pola-pola yang tersembunyi dari suatu basis data. Proses tersebut dilakukan melalui analisis data secara otomatis maupun semi-otomatis sehingga menghasilkan informasi berupa aturan, model, maupun pola yang dapat dimanfaatkan sesuai kebutuhan pengguna (Srirahayu dan Pribadie, 2023).

Data Mining memiliki beberapa teknik, di antaranya deskripsi (*description*), estimasi (*estimation), clustering*, asosiasi (*association*), prediksi (*prediction*), dan

klasifikasi (*classification*) (Srirahayu dan Pribadie, 2023). Salah satu teknik *data mining* yang banyak digunakan adalah *classification*. Teknik ini umumnya dibangun menggunakan pendekatan *machine learning*, yang secara umum terdiri atas tiga paradigma utama, yaitu *supervised learning*, *unsupervised learning* dan *reinforcement learning*. *Supervised learning* menggunakan data yang telah memiliki label untuk membangun model prediksi atau klasifikasi, sedangkan *unsupervised learning* menggunakan data tanpa label untuk menemukan pola atau kelompok yang tersembunyi di dalam data. Pada penelitian ini digunakan pendekatan *supervised learning* karena data latih telah memiliki label tipe pembelajaran siswa yang digunakan untuk membangun model klasifikasi (Nurhalizah dan Ardianto, 2024).

2.3.4. Klasifikasi

Klasifikasi merupakan salah satu teknik pada data mining yang bertujuan mengelompokkan data ke dalam kelas tertentu berdasarkan karakteristik yang dimiliki. Pada pendekatan *supervised learning*, proses klasifikasi dilakukan menggunakan data latih yang telah memiliki label sehingga model dapat mempelajari hubungan antara atribut dan kelas target. Model yang telah dibangun kemudian digunakan untuk menentukan kelas pada data baru berdasarkan pola yang telah dipelajari selama proses pelatihan (Géron, 2022).

Penerapan metode klasifikasi telah banyak digunakan dalam berbagai bidang, termasuk analisis data pendidikan. Nugroho dan Arif (2022) menunjukkan bahwa metode klasifikasi mampu mendukung proses analisis dan pengambilan keputusan melalui pengelompokan data berdasarkan karakteristik tertentu (Nugroho dan Arif, 2022).

Pada penelitian ini, metode klasifikasi digunakan untuk mengelompokkan siswa ke dalam empat tipe pembelajaran berdasarkan data aktivitas dan hasil belajar menggunakan algoritma *LightGBM*.

2.3.5. Unified Modelling Language (UML)

Unified Modelling Language (UML) adalah bahasa standar yang digunakan untuk memvisualisasikan, merancang, dan mendokumentasikan sistem berbasis objek. UML memberikan gambaran sistem secara grafis sehingga memudahkan

pengembang dalam memahami struktur, alur, dan interaksi antar komponen sistem sebelum implementasi. UML membantu membuat blueprint sistem, termasuk proses bisnis, struktur data, relasi antar kelas, serta interaksi antar modul dalam suatu aplikasi (Wayahdi dan Ruziq, 2023). UML terdiri dari sekelompok diagram atau bagan sistem. Diagram atau bagan tersebut menggambarkan permasalahan dan solusinya. Terdapat Sembilan diagram pada UML, namun dalam penelitian ini hanya akan digambarkan tiga diagram sebagai berikut :

a. Use case Diagram

Use case menggambarkan interaksi antara pengguna dan sistem, serta menunjukkan jenis-jenis interaksi yang terjadi dalam suatu program. *Use case* adalah rangkaian interaksi yang saling terkait antara sistem dan aktor. Oleh karena itu, *Use case* biasanya direpresentasikan dalam bentuk diagram untuk memperlihatkan perilaku (*behaviour*) sistem informasi yang akan dikembangkan.

Selain itu, *Use case* Diagram juga berfungsi sebagai alat komunikasi antara analis sistem, developer, dan stakeholder, karena diagram ini menggambarkan kebutuhan fungsional secara sederhana namun informatif. Dengan adanya *Use case*, kebutuhan pengguna dapat diidentifikasi lebih awal sehingga meminimalkan kesalahan dalam proses pengembangan sistem.

Use case Diagram biasanya terdiri dari beberapa elemen utama seperti aktor, *use case*, batas sistem (*system boundary*), serta hubungan antar elemen seperti *association*, *include*, dan *extend*. Aktor merepresentasikan pengguna atau sistem lain yang berinteraksi, sedangkan *use case* menggambarkan proses atau layanan yang disediakan sistem. Hubungan *include* digunakan untuk menggambarkan proses yang selalu disertakan, sementara *extend* menunjukkan proses tambahan yang bersifat opsional.

Diagram ini juga membantu dalam memetakan alur aktivitas secara keseluruhan sebelum masuk ke tahap perancangan yang lebih detail seperti *Activity diagram* atau Sequence Diagram. Dengan demikian, *Use case* Diagram menjadi dasar yang penting dalam proses analisis dan perancangan sistem karena mampu memberikan gambaran umum dari fungsi-fungsi yang akan dikembangkan

(Sandfreni dkk., 2021). Simbol *use case* bisa dilihat pada Tabel 2.1 (Sandfreni dkk., 2021).

Tabel 2. 1 Simbol *Use case*

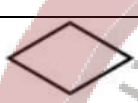
Simbol	Keteramngan
	Aktor: Mewakili peran orang, sistem yang lain, atau alat Ketika berkomunikasi
	*Use case*: Abstraksi dan interaksi antarasistem dan aktor.
	Association: Abstraksi dari penghubungantara aktor dengan *use case*.
	Generalisasi: Menunjukkan spesialisasiaktor untuk dapat berpartisipasi dengan usecase.
	Menunjukkan bahwa suatu *use case* seluruhnya merupakan fungsionalitas dari *use case* lainnya.
	Menunjukkan bahwa suatu *use case* seluruhnya merupakan fungsionalitas dari *use case* lainnya jika suatu kondisi terpenuhi.

b. Activity diagram

Activity diagram merupakan salah satu diagram perilaku (behavioral diagram) dalam UML yang digunakan untuk memodelkan alur aktivitas atau proses bisnis dalam sistem dengan menampilkan langkah-langkah kerja yang dilakukan baik oleh sistem maupun pengguna, lengkap dengan kondisi percabangan, pengulangan, serta aliran kontrol dari satu aktivitas ke aktivitas lainnya. Menurut Siewe dan Ngounou, (2025) dalam *jurnal On the Execution and Runtime Verification of UML Activity diagram*s, diagram ini berperan penting dalam memahami dinamika sistem dan dapat diberi semantik formal agar dapat diverifikasi secara runtime guna memastikan sistem berjalan sesuai dengan model

yang dirancang. Selain itu, *Activity diagram* juga membantu pengembang menganalisis efisiensi proses, mendeteksi potensi *bottleneck,* serta memberikan visualisasi yang jelas terhadap alur kerja sistem sebelum tahap implementasi dilakukan (Siewe dan Ngounou, 2025). Simbol Activity bisa dilihat pada Tabel 2.2

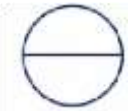
Tabel 2. 2 Simbol activity diagram

Simbol	Nama	Keterangan
	Status awal	Sebuah Diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem, aktivitasi biasanya diawali dengan kata kerja
	Percabangan/ *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.

c. Squence diagram

Menurut Sopriani dan Purwanto (2023) dalam jurnal Perancangan Sistem Informasi Persediaan Barang Berbasis Web pada PT. XYZ (Department IT Infrastructure), Sequence Diagram merupakan alat yang sangat populer dalam pengembangan sistem informasi berorientasi objek untuk menampilkan interaksi antar objek dalam suatu sistem secara berurutan berdasarkan waktu. Diagram ini membantu pengembang memahami alur komunikasi antar objek dan memastikan proses berjalan sesuai dengan logika sistem yang dirancang (Sopriani dan Purwanto, 2023). Simbol *Sequence Diagram* bisa dilihat pada Tabel 2.3 (Sopriani dan Purwanto, 2023).

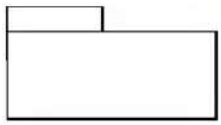
Tabel 2. 3 *Squence diagram*

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

d. Componen Diagram

Component Diagram digunakan untuk menggambarkan struktur dan hubungan antar komponen dalam sistem perangkat lunak. Diagram ini menunjukkan bagaimana setiap komponen saling berinteraksi melalui antarmuka dan *dependensi*, sehingga mempermudah proses pemodelan arsitektur sistem secara modular dan terstruktur (Sari dkk., 2022). Simbol *Component Diagram* bisa dilihat pada Tabel 2.4 (Sari dkk., 2022).

Tabel 2. 4 *Component Diagram*

Simbol	Nama	Keterangan
	*Package*	*Package* merupakan sebuah bungkusan dari satu atau lebih component
	*Component*	Bagian aplikasi berbentuk modul yang menjalankan fungsi tertentu
	dependency	Arah panah mengarah pada Componen yang dipakai
	*Interface*	Layanan yang disediakan komponen

e. Deployment Diagram

Deployment Diagram digunakan untuk menggambarkan bagaimana komponen perangkat lunak diimplementasikan pada lingkungan fisik seperti server, *client*, dan perangkat jaringan lainnya. Diagram ini menunjukkan hubungan antar node serta distribusi komponen sistem, sehingga membantu pengembang memahami arsitektur penyebaran sistem secara keseluruhan. Melalui diagram ini, pemetaan antara perangkat lunak dan perangkat keras menjadi lebih jelas, termasuk spesifikasi protokol komunikasi yang digunakan antar perangkat. Untuk memahami representasi visual yang digunakan dalam diagram tersebut (Zahra dan Priyadi, 2023). simbol *Deployment Diagram* dapat dilihat pada Tabel 2.5 (Zahra & Priyadi, 2023).

Tabel 2. 5 Deployment Diagram

Simbol	Nama	Keterangan
	*Package*	Package merupakan sebuah bungkusan dari satu atau lebih.
	*Node*	Biasanya mencakup pada perangkat keras (hardware), perangkat lunak yang tidak dibuat sendiri(software), jika didalam node disertakan komponen untuk mengkonsistenkan rancangan maka komponen yang telah didefinisikan sebelumnya pada diagram komponen.
	*Depedency*	Hubungan ketergantungan antar node, arah panah mengarah pada node yang dipakai.
	*Association*	Relasi antar *node*

2.3.6. Learning Analytics

Learning Analytics merupakan suatu pendekatan dalam bidang pendidikan yang memanfaatkan data untuk memahami, menganalisis, dan meningkatkan proses pembelajaran. *Learning Analytics* berfokus pada pengumpulan, pengolahan, serta analisis data yang dihasilkan dari aktivitas belajar siswa, seperti interaksi dengan materi, penyelesaian tugas, serta hasil evaluasi pembelajaran (Adam et al., 2025).

Menurut penelitian oleh Adam et al (2025), *Learning Analytics* dapat digunakan untuk mengidentifikasi pola perilaku belajar siswa serta memberikan gambaran mengenai tingkat keterlibatan dan performa belajar. Memanfaatkan data tersebut, pendidik dapat memperoleh insight yang lebih mendalam terkait kondisi pembelajaran siswa sehingga dapat mengambil keputusan yang lebih tepat.

Implementasi *Learning Analytics* sering dikombinasikan dengan sistem berbasis digital seperti *Learning Management System (LMS),* yang mampu menghasilkan data aktivitas siswa secara kontinu. Data tersebut kemudian diolah untuk menghasilkan informasi yang dapat digunakan dalam proses monitoring dan evaluasi pembelajaran. Penelitian oleh (Kharis dan Zili, 2022) menunjukkan bahwa pemanfaatan data aktivitas siswa dalam *Learning Analytics* dapat membantu dalam memahami performa belajar secara lebih menyeluruh.

Salah satu bentuk implementasi *Learning Analytics* adalah melalui penggunaan *dashboard*, yang dikenal sebagai *Learning Analytics Dashboard* (LAD). *Dashboard* ini berfungsi untuk menyajikan data pembelajaran dalam bentuk visual yang mudah dipahami, seperti grafik, indikator performa, dan ringkasan statistik. Sehingga adanya *dashboard*, pengguna dapat dengan cepat mengidentifikasi pola dan tren pembelajaran siswa (Paulsen dan Lindsay, 2024).

Seiring dengan perkembangan teknologi, *Learning Analytics* juga mulai dikombinasikan dengan pendekatan *machine learning* untuk meningkatkan kemampuan analisis data. Integrasi ini memungkinkan sistem tidak hanya menampilkan data secara deskriptif, tetapi juga mampu memberikan hasil analisis yang lebih mendalam, seperti klasifikasi atau prediksi terhadap performa belajar siswa. Maka demikian, *Learning Analytics* menjadi salah satu komponen penting dalam pengembangan sistem pembelajaran berbasis data.

2.3.7. *Dashboard*

Dashboard merupakan suatu media visualisasi informasi yang digunakan untuk menyajikan data secara ringkas, terstruktur, dan mudah dipahami dalam satu tampilan antarmuka. *Dashboard* berfungsi untuk membantu pengguna dalam memantau kondisi, menganalisis data, serta mendukung proses pengambilan keputusan secara cepat dan berbasis data (Miftahur Rizqy dan Putri Silmina, 2025).

Konteks sistem informasi menunjukkan bahwa *dashboard* biasanya menampilkan data dalam bentuk visual seperti grafik, tabel, indikator kinerja, serta ringkasan statistik. Penyajian data dalam bentuk visual ini bertujuan untuk mempermudah pengguna dalam memahami informasi yang kompleks menjadi lebih sederhana dan intuitif. Penelitian oleh Tiaraningsih dkk (2025) menunjukkan

bahwa penggunaan *dashboard* dalam sistem berbasis web mampu meningkatkan efektivitas monitoring data serta membantu pengguna dalam mengidentifikasi pola dan tren secara lebih cepat (Tiaraningsih dkk., 2025).

Pada penelitian ini, *dashboard* digunakan sebagai media untuk menampilkan hasil klasifikasi tipe pembelajaran siswa yang dihasilkan oleh model *machine learning*. Informasi yang ditampilkan meliputi performa belajar siswa, aktivitas pembelajaran, serta hasil klasifikasi yang divisualisasikan dalam bentuk grafik dan indikator. Adanya *dashboard* ini, diharapkan pengguna dapat memahami kondisi belajar siswa secara lebih jelas serta mendukung proses monitoring dan pengambilan keputusan secara lebih efektif.

2.3.8. Website

Website adalah salah satu koleksi dokumen *HTML* milik pribadi atau perusahaan yang berisi informasi dan berada dalam *Web Server* dan dapat diartikan sebagai kumpulan dari halaman situs yang terangkum dalam sebuah domain atau subdomain dan berada didalam *World Wide Web (WWW*). *Website* merupakan kumpulan halaman yang saling terhubung yang dapat diakses melalui jaringan internet menggunakan protokol *HTTP* atau *HTTPS. Website* digunakan sebagai media penyampaian informasi, layanan, dan fungsi interaktif kepada pengguna melalui browser tanpa memerlukan pemasangan perangkat lunak tambahan pada perangkat pengguna. Struktur suatu *website* umumnya terdiri atas konten teks, gambar, multimedia, dan navigasi yang diatur sedemikian rupa untuk memudahkan akses informasi secara real-time (Miftahuljannah dan Suharso, 2023).

Secara fungsional, website berperan sebagai sebuah sistem informasi berbasis web yang menyimpan dan menyajikan data secara terstruktur sehingga dapat diakses dengan cepat dan efisien. Perkembangan teknologi internet mengakselerasi penggunaan *website* tidak hanya sebagai media penyampaian informasi statis tetapi juga sebagai aplikasi dinamis yang mampu mengolah data, menampilkan visualisasi, serta berinteraksi dengan sistem *backend* melalui *API* (Wahyuni dkk., 2025).

Website telah menjadi komponen utama dalam berbagai sistem informasi karena kemampuannya untuk mengintegrasikan data, memperluas jangkauan

layanan, serta mendukung pengalaman pengguna (*user experience*) yang lebih baik. *Website* dapat dikembangkan dengan berbagai teknologi, baik untuk keperluan profil organisasi, sistem informasi manajemen, maupun aplikasi layanan digital lainnya (Miftahuljannah dan Suharso, 2023).

2.3.9. HTTP Status Code

Pada proses komunikasi antara *client* dan *server* melalui *Application Programming Interface (API)*, setiap request yang dikirimkan akan menghasilkan *HTTP Status Code* sebagai indikator keberhasilan atau kegagalan proses. Status code merupakan kode numerik yang dikirimkan oleh *server* untuk memberikan informasi mengenai hasil pemrosesan request yang diterima. Adanya *status code*, pengembang dapat mengetahui apakah request berhasil diproses atau terjadi kesalahan pada sisi *client* maupun *server* (Fielding et al., 2022).

Beberapa status code yang digunakan dalam penelitian ini ditunjukkan pada tabel 2.6 (Fielding et al., 2022).

Tabel 2. 6 Status HTTP Code

Status Code	Keterangan
200 *ok*	*Request* berhasil diproses oleh server dan data atau informasi yang diminta berhasil dikirimkan kembali kepada *client.*
400 *Bad Request*	*Request* yang dikirimkan tidak valid, misalnya parameter tidak sesuai, format data salah, atau terdapat data yang wajib diisi tetapi tidak diberikan.
404 *Not Found*	*Resource* atau *endpoint* yang diminta tidak ditemukan pada *server,* misalnya data dengan ID tertentu tidak tersedia.
500 *Internal Server Eror*	Terjadi kesalahan pada sisi *server* sehingga *request* tidak dapat diproses dengan baik.

2.3.10. Brevo

Brevo merupakan layanan *cloud-based* yang menyediakan fitur *transactional email* melalui *API* maupun *Simple Mail Transfer Protocol (SMTP)*

sehingga memungkinkan aplikasi mengirimkan email secara otomatis berdasarkan suatu peristiwa (*event*) tertentu, seperti notifikasi sistem, verifikasi akun, maupun pemberitahuan lainnya. Pada penelitian ini, *Brevo* digunakan sebagai layanan *transactional email* untuk mengirimkan email intervensi kepada siswa yang diklasifikasikan sebagai *Low-Performance Learner*. Integrasi dilakukan menggunakan *REST API* sehingga proses pengiriman email dapat dilakukan secara otomatis setelah hasil klasifikasi diperoleh (Brevo, 2023).

2.3.11. Tech Stack Frontend Backend

Tech stack merupakan kombinasi dari berbagai teknologi dan alat yang digunakan dalam pengembangan perangkat lunak atau aplikasi. *Tech stack* yang digunakan dalam pembuatan tugas akhir yaitu *NEXT JS, PostgreSQL*, dan Supabase berikut penjabaran lebih detailnya :

1. *Framework Frontend & Backend : NEXT JS*

   NEXT.JS merupakan *framework React.js* yang mendukung rendering *sisi server (Server-Side Rendering / SSR*), generasi situs statis (*Static Site Generation / SSG)*, serta kemampuan pengoptimalan performa dan SEO yang lebih baik dibandingkan aplikasi *single-page-app* tradisional. Penggunaan *NEXT.JS* membantu membangun antarmuka *dashboard* web yang cepat, responsif, dan *SEO-friendly*. Pada penelitian terkait *NEXT.JS* yang tersedia pada jurnal teknis Repeater Journal, disebutkan bahwa penggunaan *NEXT.JS* dalam pengembangan aplikasi web modern membantu meningkatkan *performance, responsivitas*, serta pengalaman pengguna (*user experience*) (Maharani, 2025).

2. *Runtime Server : NodeJS*

   Node.js adalah runtime environment yang dibangun di atas mesin JavaScript V8 milik Google Chrome dan digunakan untuk menjalankan JavaScript di sisi *server*. Dengan menggunakan Node.js, pengembangan aplikasi web menggunakan bahasa JavaScript dapat dilakukan secara menyeluruh baik di sisi klien maupun server. *Node.js* memungkinkan pengembang menggabungkan berbagai komponen aplikasi ke dalam satu ekosistem pengembangan yang kuat. Ini dimungkinkan oleh sistem manajemen paket *Node Package Manager (npm*), yang menawarkan ribuan modul dan pustaka untuk

mempercepat proses pengembangan. Karena dapat dijalankan, mengeksekusi, dan menyediakan fungsionalitas inti bahasa *JavaScript* di luar peramban (*browser), Node.js* sangat cocok untuk membangun aplikasi berbasis web yang ringan, skalabel, dan dalam waktu nyata (Andrianto dan Suyatno, 2024).

3. *Database : PostgreSQL*

PostgreSQL merupakan salah satu sistem manajemen basis data relasional (*Relational Database Management System / RDBMS*) yang bersifat *open-source* dan banyak digunakan dalam pengembangan aplikasi modern. *PostgreSQL* memiliki kemampuan dalam menangani data yang kompleks, mendukung transaksi *ACID (Atomicity, Consistency, Isolation, Durability)*, serta memberikan performa yang baik dalam pengolahan data berskala besar (Salunke dan Ouda, 2024).

Selain itu, *PostgreSQL* juga menyediakan berbagai fitur lanjutan seperti indexing, replikasi, partisi data, serta dukungan tipe data JSON/JSONB yang memungkinkan penyimpanan data semi-terstruktur secara fleksibel tanpa mengurangi konsistensi data. Kemampuan ini menjadikan *PostgreSQL* tidak hanya unggul sebagai basis data relasional, tetapi juga mampu mendukung kebutuhan aplikasi modern yang membutuhkan fleksibilitas dalam pengelolaan data (Salunke dan Ouda, 2024).

4. *Restful Api*

REST (Representational State Transfer) merupakan gaya arsitektur dalam pengembangan layanan web yang menggunakan protokol HTTP untuk komunikasi antara *client* dan *server. Client* mengirimkan permintaan *(request*) ke *server, server* merespons dengan data yang biasanya disampaikan dalam format JSON atau XML. Implementasi dari arsitektur ini dikenal sebagai *Restful* API, yaitu antarmuka pemrograman aplikasi yang memungkinkan pertukaran data secara efisien antar sistem informasi melalui web. Penerapan *Restful* web service dalam sistem informasi manajemen sumber daya manusia (SIM SDM) memberikan kemudahan integrasi data antar unit atau departemen. Komunikasi berbasis HTTP dan format data terstandarisasi seperti JSON menjadikan data tidak lagi tersimpan terpisah di masing-masing unit, melainkan

terpusat dan mudah diakses oleh seluruh sistem yang terintegrasi. Penerapan *Restful* web service dibutuhkan untuk meningkatkan efisiensi dan konsistensi data dalam SIM SDM (Adinta dkk., 2024).

5. *Visual Studio Code*

Visual Studio Code adalah sebuah teks editor ringan dan handal yang dibuat oleh Microsoft untuk sistem operasi multiplatform yang artinya tersedia juga untuk *versi Linux, Mac, dan Window*s. Teks editor ini secara langsung mendukung bahasa pemrograman *JavaScript,Typescript, dan Node.js*, serta bahasa pemrograman lainnya dengan bantuan plugin yang dapat dipasang melalui marketplace Visual Studio Code seperti *C++, C#, Python, Go, Java*. Banyak sekali fitur-fitur yang disediakan oleh Visual Studio Code, diantaranya Intellisense, Git Integration, Debugging, dan fitur ekstensi yang menambah kemampuan teks editor. Fitur-fitur tersebut akan terus bertambah seiring dengan bertambahnya versi Visual Studio Code. Pembaruan versi Visual Studio Code ini juga dilakukan berkala setiap bulan, dan inilah yang membedakan Visual Studio Code dengan teks editor yang lain (Wilyanto dkk., 2023)

2.3.12. Gaussian Copula

Gaussian Copula merupakan metode statistik untuk membangkitkan data sintetis dengan mempertahankan struktur ketergantungan *(dependency*) antar variabel pada data asli. Metode ini memodelkan hubungan antar atribut menggunakan distribusi normal *multivariat* sehingga data sintetis yang dihasilkan memiliki karakteristik statistik yang menyerupai data asli. Oleh karena itu, *Gaussian Copula* banyak digunakan dalam pembangkitan data sintetis ketika jumlah data terbatas, sekaligus menjaga kualitas data yang digunakan dalam pelatihan model *machine learning* (Restrepo dkk., 2023).

2.3.13. SMOTE

Synthetic Minority Over-sampling Technique (SMOTE) merupakan metode oversampling yang digunakan untuk mengatasi permasalahan ketidakseimbangan kelas (*class imbalance*) pada dataset. Berbeda dengan metode *oversampling konvensional* yang hanya menggandakan data pada kelas *minoritas*, *SMOTE*

membentuk data sintetis baru berdasarkan interpolasi antara sampel kelas minoritas yang berdekatan. Pendekatan ini membantu mengurangi risiko *overfitting* serta meningkatkan kemampuan model dalam mengenali pola pada kelas minoritas (Nemade dkk., 2023).

Pada penelitian ini, *SMOTE* diterapkan setelah proses pembangkitan data sintetis menggunakan *Gaussian Copula* untuk menyeimbangkan jumlah data pada masing-masing kategori tipe pembelajaran siswa sebelum dilakukan proses pelatihan model *LightGBM*. Dengan demikian, model diharapkan memiliki kemampuan klasifikasi yang lebih baik pada seluruh kelas tanpa bias terhadap kelas mayoritas.

2.3.14. LightGBM

Light Gradient Boosting Machine (LightGBM) merupakan salah satu algoritma *machine learning* berbasis *ensemble learning* yang termasuk dalam metode *gradient boosting*. Algoritma ini bekerja dengan membangun model secara bertahap melalui kombinasi beberapa *decision tree* yang dilatih secara berurutan untuk meminimalkan kesalahan prediksi. *LightGBM* dirancang untuk meningkatkan efisiensi komputasi serta mampu menangani data berukuran besar dengan performa yang tinggi (Maulita dan Wahid, 2024).

Salah satu keunggulan utama *LightGBM* adalah penggunaan *metode leaf-wise tree growth*, yaitu teknik pertumbuhan pohon yang berfokus pada node dengan potensi penurunan error terbesar. Pendekatan ini memungkinkan model menghasilkan akurasi yang lebih tinggi dibandingkan *metode level-wise* pada algoritma boosting lainnya. Selain itu, *LightGBM* juga memiliki kecepatan pelatihan yang tinggi serta efisiensi dalam penggunaan memori.

Pada penelitian ini, *LightGBM* digunakan sebagai model klasifikasi untuk mengelompokkan siswa ke dalam tipe pembelajaran berdasarkan data aktivitas dan performa belajar. Model ini dipilih karena kemampuannya dalam menangani data tabular, efisiensi dalam proses pelatihan, serta tingkat akurasi yang tinggi dalam menghasilkan prediksi.

2.3.15. Preprocessing dan Evaluasi Model

Preprocessing data merupakan tahap awal dalam *machine learning* untuk menyiapkan data sebelum digunakan dalam pelatihan model. Tahap ini dapat mencakup pembersihan data, penanganan data kosong, pengubahan data kategorikal menjadi numerik, serta pemilihan fitur yang relevan agar data dapat diproses oleh algoritma. Kualitas data yang baik diperlukan karena proses *preprocessing* dan pemilihan fitur dapat memengaruhi performa model *machine learning* (Ahmed et al., 2024).

Evaluasi model klasifikasi digunakan untuk mengetahui kemampuan model dalam memprediksi data uji. Evaluasi dapat dilakukan menggunakan *confusion matrix* dan metrik seperti *accuracy, precision, recall*, dan *F1-score*. *Confusion matrix* merupakan tabel yang membandingkan hasil prediksi model dengan label aktual, sehingga dapat diketahui jumlah prediksi benar dan salah pada setiap kelas (Riehl et al., 2023).

Accuracy menunjukkan perbandingan jumlah prediksi benar terhadap seluruh data yang diuji. Pada klasifikasi *multiclass, accuracy* dapat dipahami sebagai rasio prediksi benar terhadap seluruh data uji. Rumus *accuracy* yang digunakan pada penelitian ini ditunjukkan menggunakan Persamaan (2.1) (Opitz, 2024) :

$$Accuracy = \frac{\sum_i m_{i,i}}{\sum_{(i,j)} m_{i,j}} \qquad (2.1)$$

$$Accuracy = \frac{Jumlah\ Prediksi\ Benar}{Jumlah\ Seluruh\ Data\ Uji}$$

Keterangan:

- $m_{i,i}$= elemen diagonal *confusion matrix* (prediksi benar)
- $\sum_i m_{i,i}$= jumlah seluruh prediksi benar
- $\sum_{(i,j)} m_{i,j}$= jumlah seluruh data pada confusion matrix

Rumus tersebut digunakan karena model pada penelitian ini melakukan klasifikasi ke dalam empat kelas tipe pembelajaran. Sehingga prediksi dianggap

benar apabila kelas rekomendasi yang diprediksi model sama dengan kelas aktual pada data uji (Opitz, 2024).

Precision digunakan untuk melihat ketepatan prediksi pada suatu kelas, *recall* digunakan untuk melihat kemampuan model menemukan data pada kelas tertentu, sedangkan *F1-score* digunakan untuk melihat keseimbangan antara *precision* dan *recall*. Pada penelitian ini, metrik tersebut digunakan untuk mengevaluasi model *LightGBM* dalam mengklasifikasikan tipe pembelajaran siswa (Opitz, 2024).

2.3.16. FastAPI

FastAPI adalah *framework* web modern berbasis Python yang dirancang khusus untuk membangun RESTful API berkinerja tinggi dengan dukungan *asynchronous programming* dan dokumentasi otomatis. FastAPI menggabungkan penggunaan Python type hint untuk melakukan validasi input secara otomatis, pengelolaan rute (*routing*), dan pemrosesan *request* secara efisien sehingga sangat cocok untuk aplikasi backend seperti layanan rekomendasi dan *microservices*. *FastAPI* dibangun di atas *ASGI (Asynchronous Server Gateway Interface)* yang memungkinkan penanganan permintaan secara *non-blocking* sehingga memberikan performa lebih baik pada beban tinggi dibandingkan beberapa kerangka kerja tradisional berbasis *WSGI (Bednarz dan Miłosz, 2025).*

2.3.17. Pengujian *Black Box*

Black Box Testing merupakan salah satu metode pengujian perangkat lunak yang dilakukan tanpa memperhatikan struktur internal atau kode program. Pengujian ini berfokus pada pemeriksaan fungsi sistem berdasarkan masukan (*input*) dan keluaran (*output*) untuk memastikan setiap fitur bekerja sesuai dengan kebutuhan dan spesifikasi yang telah ditetapkan (Sahyudi dan Voutama, 2025).

Tujuan utama *Black Box Testing* adalah memverifikasi bahwa setiap fungsi aplikasi mampu menghasilkan keluaran yang benar berdasarkan skenario pengujian yang telah dirancang. Pada metode ini, penguji tidak perlu mengetahui proses internal sistem, tetapi hanya memastikan bahwa setiap fitur telah memenuhi kebutuhan pengguna. Oleh karena itu, *Black Box Testing* banyak digunakan untuk

menguji antarmuka pengguna (*user interface*), validasi data, proses bisnis, serta integrasi antar fitur dalam suatu aplikasi (Sahyudi dan Voutama, 2025)

2.3.18. Pengujian *Api* Menggunakan Postman

Pengujian *Application Programming Interface (API)* dilakukan untuk memastikan proses komunikasi data antara *frontend, backend*, basis data, dan layanan pendukung sistem dapat berjalan dengan baik. Salah satu *tools* yang banyak digunakan untuk melakukan pengujian *API* adalah *Postman*. *Postman* merupakan aplikasi yang digunakan untuk mengirimkan request *HTTP,* seperti *GET, POST, PUT,* dan *DELETE*, kemudian menganalisis *response* yang dihasilkan oleh setiap *endpoint API* (Andrianto dan Suyatno, 2024).

Melalui *Postman*, pengembang dapat memverifikasi *status code*, struktur data yang dikembalikan, waktu respons, serta memastikan setiap *endpoint* bekerja sesuai dengan fungsinya. Pengujian ini juga membantu proses validasi komunikasi antar komponen sistem dan mempermudah identifikasi kesalahan (*debugging*) selama proses pengembangan (Andrianto dan Suyatno, 2024).