BAB II

KAJIAN PUSTAKA

A. Kajian Teori

1. Sistem

Sistem dapat diartikan sebagai suatu kesatuan yang tersusun atas berbagai elemen yang saling berhubungan dan bekerja sama secara terintegrasi untuk mencapai tujuan yang telah ditetapkan. Setiap sistem memiliki komponen, batasan, serta mekanisme kerja yang terstruktur agar dapat menghasilkan keluaran sesuai sasaran yang ditetapkan. Dalam konteks teknologi, sistem digunakan untuk menyederhanakan proses yang kompleks menjadi alur kerja yang lebih teratur dan mudah dikendalikan.

Menurut Fajar & Chotijah (2022) sistem didefinisikan sebagai sekumpulan komponen yang saling berkaitan serta beroperasi secara terpadu dalam suatu mekanisme untuk mewujudkan tujuan atau sasaran yang telah ditentukan. Sejalan dengan itu, Suli (2023) menjelaskan bahwa sistem merupakan kerangka kerja yang mengatur interaksi antar bagian menjadi suatu kesatuan dengan suatu tujuan tertentu. Mengutip dari situs Universitas Dian Nuswantoro. Sistem memiliki beberapa karakteristik.

a. Komponen sistem (*component*)

Sebuah sistem tersusun atas berbagai komponen yang saling berhubungan dan berinteraksi satu sama lain. Setiap komponen bekerja

secara terpadu sehingga membentuk suatu kesatuan yang mampu menjalankan fungsi sistem secara menyeluruh.

b. Subsistem (*subsystem*)

Komponen dalam suatu sistem dapat berupa beberapa subsistem yang memiliki fungsi masing-masing. Setiap subsistem mempunyai karakteristik yang mendukung jalannya sistem utama serta memberikan pengaruh terhadap kinerja sistem secara keseluruhan. Selain itu, suatu sistem juga dapat menjadi bagian dari sistem yang lebih besar, yang dikenal sebagai *supra system*.

c. Batasan sistem (*boundary*)

Batas sistem merupakan ruang lingkup yang membedakan suatu sistem dengan sistem lainnya maupun dengan lingkungan di sekitarnya. Keberadaan batas tersebut menjadikan sistem dapat dipandang sebagai satu kesatuan yang utuh dan memiliki ruang kerja yang jelas.

d. Lingkungan di luar sistem (*environment*)

Lingkungan luar sistem mencakup seluruh kondisi atau faktor yang berada di luar batas sistem, tetapi memiliki pengaruh terhadap proses maupun kinerja sistem selama beroperasi.

e. Penghubung sistem (*interface*)

Penghubung sistem berfungsi sebagai media yang menghubungkan satu subsistem dengan subsistem lainnya. Melalui penghubung tersebut, aliran sumber daya maupun informasi dapat

berlangsung sehingga keluaran dari suatu subsistem dapat dimanfaatkan sebagai masukan bagi subsistem yang lain.

f. Masukan sistem (*input*)

Masukan sistem terdiri atas *maintenance input* dan *signal input*. *Maintenance input* merupakan masukan yang berfungsi untuk mendukung operasional sistem, sedangkan *signal input* adalah data yang diproses menjadi informasi. Pada sistem komputer, program berperan sebagai *maintenance input*, sedangkan data yang diolah merupakan *signal input*.

g. Keluaran sistem (*output*)

Keluaran sistem merupakan hasil dari proses pengolahan masukan yang telah diklasifikasikan sesuai kebutuhan. Informasi yang dihasilkan tersebut selanjutnya dapat dimanfaatkan sebagai masukan bagi subsistem lainnya.

h. Sasaran sistem (*object*)

Setiap sistem memiliki tujuan yang jelas sebagai acuan dalam pelaksanaan prosesnya. Suatu sistem dapat dinyatakan berhasil apabila mampu mencapai sasaran atau tujuan yang telah ditetapkan sebelumnya.

i. Pengolah sistem (*process*)

Proses merupakan mekanisme yang dilakukan sistem untuk mengolah masukan menjadi keluaran. Melalui tahapan ini, data yang diterima diproses sehingga menghasilkan informasi atau keluaran yang sesuai dengan tujuan sistem.

2. Analisis

Analisis adalah proses menguraikan suatu Suatu permasalahan diuraikan ke dalam bagian-bagian yang lebih sederhana agar hubungan antar setiap komponen dapat dipahami dengan lebih jelas. Dalam bidang sistem informasi, analisis digunakan untuk mengidentifikasi kebutuhan sistem serta menentukan solusi yang tepat terhadap suatu masalah.

Menurut Ulfah et al. (2023), Analisis adalah cara meneliti secara langsung dan menyelidiki suatu masalah atau hal yang terjadi dalam kehidupan nyata. Sementara itu, Sofwatillah et al. (2024) menjelaskan bahwa analisis bertujuan untuk menghasilkan informasi yang dapat dimanfaatkan sebagai bahan pertimbangan dalam proses pengambilan keputusan.

3. Keuangan (Pengeluaran)

Keuangan, khususnya pengeluaran, merupakan arus keluar dana yang digunakan untuk memenuhi kebutuhan operasional suatu usaha. Pengelolaan pengeluaran yang baik sangat penting agar kondisi finansial tetap stabil dan terkontrol.

Menurut Andi et al. (2026), Pengelolaan finansial individu masih mengalami kendala pada pencatatan secara manual maupun dengan aplikasi tradisional, seperti kurangnya konsistensi dan bias dalam kategori pengeluaran. Sejalan dengan itu, Syah & Satato (2024) menyatakan bahwa pencatatan pengeluaran yang terstruktur dapat membantu dalam proses evaluasi dan perencanaan keuangan.

Dalam penelitian ini, terdapat beberapa rumus perhitungan keuangan yang digunakan sebagai dasar verifikasi manual terhadap hasil prediksi sistem. Rumus-rumus tersebut digunakan untuk menghitung total pengeluaran aktual suatu periode dan rata-rata pengeluaran bulanan.

a) Rumus total pengeluaran aktual

pada rumus Total Pengeluaran Aktual terdiri dari variabel x_i yang menunjukkan nilai pengeluaran pada transaksi ke-i, sedangkan n menunjukkan jumlah seluruh transaksi pengeluaran yang terjadi dalam periode pengamatan. Simbol Σ (*sigma*) digunakan untuk menyatakan proses penjumlahan seluruh nilai pengeluaran yang tercatat. Rumus total pengeluaran aktual dapat dilihat pada persamaan 2.1.

$$TotalPengeluaran = \sum_{i=1}^{n} x_i \quad (2.1)$$

b) Rumus rata-rata pengeluaran bulanan

Pada rumus rata-rata pengeluaran bulanan, Σ Pengeluaran Bulanan menunjukkan total seluruh pengeluaran yang terjadi selama periode pengamatan, sedangkan n merupakan jumlah bulan yang digunakan sebagai sampel penelitian. Hasil perhitungan rata-rata ini menggambarkan kondisi pengeluaran normal bengkel dalam satu bulan sehingga dapat digunakan sebagai indikator untuk mengetahui kecenderungan pengeluaran pada periode berikutnya. Rumus rata-rata pengeluaran bulanan dapat dilihat pada persamaan 2.2.

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \quad (2.2)$$

c) Rumus *Eror Absolute*

Pada rumus Error Absolut, variabel y_{aktual} menyatakan nilai pengeluaran aktual yang benar-benar terjadi pada suatu periode, sedangkan $\hat{y}_{prediksi}$ merupakan nilai pengeluaran yang diprediksi oleh sistem menggunakan model *LSTM*. Simbol nilai mutlak $|\ |$ digunakan untuk memastikan bahwa hasil selisih selalu bernilai positif tanpa memperhatikan arah kesalahan. Perhitungan *error absolut* bertujuan untuk mengetahui besarnya penyimpangan antara hasil prediksi dan data aktual. Rumus Error Absolute dapat dilihat pada persamaan 2.3.

$$|y_{aktual} - \hat{y}_{prediksi}| \quad (2.3)$$

d) Rumus *Mean Absolute Percentage Error (MAPE)*

Pada rumus *Mean Absolute Percentage Error (MAPE)*, variabel n menunjukkan jumlah data yang digunakan dalam proses evaluasi, sedangkan y_{aktual} merupakan nilai pengeluaran aktual dan $\hat{y}_{prediksi}$ merupakan nilai pengeluaran hasil prediksi sistem. *MAPE* dihitung dengan mengukur besarnya selisih absolut antara nilai aktual dan nilai hasil prediksi dengan menjadikan nilai aktual sebagai acuan, kemudian hasil perhitungannya dinyatakan dalam bentuk persentase. Nilai *MAPE* digunakan untuk mengukur tingkat kesalahan prediksi secara keseluruhan. Semakin kecil nilai *MAPE* yang diperoleh, maka semakin tinggi tingkat akurasi model dalam melakukan prediksi. Dalam penelitian ini, *MAPE* digunakan sebagai metode evaluasi utama

untuk mengukur kinerja model *LSTM* dalam memprediksi pengeluaran Bengkel Mobil Ilham Putra. Rumus *Mean Error Percentage Error* dapat dililhat pada persamaan 2.4.

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \quad (2.4)$$

4. Sistem Pendukung

Dalam konteks penelitian ini, sistem pendukung digunakan untuk membantu proses analisis keuangan, khususnya dalam mengelola data pengeluaran. Sistem yang dikembangkan tidak hanya dimanfaatkan untuk melakukan pencatatan data, tetapi juga berperan dalam mengolah data tersebut menjadi informasi yang bernilai dan dapat dimanfaatkan untuk berbagai kebutuhan, seperti tren pengeluaran dan prediksi biaya di masa depan. Dengan demikian, sistem pendukung memiliki peran penting dalam membantu pengambilan keputusan yang lebih berbasis data.

5. *Android*

Android adalah sistem operasi yang dibangun di atas kernel *Linux* dan dirancang untuk digunakan pada berbagai perangkat bergerak, seperti *smartphone* maupun tablet. Sistem ini bersifat *open source* sehingga memungkinkan pengembang untuk menciptakan berbagai aplikasi sesuai kebutuhan pengguna.

Menurut Trivaika & Senubekti (2022), *Android* adalah platform yang menyediakan lingkungan pengembangan aplikasi yang fleksibel dan mudah digunakan. Sementara itu, Budiarto et al. (2025) menjelaskan bahwa *Android* merupakan salah satu sistem operasi yang paling banyak digunakan

karena menawarkan kemudahan dalam proses akses dan penggunaannya, dukungan komunitas yang luas, serta kompatibilitas dengan berbagai perangkat. Dalam penelitian ini, *android* digunakan sebagai media untuk membangun aplikasi pencatatan dan analisis keuangan karena sifatnya yang praktis, mudah diakses, dan mendukung penggunaan secara *real-time*.

6. Time series

Menurut Rini & Ananda (2022) pada penelitiannya, metode *time series* digunakan untuk membandingkan peramalan permintaan untuk menentukan mana metode yang peramalan yang terbaik. Sejalan dengan itu, Azahra et al., (2022) menjelaskan bahwa metode *time series* digunakan untuk melakukan peramalan (*forecasting*) berdasarkan data historis yang telah dikumpulkan sebelumnya.

Dalam penelitian ini, metode *time series* digunakan untuk menganalisis data pengeluaran yang terjadi secara periodik, sehingga dapat diketahui pola pengeluaran serta dilakukan prediksi terhadap pengeluaran di masa mendatang. Dengan adanya analisis ini, pengguna dapat mengambil keputusan yang lebih tepat dalam pengelolaan keuangan. Tabel 2.1 menunjukkan jenis-jenis *time series* yang umum digunakan dalam analisis data.

Tabel 2. 1 Jenis-Jenis Data *Time Series*

No	Jenis-Jenis *Time Series*	Deskripsi
1	Stasioner (*Stationary*)	Data *time series* yang memiliki rata-rata dan variansi yang relatif konstan dari waktu ke waktu, serta tidak menunjukkan adanya tren maupun pola musiman yang

		signifikan. Data jenis ini cenderung berfluktuasi di sekitar nilai rata-rata tertentu secara konsisten. Contoh: data pengeluaran operasional harian yang relatif stabil.
2	Non-Stasioner (*Non-Stationary*)	Data time series yang menunjukkan perubahan nilai rata-rata, variansi, atau keduanya seiring berjalannya waktu, sehingga tidak memiliki pola yang tetap. Data ini memerlukan transformasi tertentu sebelum dapat dianalisis lebih lanjut. Contoh: data pendapatan usaha yang terus meningkat setiap tahun.
3	Musiman (*Seasonal*)	Data *time series* yang menunjukkan adanya pola yang berulang secara konsisten dalam interval waktu tertentu, misalnya pada periode harian, mingguan, bulanan, maupun tahunan. Pola musiman ini terjadi akibat pengaruh faktor-faktor yang bersifat siklikal dan dapat diprediksi. Contoh: pengeluaran bengkel yang meningkat setiap akhir bulan.
4	Tren (*Trending*)	Data time series yang memperlihatkan kecenderungan naik atau turun secara konsisten dalam jangka panjang. Tren dapat bersifat linear maupun non-linear tergantung pada karakteristik data. Contoh: tren kenaikan pengeluaran bahan baku sparepart dari bulan ke bulan.
5	Siklikal (*Cyclical*)	Data time series yang memperlihatkan pola fluktuasi naik dan turun dalam periode yang lebih panjang dari satu tahun dan tidak memiliki frekuensi yang tetap. Pola ini sering dikaitkan dengan siklus ekonomi atau bisnis. Contoh: siklus pengeluaran investasi alat besar bengkel tiap beberapa tahun sekali.
6	Tidak Beraturan (*Irregular/Random*)	Data time series yang pergerakannya bersifat acak dan tidak dapat diprediksi, disebabkan oleh faktor-faktor yang bersifat tidak terduga seperti kejadian luar biasa, bencana, atau perubahan kebijakan

	mendadak. Contoh: lonjakan pengeluaran tak terduga akibat kerusakan peralatan bengkel.

B. Perancangan Sistem

1. *Android Studio*

Menurut Erich et al. (2023), *Android Studio* merupakan perangkat lunak resmi yang berfungsi sebagai lingkungan pengembangan untuk membangun dan mengembangkan aplikasi berbasis *Android.* yang dilengkapi dengan berbagai fitur seperti *editor kode*, *emulator*, dan *debugging tools*. Sejalan dengan itu, Zalukhu et al., (2023) menyatakan bahwa *Android Studio* mempermudah proses pengembangan aplikasi karena menyediakan berbagai fasilitas yang terintegrasi dalam satu platform.

Android Studio digunakan dalam penelitian ini untuk merancang, membangun, dan menguji aplikasi sistem pendukung analisis keuangan. Dengan menggunakan *Android Studio*, proses pengembangan menjadi lebih terstruktur karena adanya fitur visualisasi *layout*, pengelolaan *resource*, serta integrasi dengan *database*. Hal ini memungkinkan pengembang untuk menciptakan aplikasi yang lebih efisien dan sesuai dengan kebutuhan pengguna.

2. *Database*

Menurut Afiifah et al., (2022), *Database* merupakan sekumpulan data yang disusun dan disimpan secara terstruktur sehingga memudahkan proses akses, pengelolaan, serta pembaruan data sesuai dengan kebutuhan.

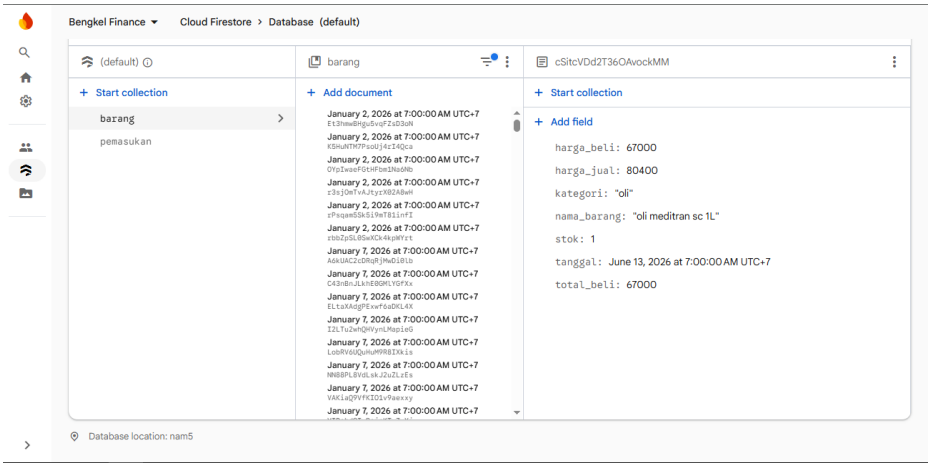
Sementara itu, Hasibuan & Triase (2022) menjelaskan bahwa *Database* berperan sebagai media penyimpanan utama dalam sistem informasi yang memungkinkan proses pengelolaan dan pengolahan data dilakukan secara lebih efektif dan efisien.

Dalam sistem yang dibangun, *database* digunakan untuk menyimpan seluruh data transaksi pengeluaran yang diinput oleh pengguna. Data tersebut kemudian diolah untuk menghasilkan informasi berupa laporan dan analisis. Penggunaan *database* memungkinkan data tersimpan dengan aman, terstruktur, serta dapat digunakan kembali untuk keperluan analisis di masa mendatang.

3. Firebase

Menurut Firdaus et al., (2025), *Firebase* merupakan platform *backend* yang menyediakan berbagai layanan seperti *database* realtime, autentikasi pengguna, serta penyimpanan data yang terintegrasi dalam satu sistem. Sejalan dengan itu, Setyawan (2024) menjelaskan bahwa *Firebase* memungkinkan sinkronisasi data secara langsung (*real-time*) antar perangkat, sehingga sangat mendukung pengembangan aplikasi berbasis *mobile* yang membutuhkan kecepatan dan efisiensi dalam pengelolaan data. Dalam penelitian ini, *Firebase* digunakan sebagai *database* untuk menyimpan data transaksi pengeluaran yang diinput melalui aplikasi *android*. Data yang disimpan meliputi informasi tanggal transaksi, jumlah pengeluaran, kategori, serta keterangan tambahan lainnya. Penggunaan *Firebase* memungkinkan data tersimpan secara online dan dapat diakses

kapan saja serta dari berbagai perangkat. Selain itu, fitur *real-time database* memungkinkan data yang dimasukkan langsung diperbarui tanpa perlu melakukan *refresh* secara manual, sehingga sangat mendukung proses analisis data keuangan secara cepat dan akurat menggunakan metode *time series.* Dengan demikian, *Firebase* menjadi alternatif yang efektif dalam meningkatkan tingkat fleksibilitas, keamanan, dan ketersediaan data dalam sistem yang dikembangkan. Dibawah merupakan gambar halaman dari *firebase firestore* yang digunakan pada penelitian ini, dapat dilihat pada gambar 2.1.

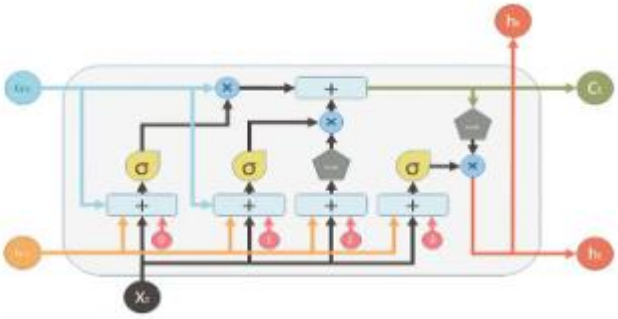


Gambar 2. 1 Firebase Console

4. *Long Short-Term Memory (LSTM)*

Menurut Masipupu et al., (2025), model *Long Short-Term Memory (LSTM)* dipilih dalam penelitian prediksi laju inflasi karena memiliki kemampuan yang baik dalam menangkap serta mempertahankan informasi atau pola jangka panjang pada data deret waktu. Temuan dari penelitian ini menunjukkan bahwa model *Long Short-Term Memory (LSTM)* dapat menghasilkan hasil prediksi yang efektif berdasarkan nilai evaluasi yang

diperoleh, Studi lain oleh Susilo et al., (2025) menyatakan bahwa penggunaan *Long Short-Term Memory (LSTM)* dalam memperkirakan kebutuhan persediaan barang mampu menghasilkan tingkat akurasi yang lebih baik apabila dibandingkan dengan metode-metode tradisional, karena bisa memanfaatkan pola data historis dengan lebih efisien. Arsitektur *LSTM* terdiri dari satu sel memori (*memory cell*) yang dikendalikan oleh tiga *gate*, yaitu *forget gate*, *input gate*, dan *output gate*, yang masing-masing bekerja sama untuk mengatur aliran informasi di dalam jaringan. Gambar 2.2 menunjukkan arsitektur dari algoritma *LSTM.*



Gambar 2. 2 Arsitektur *LSTM*

Sumber: (Putri et al., 2025)

Secara matematis, proses komputasi yang terjadi di dalam setiap sel *LSTM* pada setiap *time step* ke-t dapat dijelaskan melalui serangkaian persamaan berikut. Setiap persamaan merepresentasikan fungsi dari masing-masing komponen *gate* yang bekerja secara terintegrasi untuk menghasilkan keluaran yang tepat.

a) *Forget Gate*

Forget gate bertugas menentukan seberapa besar proporsi informasi dari sel memori sebelumnya (*cell state*) yang harus

dipertahankan atau dilupakan. Nilai keluaran *forget gate* berupa vektor bernilai antara 0 hingga 1, di mana nilai mendekati 0 berarti informasi dilupakan sepenuhnya dan nilai mendekati 1 berarti informasi dipertahankan sepenuhnya. Persamaan 2.5 menunjukkan rumus *Forget Gate*.

$$f_t = \sigma(Wf \cdot [h_{t-1}, x_t] + bf) \quad (2.5)$$

Nilai f_t merupakan keluaran *forget gate* pada *time step* ke-t yang bernilai antara 0 hingga 1, σ adalah fungsi aktivasi *sigmoid*, Wf merupakan matriks bobot *forget gate*, h_{t-1} adalah keluaran *hidden state* pada *time step* sebelumnya, x_t merupakan vektor input pada *time step* ke-t, dan bf adalah vektor bias *forget gate*.

b) *Input Gate*

Input gate pada arsitektur *LSTM* tersusun atas dua komponen yang bekerja secara bersamaan. Komponen pertama berupa lapisan *sigmoid* yang bertugas menentukan informasi mana yang perlu diperbarui, sedangkan komponen kedua adalah lapisan tanh yang menghasilkan vektor kandidat berisi nilai-nilai baru untuk ditambahkan ke dalam *cell state* atau sel memori. Persamaan 2.6 menunjukkan rumus *Input Gate*.

$$i_t = \sigma(Wi \cdot [h_{t-1}, x_t] + bi) \quad (2.6)$$

Nilai i_t merupakan keluaran *input gate* pada *time step* ke-t, σ adalah fungsi aktivasi *sigmoid*, Wi merupakan matriks bobot *input gate*, h_{t-1} adalah keluaran *hidden state* pada *time step* sebelumnya, x_t

merupakan vektor *input* pada *time step* ke-t, dan bi adalah vektor bias *input gate*. Persamaan 2.7 menunjukkan rumus kandidat *Cell State*.

$$C_t = tanh(Wc \cdot [h_{t-1}, x_t] + bc) \quad (2.7)$$

Nilai C_t merupakan vektor kandidat *cell state* baru yang akan ditambahkan ke dalam memori pada *time step* ke-t, $tanh$ adalah fungsi aktivasi tangen hiperbolik yang menghasilkan nilai antara -1 dan 1, Wc merupakan matriks bobot kandidat *cell state*, h_{t-1} adalah keluaran *hidden state* pada *time step* sebelumnya, x_t merupakan vektor input pada *time step* ke-t, dan bc adalah vektor bias kandidat *cell state*.

c) Pembaruan *Cell State*

Cell state merupakan komponen inti dari arsitektur LSTM yang berfungsi sebagai memori jangka panjang. Pembaruan *cell state* dilakukan dengan menggabungkan hasil *forget gate* (yang menentukan informasi lama yang dipertahankan) dan hasil *input gate* (yang menambahkan informasi baru ke dalam memori). Persamaan 2.8 menunjukkan rumus *Cell State*.

$$C_t = f_t \odot C_{t-1} + i_t \odot C_t, \quad (2.8)$$

Nilai C_t merupakan nilai *cell state* yang diperbarui pada *time step* ke-t, f_t adalah keluaran *forget gate* yang menentukan proporsi informasi lama yang dipertahankan, C_{t-1} adalah nilai *cell state* pada *time step* sebelumnya, i_t adalah keluaran *input gate* yang menentukan proporsi informasi baru yang ditambahkan, C_t merupakan vektor

kandidat *cell state* baru, dan simbol ⊙ menyatakan operasi perkalian elemen per elemen (*Hadamard product*).

d) Output *Gate*

Output gate berfungsi untuk menentukan nilai *hidden state* pada *time step* saat ini dengan memanfaatkan *cell state* yang telah diperbarui pada tahap sebelumnya. Keluaran *hidden state* ini berfungsi sebagai *output* dari sel *LSTM* yang akan diteruskan ke *time step* berikutnya sekaligus menjadi input bagi lapisan berikutnya dalam jaringan. Persamaan 2.9 menunjukkan rumus *output gate*.

$$o_t = \sigma(Wo \cdot [h_{t-1}, x_t] + bo) \quad (2.9)$$

Nilai o_t merupakan keluaran *output gate* pada *time step* ke-t, σ adalah fungsi aktivasi *sigmoid*, Wo merupakan matriks bobot *output gate*, h_{t-1} adalah keluaran *hidden state* pada *time step* sebelumnya, x_t merupakan vektor input pada *time step* ke-t, dan bo adalah vektor bias *output gate*. Persamaan 2.10 menunjukkan rumus *Hidden State Output*.

$$h_t = o_t \odot tanh(C_t) \quad (2.10)$$

Nilai h_t merupakan keluaran *hidden state* pada *time step* ke-t yang sekaligus menjadi output akhir sel LSTM dan input bagi *time step* berikutnya, o_t adalah keluaran *output gate*, tanh adalah fungsi aktivasi tangen hiperbolik, C_t merupakan nilai *cell state* yang telah diperbarui pada *time step* ke-t, dan simbol ⊙ menyatakan operasi perkalian elemen per elemen (*Hadamard product*).

Berdasarkan beberapa penelitian tersebut dapat disimpulkan bahwa *Long Short-Term Memory (LSTM)* merupakan metode yang sesuai digunakan untuk melakukan prediksi pengeluaran pada Bengkel Mobil Ilham Putra karena mampu mempelajari pola historis data keuangan yang bersifat deret waktu dan menghasilkan prediksi yang lebih akurat.

5. Evaluasi Model

Evaluasi metrik merupakan ukuran kuantitatif yang digunakan untuk mengukur tingkat kinerja suatu model prediksi berdasarkan kedekatan antara hasil prediksi yang dihasilkan dengan nilai aktual. Dalam konteks pemodelan berbasis *machine learning* dan *time series*, evaluasi metrik berperan krusial sebagai alat ukur objektif yang membantu peneliti menentukan apakah model yang dibangun telah memenuhi standar akurasi yang diharapkan. Tanpa adanya evaluasi metrik yang tepat, sulit untuk membandingkan performa antar model maupun mengevaluasi kualitas prediksi secara ilmiah.

Menurut Nirraca & Hartati (2024), Dalam penelitiannya mengenai prediksi *Bitcoin* menggunakan metode *Long Short-Term Memory (LSTM)*, evaluasi terhadap kinerja model dilakukan dengan menerapkan beberapa metrik kesalahan, yaitu *Root Mean Squared Error (RMSE), Mean Squared Error (MSE),* dan *Mean Absolute Error (MAE)*. Ketiga metrik tersebut digunakan untuk menilai tingkat akurasi hasil prediksi yang dihasilkan oleh model. Lebih lanjut, Nugraha et al., (2026) dalam penelitiannya tentang

evaluasi kinerja model *Long Short-Term Memory (LSTM)* dan *GRU* untuk prediksi data *time series* menyatakan bahwa metrik evaluasi seperti *MAE* dan R*MSE* digunakan untuk mengukur tingkat kesalahan prediksi model serta membandingkan performa antar pendekatan yang berbeda.

a) *Mean Absolute Error (MAE)*

Mean Absolute Error (MAE) merupakan salah satu metrik evaluasi yang digunakan untuk menghitung nilai rata-rata dari selisih absolut antara data aktual dengan hasil prediksi, sehingga dapat menunjukkan tingkat kesalahan model dalam melakukan prediksi. Nilai *MAE* menggambarkan rata-rata besar kesalahan prediksi berdasarkan selisih absolut antara hasil prediksi dan data aktual tanpa mempertimbangkan arah penyimpangannya. Semakin rendah nilai *MAE* yang dihasilkan, semakin tinggi tingkat akurasi dan semakin baik kinerja model dalam menghasilkan prediksi. Rumus *Mean Absolute Error* dapat dilihat pada persamaan 2.11.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \qquad (2.\ 11)$$

Pada Persamaan 2.11 Simbol n menunjukkan jumlah data yang digunakan dalam proses evaluasi. Simbol yi menyatakan nilai aktual atau nilai sebenarnya dari data pengeluaran, sedangkan ŷi merupakan nilai hasil prediksi yang dihasilkan oleh model. Selisih antara yi dan ŷi dihitung dalam bentuk nilai absolut sehingga seluruh nilai kesalahan bernilai positif. Hasil perhitungan kemudian dijumlahkan dan dibagi dengan jumlah data untuk memperoleh rata-rata kesalahan prediksi.

b) *Mean Squared Error (MSE)*

Mean Squared Error (MSE) merupakan salah satu metrik evaluasi yang digunakan untuk mengukur rata-rata kuadrat dari selisih antara data aktual dan hasil prediksi. Proses pengkuadratan setiap nilai kesalahan bertujuan untuk memberikan bobot yang lebih besar terhadap kesalahan prediksi yang bernilai tinggi, sehingga penyimpangan yang ekstrem akan memberikan pengaruh lebih signifikan terhadap hasil evaluasi. Oleh karena itu, *MSE* sering digunakan untuk mengevaluasi performa model regresi dan prediksi data numerik. Rumus *Mean Squared Error* dapat dilihat pada persamaan 2.12.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \qquad (2.\ 12)$$

Pada Persamaan 2.12 Simbol n menunjukkan jumlah data yang digunakan dalam proses evaluasi model. Simbol yi menyatakan nilai aktual atau nilai sebenarnya, sedangkan ŷi menunjukkan nilai prediksi yang dihasilkan oleh model. Selisih antara nilai aktual dan nilai prediksi dikuadratkan untuk memberikan penalti yang lebih besar terhadap kesalahan yang tinggi. Setelah seluruh nilai kesalahan dikuadratkan, setiap nilai tersebut dijumlahkan dan kemudian dibagi dengan banyaknya data untuk memperoleh rata-rata kesalahan kuadrat yang merepresentasikan tingkat kesalahan model prediksi.

C. Teori Perancangan Sistem

1. Perancangan Sistem

Perancangan sistem merupakan tahapan yang dilakukan untuk menyusun rancangan suatu sistem sesuai dengan kebutuhan pengguna. Tahapan ini mencakup pembuatan arsitektur data, rancangan antarmuka, serta alur logika yang sesuai dengan hasil analisis kebutuhan. Dengan perancangan yang tepat, sistem dapat berjalan efektif serta mampu menyelesaikan permasalahan yang ada.

Menurut Andi et al. (2026), Perancangan sistem dilakukan untuk menciptakan perangkat lunak yang berkaitan dengan kebutuhan pengguna dan memiliki kemampuan menyesuaikan diri dengan perubahan dalam penggunaan. Penelitian lain oleh Budiarto et al. (2025) menjelaskan bahwa Perancangan arsitektur dan fungsi sistem dilakukan dengan menggunakan *Unified Modeling Language (UML). UML* berfungsi sebagai bahasa pemodelan visual yang diterima secara luas untuk menggambarkan, merancang, serta mendokumentasikan sistem perangkat lunak yang berbasis objek.

a) *Flowchart*

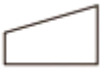
Flowchart merupakan *diagram* yang digunakan untuk memvisualisasikan urutan proses atau alur logika dalam suatu sistem secara sistematis sehingga mudah dipahami. Diagram ini menyajikan setiap tahapan proses secara berurutan dengan memanfaatkan simbol-

simbol standar, seperti persegi panjang untuk menunjukkan proses, belah ketupat sebagai simbol pengambilan keputusan, serta panah yang berfungsi menunjukkan arah aliran proses. *Flowchart* berfungsi untuk membantu pengembang dalam memahami urutan kegiatan serta hubungan antarproses sebelum diimplementasikan ke dalam program.

Menurut Listyoningrum et al., (2023), *flowchart* memiliki peran penting dalam memvisualisasikan urutan aktivitas suatu sistem, karena mampu meningkatkan efisiensi dan akurasi dalam analisis maupun pengembangan sistem berbasis komputer. Sejalan dengan itu, Noija et al., (2023) menyatakan bahwa penggunaan *flowchart* membantu perancang sistem dalam menyusun langkah-langkah kerja secara logis, sehingga mempermudah proses pemecahan masalah dan perencanaan solusi yang sistematis. Keterangan *Flowchart* dapat dilihat pada tabel 2.2.

Tabel 2. 2 *Flowchart*

No.	Notasi	Simbol	Keterangan
1.	*Flow Direction*		Simbol panah yang berfungsi sebagai instruksi alur untuk mengintegrasikan satu komponen simbol dengan simbol lainnya.
2.	*Terminator*		Penanda yang merepresentasikan titik awal (*start*) maupun titik akhir (*stop*) dari sebuah rangkaian prosedur dalam sistem.

3.	*Connector*		Digunakan sebagai titik penghubung antarproses yang terputus namun masih berada pada lembar atau halaman yang sama.
4.	*Connector*		Berfungsi sebagai media penyambung alur kerja apabila proses berlanjut ke halaman atau lembar kertas yang berbeda.
5.	*Processing*		Menggambarkan tahapan pengerjaan atau eksekusi data yang dilakukan secara otomatis oleh perangkat komputer.
6	*Manual operation*		Menunjukkan tahapan kegiatan atau pengolahan data yang dilakukan secara konvensional tanpa melibatkan bantuan komputer.
7	*Decision*		percabangan untuk memilih langkah selanjutnya berdasarkan kriteria atau kondisi logika tertentu yang tersedia.
8	*Input-Output*		Representasi dari proses memasukkan data atau menghasilkan informasi tanpa bergantung pada jenis media perangkat yang digunakan.
9	*Manual input*		Tahapan memasukkan data ke dalam sistem secara langsung oleh pengguna melalui perangkat papan ketik (*keyboard*).

10	Preparation		Instruksi untuk menyiapkan ruang penyimpanan atau memori yang akan dipakai sebagai tempat pemrosesan data.
11	*Predifined proses*		Menunjukkan pelaksanaan sebuah sub-program, prosedur, atau bagian tertentu dari sistem yang telah didefinisikan sebelumnya.
12	*Display*		Menyatakan penggunaan perangkat keluaran visual seperti layar monitor, printer, atau plotter untuk menampilkan hasil.
13	*Disk and online storage*		Mengindikasikan bahwa data masukan berasal dari cakram penyimpanan atau hasil pengolahan disimpan ke dalam media storage online.
14	Dokumen		Keterangan bahwa input berasal dari berkas fisik/kertas atau output yang akan dicetak dalam bentuk dokumen cetak

b) UML (Unified Modeling Language)

UML merupakan bahasa pemodelan standar yang dimanfaatkan untuk memvisualisasikan, merancang, dan mendokumentasikan sistem perangkat lunak berorientasi objek. *UML* menyediakan berbagai jenis diagram yang digunakan untuk merepresentasikan struktur maupun perilaku suatu sistem. Melalui

pemodelan menggunakan UML, pengembang dapat memperoleh gambaran yang lebih jelas mengenai alur kerja sistem serta hubungan antar komponen yang saling berinteraksi.

Menurut Firdaus et al., (2024), *UML* memberikan cara yang konsisten dalam menggambarkan struktur maupun perilaku sistem berbasis objek. Sejalan dengan itu, Aryani et al., (2025) menyatakan bahwa *UML* menyajikan informasi struktur sistem dalam bentuk *visual* yang mudah dipahami.

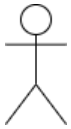
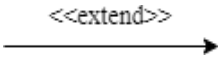
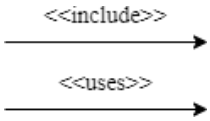
1. *Use Case Diagram*

Use case diagram merupakan *diagram* yang digunakan untuk menggambarkan interaksi antara aktor (pengguna) dengan sistem dalam menjalankan berbagai fungsi yang tersedia. Diagram ini membantu mengidentifikasi kebutuhan fungsional sistem dari sudut pandang pengguna sehingga setiap fungsi yang harus disediakan dapat dipetakan secara lebih jelas. Oleh karena itu, *use case diagram* umumnya disusun pada tahap awal perancangan sistem sebagai dasar dalam proses pengembangan.

Menurut Firdaus et al., (2024), *use case diagram* merupakan pola atau gambaran tentang sifat atau kebiasaan sistem. Sementara itu, Menurut Wayahdi & Ruziq (2023), *Use Case Diagram* menjadi alat penting untuk menjelaskan bagaimana sistem berinteraksi dengan pengguna dalam konteks fungsi yang

disediakan. Keterangan *Use Case Diagram* dapat dilihat pada tabel 2.3.

Tabel 2. 3 *Use case Diagram*

No.	Notasi	Simbol	Keterangan
1.	*Actor*		Perwakilan dari pengguna atau pihak luar (bisa manusia maupun sistem lain) yang menjalin interaksi aktif dengan aplikasi.
2	*Extend*		Hubungan opsional di mana sebuah use case dapat memperluas perilaku atau fungsi dari use case utama lainnya.
3.	*Uses*/*Include*		Relasi ketergantungan di mana suatu fungsi utama memerlukan fungsionalitas tambahan agar prosesnya dapat diselesaikan dengan lengkap.
4.	*Association*		Interaksi yang terjadi terhadap *actor* dan *user*
5.	*Generalizato n*		Menunjukkan hubungan kearah *use case* yang lebih umum

2. *Activity Diagram*

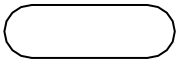
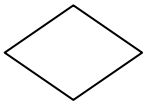
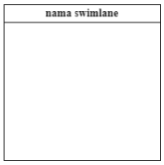
Activity diagram merupakan diagram yang digunakan untuk memodelkan alur aktivitas atau proses bisnis yang berlangsung dalam suatu sistem. Diagram ini menggambarkan urutan setiap aktivitas secara sistematis sehingga tahapan kerja

dan aliran proses dapat dipahami dengan lebih jelas. *Diagram* ini membantu memahami bagaimana suatu proses berlangsung, termasuk kondisi keputusan yang memengaruhi transisi antar aktivitas. *Activity diagram* juga mempermudah identifikasi logika kerja dalam sistem.

Menurut Firdaus et al., (2024), *activity diagram* mampu menyajikan visualisasi yang rinci mengenai urutan aktivitas yang berlangsung di dalam suatu sistem. Melalui *diagram* ini, setiap tahapan proses dapat dipahami dengan lebih jelas sehingga kebutuhan proses dan alur kerja sistem dapat diidentifikasi secara lebih akurat. Hal serupa dijelaskan oleh Wayahdi & Ruziq (2023) bahwa *activity diagram* dapat dimanfaatkan untuk memvisualisasikan logika bisnis serta alur kerja suatu sistem secara sistematis. Dengan penyajian tersebut, analis maupun pengembang dapat memahami urutan proses yang terjadi dengan lebih mudah sehingga mendukung proses analisis dan pengembangan sistem. Keterangan *Activity Diagram* dapat dilihat pada tabel 2.4.

Tabel 2. 4 *Activity Diagram*

No.	Notasi	Simbol	Keterangan
1.	Status Awal/akhir		Simbol yang mengidentifikasi titik pembuka dan titik penutup dari seluruh rangkaian aktivitas dalam diagram.

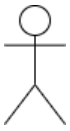
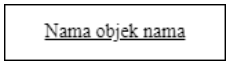
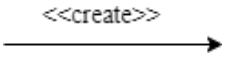
2.	Aktivitas		Representasi dari sebuah pekerjaan atau tindakan yang dilakukan oleh sistem, biasanya dituliskan menggunakan kata kerja.
3.	*Decision*		Hubungan yang terjadi saat kita memiliki beberapa pilihan untuk membuat keputusan dalam suatu aktivitas.
4.	*Swimlane*		Pembagian area dalam diagram yang memetakan tanggung jawab masing-masing bagian organisasi terhadap aktivitas yang berjalan.
5.	*Join*		Hubungan penggabungan di mana dua atau lebih aliran aktivitas yang berjalan paralel menyatu kembali menjadi satu jalur.

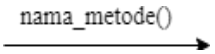
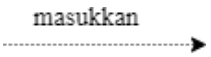
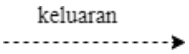
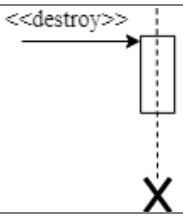
3. *Sequence Diagram*

Sequence diagram merupakan diagram yang digunakan untuk memodelkan interaksi antarobjek dalam suatu sistem berdasarkan urutan waktu pelaksanaannya. Diagram ini memperlihatkan proses pertukaran pesan antarobjek dalam menjalankan suatu fungsi atau layanan hingga proses tersebut selesai. Melalui sequence diagram, alur komunikasi serta hubungan antar komponen sistem dapat dipahami secara lebih jelas dan terstruktur.

Menurut Firdaus et al., (2024), *sequence diagram* memiliki peran penting dalam menjelaskan alur komunikasi antar bagian sistem secara kronologis. Senada dengan itu, Wayahdi & Ruziq (2023) menyebutkan bahwa *sequence diagram* digunakan untuk mendokumentasikan skenario interaksi antar objek dalam sistem sehingga mempermudah pengembangan perangkat lunak berbasis objek. Keterangan Sequence Diagram dapat dilihat pada tabel 2.5.

Tabel 2. 5 *Sequence Diagram*

No.	Notasi	Simbol	Keterangan
1.	*Actor*		Entitas di luar sistem yang memicu atau terlibat dalam urutan interaksi antarobjek.
2.	*Lifeline*		Garis vertikal yang memvisualisasikan durasi keberadaan atau masa aktif sebuah objek selama interaksi berlangsung.
3.	Objek		Merupakan interaksi pesan yang dilakukan oleh objek.
4.	Waktu aktif		Menandakan interaksi objek.
5.	Pesan tipe *create*		Instruksi yang dikirimkan oleh satu objek untuk membuat atau menginisialisasi objek baru lainnya.

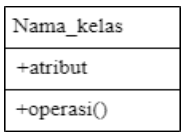
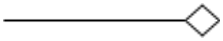
6.	Pesan tipe *call*		Aksi memanggil fungsi, metode, atau prosedur tertentu yang terdapat pada objek tujuan.
7.	Pesan tipe *send*		Proses pengiriman data, informasi, atau masukan dari satu komponen ke komponen lain dalam sistem.
8.	Pesan tipe *return*		Pernyataan objek menjalankan suatu perintah dan memberi keluaran ke objek tertentu.
9.	Pesan tipe *destroy*		Pernyataan objek yang dimatikan oleh objek lainnya.

4. *Class Diagram*

Class diagram merupakan *diagram* yang digunakan untuk merepresentasikan struktur statis suatu sistem dengan menampilkan kelas-kelas yang dimiliki beserta atribut, metode (*operation*), dan hubungan antar kelas. Diagram ini memberikan gambaran mengenai organisasi serta keterkaitan setiap komponen dalam sistem berorientasi objek. *Diagram* ini berfungsi untuk mendefinisikan kerangka dasar sistem sebelum diimplementasikan ke dalam kode program. Dalam pengembangan perangkat lunak berbasis objek, *class diagram* menjadi salah satu alat utama yang digunakan.

Menurut Firdaus et al., (2024), *class diagram* adalah kumpulan objek yang memiliki struktur, sifat, dan relasi yang umum. Sementara itu, Wayahdi & Ruziq (2023) menambahkan bahwa *class diagram* sangat bermanfaat dalam mengorganisasikan komponen perangkat lunak karena dapat menggambarkan hubungan antar kelas secara jelas dan terstruktur. Keterangan Class Diagram dapat dilihat pada tabel 2.6.

Tabel 2. 6 *Class Diagram*

No	Notasi	Simbol	Keterangan
1.	*Class*		Blok bangunan utama sistem yang berisi kumpulan atribut (data) dan operasi (fungsi) yang dimiliki oleh objek.
2.	*Association*		Merupakan relasi antarkelas, yang dilengkapi dengan *multiplicity*
3.	Generalisasi		Merupakan relasi kelas dengan bermakna generalisai.
4.	*Dependency*		Merupakan realasi Keberuntungan antar kelas.
5.	*Aggregation*		Jenis hubungan yang menyatakan bahwa suatu kelas merupakan bagian dari kesatuan yang lebih besar (whole-part).

6.	*Interface*		Merupakan bagian yang ditampilkan namun tidak ada isi dan atribut.
7.	*Directed Association*		Hubungan antara dua kelas di mana satu kelas digunakan oleh kelas lain dan memiliki *multiplicity* yang spesifik.

D. Pengujian Sistem

Black Box Testing pada penelitian ini diterapkan untuk menguji seluruh fungsi yang tersedia pada aplikasi, meliputi proses memasukkan data pengeluaran, penyimpanan data ke dalam *database Firebase*, serta memastikan setiap fitur berjalan sesuai dengan kebutuhan fungsional sistem, serta tampilan laporan dan hasil analisis. Proses pengujian dilaksanakan dengan menggunakan berbagai variasi data masukan untuk mengevaluasi apakah sistem dapat menghasilkan keluaran yang sesuai dengan hasil yang diharapkan berdasarkan kebutuhan fungsional. Metode ini sangat penting untuk memastikan bahwa aplikasi berjalan dengan baik dari sisi pengguna (*user perspective*), mudah digunakan, serta bebas dari kesalahan fungsional. Dengan demikian, hasil pengujian ini dapat menjadi indikator bahwa sistem yang dikembangkan telah memenuhi kebutuhan dan siap untuk digunakan secara optimal. Contoh pengujian *Black Box* dapat dilihat pada tabel 2.7.

Tabel 2. 7 Contoh Pengujian *Black Box*

Fitur yang diuji	Skenario/Input	Hasil yang diharapkan	Keterangan
Validasi *Input* Kosong	Mengosongkan formulir *input* dan menekan tombol simpan.	Sistem menampilkan pesan peringatan bahwa data tidak boleh kosong.	Sesuai/Tidak
Validasi Tipe Data	Memasukkan karakter huruf pada kolom yang seharusnya diisi angka (misal: harga).	Sistem menolak *input* dan memberikan notifikasi kesalahan format.	Sesuai/Tidak
Keberhasilan Simpan Data	Mengisi seluruh formulir dengan benar dan menekan tombol simpan.	Data berhasil tersimpan ke dalam basis data dan muncul pada daftar tampilan.	Sesuai/Tidak
Fungsi Tombol Navigasi	Menekan tombol "Kembali" atau ikon menu tertentu.	Sistem mengarahkan pengguna ke halaman yang sesuai tanpa adanya *error* atau *force close*.	Sesuai/Tidak

Menurut Febriyanti et al. (2021), *Black Box Testing* merupakan metode pengujian perangkat lunak yang dilakukan dengan cara menguji fungsi sistem tanpa melihat struktur kode program di dalamnya. Pengujian ini berfokus pada kesesuaian antara input yang diberikan dengan output yang dihasilkan oleh sistem. Sejalan dengan itu, Shadiq et al., (2021) menyatakan bahwa *Black Box Testing* diterapkan untuk memastikan bahwa seluruh fungsi pada sistem telah berjalan sesuai dengan kebutuhan pengguna serta memenuhi spesifikasi yang telah ditetapkan selama proses perancangan.

E. Kajian Empiris

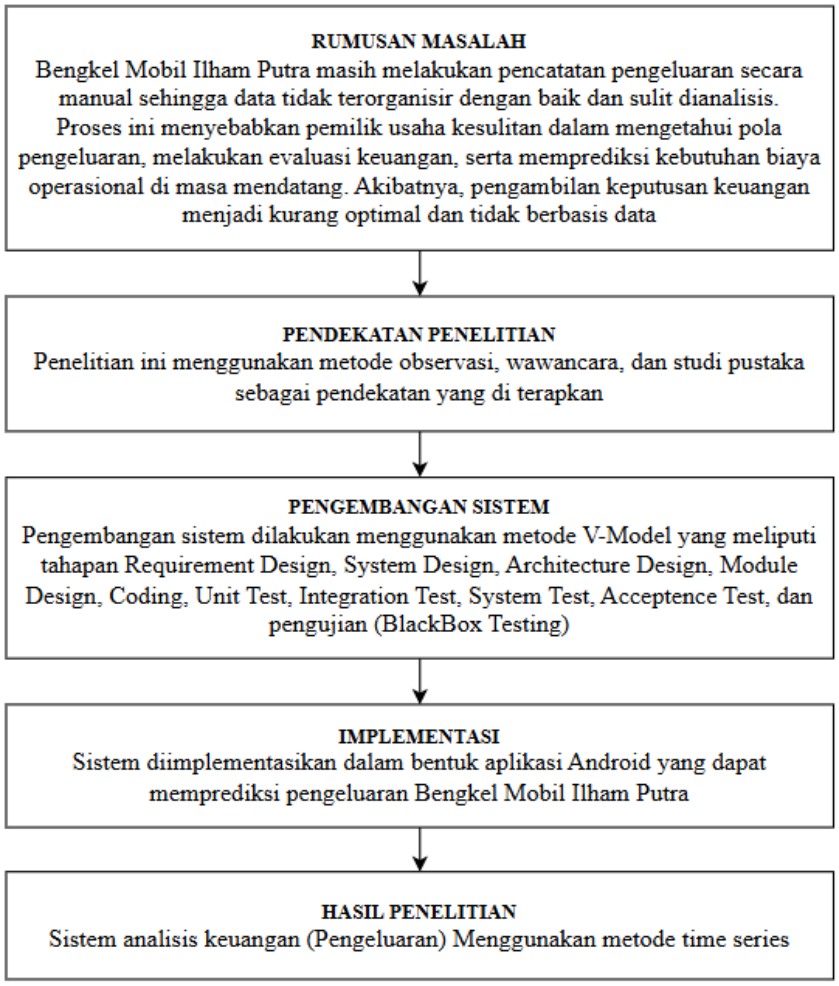
Beberapa penelitian terdahulu yang relevan dengan penelitian ini antara lain penelitian oleh Lampang et al. (2023) yang membahas pentingnya

penggunaan aplikasi *mobile* dalam pengelolaan keuangan untuk meningkatkan kesadaran finansial. Penelitian lain oleh Trivaika & Senubekti (2022) mengembangkan aplikasi pencatatan keuangan berbasis *Android* yang membantu pengguna dalam mencatat transaksi secara praktis. Selain itu, penelitian oleh Nugraha et al. (2026) menunjukkan bahwa metode *Long short-Term Memory (LSTM)* efektif digunakan untuk menganalisis data berbasis waktu dan menghasilkan prediksi yang cukup akurat. Berdasarkan kajian tersebut, penelitian ini memiliki perbedaan pada integrasi antara sistem pendukung berbasis *Android* dengan metode *time series* yang difokuskan pada analisis pengeluaran di lingkungan usaha bengkel, sehingga diharapkan mampu memberikan solusi yang lebih spesifik dan aplikatif.

F. Kerangka Berfikir

Kerangka berfikir dalam penelitian ini disusun berdasarkan permasalahan yang terjadi pada Bengkel Mobil Ilham Putra, yaitu belum adanya sistem yang mampu mencatat dan menganalisis pengeluaran secara terstruktur. Dari permasalahan tersebut, peneliti menggunakan pendekatan penelitian kuantitatif dengan memanfaatkan metode *time series* untuk menganalisis data pengeluaran. Selanjutnya dilakukan proses pengembangan sistem berbasis *Android* yang terintegrasi dengan *Database* Firebase. Sistem yang telah dikembangkan kemudian diimplementasikan dan diuji menggunakan metode *Black Box Testing* untuk memastikan fungsionalitasnya berjalan dengan baik. Hasil dari penelitian ini diharapkan berupa informasi tren dan prediksi pengeluaran yang dapat membantu dalam pengambilan keputusan keuangan.

Adapun bagan kerangka berfikir pada penelitian ini dapat dilihat pada gambar 2.3.



Gambar 2. 3 Kerangka Berfikir