**ANALISIS PREDIKSI HARGA SAHAM HAPAG-LLOYD AG MENGGUNAKAN METODE LONG SHORT-TERM MEMORY (LSTM)**

**SKRIPSI**

**Diajukan untuk Memenuhi Salah Satu Syarat Kelulusan Strata 1   
di Program Studi Informatika Universitas Widyatama**

**Oleh**

|  |  |  |
| --- | --- | --- |
| **NAMA** | **:** | **FRANSISCUS KRISTIAN SUSANTO** |
| **NPM** | **:** | **40621100012** |



**PROGRAM STUDI INFORMATIKA**

**FAKULTAS TEKNIK UNIVERSITAS WIDYATAMA**

**BANDUNG**

**2025**

LEMBAR PENGESAHAN

**ANALISIS PREDIKSI HARGA SAHAM HAPAG-LLOYD AG MENGGUNAKAN METODE LONG SHORT-TERM MEMORY (LSTM)**

**SKRIPSI**

Program Studi Informatika

Fakultas Teknik

Universitas Widyatama

**Oleh:**

**Fransiscus Kristian Susanto**

**40621100012**

Telah disetujui dan disahkan di Bandung, \_\_\_\_\_\_\_\_\_ 2025

Menyetujui,

|  |
| --- |
| **Pembimbing,**  **Dr. Feri Sulianta, S.T., M.T.**  **NIDN. 0420027901** |

Mengetahui,

|  |  |
| --- | --- |
| **Ketua Program Studi Informatika,**  **Ari Purno Wahyu Wibowo, S.Kom., M.Kom.**  **NIDN. 0415078402** | **Dekan Fakultas Teknik,**  **Dr. Arief Rahmana, ST., MT., CIPMP., IPU., ASEAN Eng.**  **NIDN. 0429097401** |

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini:

|  |  |  |
| --- | --- | --- |
| Nama | : | Fransiscus Kristian Susanto |
| NPM | : | 40621100012 |
| Tempat, Tanggal Lahir | : | Bandung, 21 Juni 2003 |
| Alamat Asal | : | Bandung |
| Alamat Bandung | : | Komplek Permata Biru Blok H No. 93, Cinunuk, Kec. Cileunyi |

Dengan ini menyatakan bahwa skripsi dengan judul: **ANALISIS PREDIKSI HARGA SAHAM HAPAG-LLOYD AG MENGGUNAKAN METODE LONG SHORT-TERM MEMORY (LSTM)** adalah hasil pekerjaan saya dan seluruh ide, pendapat, atau materi dari sumber lain telah dikutip dengan cara penulisan referensi yang sesuai.

Pernyataan ini saya buat dengan sebenar-benarnya dan jika pernyataan ini tidak sesuai dengan kenyataan, maka saya bersedia menanggung sanksi yang akan dikenakan kepada saya termasuk pencabutan gelar Sarjana yang telah saya dapatkan.

Bandung, \_\_\_\_\_\_\_\_\_ 2025

Fransiscus Kristian Susanto

ABSTRAK

Analisis prediksi harga saham merupakan hal yang penting dalam mempertimbangan perdagangan modal terhadap suatu perusahaan. Dengan menggunakan metode Long Short-Term Memory (LSTM), prediksi tersebut memberikan gambaran terkait mengenali pola terhadap data time series dan memperkirakan pergerakan harga saham dalam data historis. Analisis prediksi yang dikembangkan memanfaatkan algoritma LSTM sebagai sebuah jenis jaringan saraf berulang yang dirancang untuk mengatasi masalah dalam urutan data waktu panjang. Penelitian ini melibatkan pengumpulan data historis harga saham Hapag-Lloyd AG dan menjalankan data preprocessing terhadap dataset. Model LSTM kemudian dikembangkan dan dilatih menggunakan dataset tersebut hingga akhirnya memberikan evaluasi terhadap model tersebut. Hasil analisis diharapkan dapat memberikan wawasan tentang efektivitas metode LSTM dalam memprediksi harga saham dan menjadi salah satu bahan pertimbangan dalam menentukan keputusan pemegangan modal di suatu perusahaan.

**Kata Kunci:** Saham, Prediksi, LSTM, Harga.

*ABSTRACT*

*Stock price prediction analysis is important in considering capital trading for a company. By using the Long Short-Term Memory (LSTM) method, the prediction provides an overview of recognizing patterns in time series data and estimating stock price movements in historical data. The prediction analysis developed utilizes the LSTM algorithm as a type of recurrent neural network designed to solve problems in long-term data sequences. This study involves collecting historical Hapag-Lloyd AG stock price data and running data preprocessing on the dataset. The LSTM model is then developed and trained using the dataset until finally providing an evaluation of the model. The results of the analysis are expected to provide insight into the effectiveness of the LSTM method in predicting stock prices and become one of the considerations in determining capital holding decisions in a company.*

***Keywords:*** *Stocks, Prediction, LSTM, Price.*

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa karena dengan rahmat dan karunia-Nya, penulis dapat menyelesaikan tugas akhir yang berjudul **“Analisis Prediksi Harga Saham Hapag-Lloyd AG Menggunakan Metode Long Short-Term Memory (LSTM)”** tepat pada waktunya. Tugas akhir ini disusun dalam rangka memenuhi salah satu syarat untuk menyelesaikan studi Strata 1 pada Program Studi Informatika Universitas Widyatama.

Dalam menyelesaikan tugas akhir ini penulis mendapatkan banyak bantuan, masukan, bimbingan, dan dukungan dari berbagai pihak. Oleh karena itu, penulis ingin mengucapkan terima kasih kepada:

1. Orang tua dan kakak tercinta yang memberikan perhatian dan kasih sayang kepada penulis.
2. Dr. Feri Sulianta, S.T., M.T. selaku dosen pembimbing yang telah memberi masukan dan waktunya dalam membimbing penulis.
3. Seluruh dosen pengajar di Program Studi Informatika Universitas Widyatama yang telah membekali penulis dengan berbagai ilmu pengetahuan.
4. Teman-teman Program Studi Informatika Angkatan 2021 yang telah memberi dukungan dalam pengerjaan tugas akhir.
5. Semua teman dan pihak yang telah memberikan bantuan kepada penulis baik secara langsung maupun tidak langsung.

Penulis menyadari bahwa penyusunan tugas akhir ini masih banyak kekurangan sehingga kritik dan saran yang membangun sangat membantu penulis guna perbaikan tugas akhir ini. Penulis berharap tugas akhir ini dapat bermanfaat bagi semua pihak.

Bandung, \_\_\_\_\_\_\_\_\_ 2025

Fransiscus Kristian Susanto

DAFTAR ISI

[LEMBAR PENGESAHAN i](#_Toc198712923)

[SURAT PERNYATAAN ii](#_Toc198712924)

[ABSTRAK iii](#_Toc198712925)

[*ABSTRACT* iv](#_Toc198712926)

[KATA PENGANTAR v](#_Toc198712927)

[DAFTAR ISI vi](#_Toc198712928)

[DAFTAR GAMBAR viii](#_Toc198712929)

[DAFTAR TABEL ix](#_Toc198712930)

[DAFTAR LAMPIRAN x](#_Toc198712931)

[BAB I PENDAHULUAN 1](#_Toc198712932)

[1.1 Latar Belakang 1](#_Toc198712933)

[1.2 Rumusan Masalah 2](#_Toc198712934)

[1.3 Tujuan Penelitian 3](#_Toc198712935)

[1.4 Batasan Penelitian 3](#_Toc198712936)

[1.5 Manfaat Penelitian 4](#_Toc198712937)

[BAB II KAJIAN PUSTAKA 5](#_Toc198712938)

[2.1 Tinjauan Pustaka 5](#_Toc198712939)

[2.2 Landasan Teori 11](#_Toc198712940)

[2.2.1 Saham 11](#_Toc198712941)

[2.2.2 Hapag-Lloyd AG 11](#_Toc198712942)

[2.2.3 Pengembangan Sistem *Prototyping* 11](#_Toc198712943)

[2.2.4 *Data Mining* 11](#_Toc198712944)

[2.2.5 *Long Short-Term Memory* 11](#_Toc198712945)

[2.2.6 *Cross-Industry Standard Process for Data Mining* 11](#_Toc198712946)

[2.2.7 *Time Series Forecasting* 11](#_Toc198712947)

[2.2.8 Streamlit 11](#_Toc198712948)

[BAB III METODE PENELITIAN 12](#_Toc198712949)

[3.1 Metode *Prototyping* 12](#_Toc198712950)

[3.2 Alur Kerja CRISP-DM 12](#_Toc198712951)

[3.3 Kebutuhan Sistem 12](#_Toc198712952)

[3.4 Perancangan Sistem 12](#_Toc198712953)

[3.4.1 Use Case 12](#_Toc198712954)

[3.4.2 Activity 12](#_Toc198712955)

[3.4.3 Sequence 12](#_Toc198712956)

[3.5 Perancangan Antar Muka 12](#_Toc198712957)

[BAB IV HASIL PENELITIAN DAN PEMBAHASAN 13](#_Toc198712958)

[4.1 Implementasi Sistem 13](#_Toc198712959)

[4.2 Implementasi Antarmuka 13](#_Toc198712960)

[4.3 Pengujian Sistem 13](#_Toc198712961)

[BAB V PENUTUP 14](#_Toc198712962)

[5.1 Kesimpulan 14](#_Toc198712963)

[5.2 Saran 14](#_Toc198712964)

[DAFTAR PUSTAKA 15](#_Toc198712965)

[LAMPIRAN 16](#_Toc198712966)

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

[**Lampiran 1. Riwayat Hidup** 16](#_Toc197436132)

[**Lampiran 2. Kartu Bimbingan Tugas Akhir** 17](#_Toc197436133)

BAB I  
PENDAHULUAN

1.1 Latar Belakang

Investasi merupakan salah satu cara penting bagi individu maupun perusahaan untuk meningkatkan nilai aset mereka. Dalam dunia investasi terdapat berbagai jenis instrumen yang dapat dipilih oleh investor di mana salah satunya adalah saham. Investasi saham memberikan kesempatan bagi investor untuk memperoleh keuntungan yang menarik. Oleh karena itu, saham menjadi pilihan yang menarik untuk investasi jangka panjang meskipun terdapat risiko yang perlu dipertimbangkan.

Investasi saham umumnya dilakukan pada perusahaan terbuka di mana terdapat banyak perusahaan yang sahamnya diperdagangkan di pasar modal dan dapat dibeli oleh masyarakat umum. Salah satu sektor yang menarik perhatian investor adalah sektor logistik yang memiliki peran penting dalam rantai pasokan barang dan distribusi global. Di sektor logistik terdapat perusahaan-perusahaan transportasi peti kemas yang berperan penting dalam mendukung kegiatan perdagangan internasional. Salah satu perusahaan yang memiliki pengaruh besar dalam sektor ini adalah Hapag-Lloyd AG, sebuah perusahaan transportasi peti kemas asal Jerman yang terkenal secara global.

Namun, investasi saham tidaklah tanpa tantangan. Salah satu tantangan terbesar adalah memprediksi harga saham yang sering kali dipengaruhi oleh berbagai faktor ekonomi, politik, dan sosial baik dari satu negara tersebut maupun antar negara. Maka karena itu analisis prediksi harga saham menjadi sangat penting bagi para investor dalam membuat keputusan yang lebih baik dan lebih menguntungkan.

Dalam hal ini, teknik analisis prediksi harga saham dapat menggunakan metode *data mining*. *Data mining* atau penambangan data merupakan proses pencarian pola atau informasi berharga dari data yang besar dan dapat digunakan untuk menganalisis berbagai aspek yang mempengaruhi harga saham. Salah satu metodologi yang sering digunakan dalam proyek penambangan data adalah CRISP-DM (*Cross-Industry Standard Process for Data Mining*). CRISP-DM memberikan panduan dalam melakukan analisis data dengan langkah-langkah yang sistematis yang mulai dari pemahaman masalah hingga penerapan model.

Penelitian terkait analisis prediksi harga saham dengan menggunakan metodologi CRISP-DM telah dilakukan. Penelitian Farid dkk. (2023) mengimplementasikan CRISP-DM untuk menganalisis prediksi harga saham dengan menggunakan teknik penambangan data seperti klasifikasi. Penelitian tersebut menyimpulkan bahwa meskipun penambangan datadapat memberikan kontribusi dalam memprediksi harga saham, akurasi yang rendah menunjukkan perlunya model yang lebih menyeluruh untuk meningkatkan akurasi prediksi [1].

Salah satu metode yang populer dalam mengantisipasi hal di atas untuk analisis prediksi harga saham adalah model *Long Short-Term Memory* (LSTM), yang merupakan jenis jaringan saraf tiruan berbasis *Recurrent Neural Networks* (RNN). LSTM sangat efektif dalam memproses data deret waktu (*time series*), seperti data harga saham yang memiliki ketergantungan waktu. Model ini memiliki kemampuan untuk mengingat pola jangka panjang dalam data yang sangat berguna dalam memprediksi pergerakan harga saham. Berdasarkan penelitian yang dilakukan oleh Pawar dkk. (2019), hasil yang diperoleh menunjukkan bahwa model LSTM dapat memberikan prediksi harga saham yang lebih akurat dibandingkan dengan algoritma *machine learning* tradisional lainnya [2].

Penelitian ini bertujuan untuk menganalisis prediksi harga saham Hapag-Lloyd AG menggunakan metode LSTM dengan menerapkan metodologi CRISP-DM dalam proses analisis data. Melalui penelitian ini, diharapkan dapat memberikan wawasan lebih dalam tentang cara kerja LSTM dalam memprediksi harga saham serta memberikan informasi yang berguna bagi investor dalam membuat keputusan investasi.

1.2 Rumusan Masalah

Adapun rumusan masalah dalam tugas akhir ini adalah sebagai berikut.

1. Bagaimana cara memprediksi harga saham Hapag-Lloyd AG menggunakan metode LSTM?
2. Bagaimana evaluasi model LSTM dapat memberikan prediksi yang akurat terhadap harga saham Hapag-Lloyd AG?
3. Bagaimana penerapan metodologi CRISP-DM dapat meningkatkan keakuratan dalam analisis prediksi harga saham menggunakan LSTM?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah yang telah dijabarkan, adapun tujuan penelitian adalah sebagai berikut.

1. Mengembangkan model prediksi harga saham Hapag-Lloyd AG menggunakan metode LSTM.
2. Menganalisis dan mengevaluasi kemampuan model LSTM dapat memberikan prediksi yang akurat terhadap harga saham Hapag-Lloyd AG dengan menggunakan data historis yang tersedia.
3. Menyusun penerapan metodologi CRISP-DM dalam proses analisis prediksi harga saham, yang melibatkan tahapan-tahapan seperti pemahaman masalah, pemrosesan data, pemodelan, evaluasi, dan penerapan aplikasi.

1.4 Batasan Penelitian

Adapun batasan masalah dalam tugas akhir ini adalah sebagai berikut.

1. Data yang digunakan dalam penelitian ini hanya mencakup harga saham Hapag-Lloyd AG selama periode tertentu yang tersedia di Yahoo Finance sebagai sumber data. Data tersebut mencakup harga penutupan harian yang digunakan untuk model prediksi harga saham.
2. Penelitian ini hanya menggunakan metode LSTM untuk menganalisis prediksi harga saham Hapag-Lloyd AG tanpa membandingkannya dengan metode lain.
3. Penelitian ini terbatas pada implementasi metodologi CRISP-DM dalam analisis prediksi harga saham dengan menggunakan LSTM tanpa mencakup analisis fundamental lainnya yang berkaitan dengan saham.

1.5 Manfaat Penelitian

Adapun manfaat dalam penelitian ini adalah sebagai berikut.

1. Penelitian ini dapat memberikan wawasan yang lebih baik dalam memprediksi harga saham Hapag-Lloyd AG, yang dapat digunakan untuk pengambilan keputusan investasi yang lebih informatif.
2. Hasil penelitian ini dapat menjadi referensi untuk penelitian lebih lanjut dalam penggunaan metode LSTM untuk analisis prediksi harga saham, serta penerapan metodologi CRISP-DM dalam proyek penambangan data lainnya.
3. Penelitian ini dapat memberikan contoh implementasi praktis dalam pengolahan dan pemodelan data urutan waktu menggunakan metode LSTM untuk prediksi harga saham.
4. Penelitian ini dapat memberikan pemahaman lebih luas mengenai pentingnya investasi saham, khususnya dalam sektor logistik, serta memberikan gambaran tentang cara kerja teknik prediksi harga saham menggunakan pendekatan *data mining*.

BAB II  
KAJIAN PUSTAKA

2.1 Tinjauan Pustaka

Bagian ini membahas sejumlah penelitian terdahulu yang memiliki keterkaitan dengan topik skripsi ini. Studi-studi sebelumnya memberikan wawasan mengenai konsep dasar, penerapan, serta temuan yang diperoleh dari penelitian sejenis. Pemanfaatan penelitian terdahulu sebagai landasan dalam tinjauan pustaka digunakan untuk mendukung analisis prediksi harga saham dengan menggunakan metode LSTM.

Salah satu penelitian yang mendukung kajian ini dilakukan oleh Nadia Afrin Ritu dkk dengan menggunakan data dari Bursa Saham Dhaka yang mencakup empat bank, seperti Bank Asia, Brac Bank, Dhaka Bank, dan Islami Bank. Data yang dianalisis berasal dari periode 15 Juli 2018 hingga 25 Maret 2020 dengan fokus pada variabel harga penutupan. Data dibagi dengan proporsi 80% untuk pelatihan dan 20% untuk pengujian model. Sebelum diterapkan dalam model LSTM, seluruh data terlebih dahulu melalui proses standarisasi. Evaluasi performa model dilakukan menggunakan metrik *Root Mean Square Error* (RMSE) dengan nilai untuk masing-masing adalah 4,5124 (Brac Bank), 1,1898 (Bank Asia), 0,84084 (Islami Bank), dan 0,33193 (Dhaka Bank) yang menunjukkan variasi performa antar saham yang diuji [3].

Dalam penelitian yang dilakukan oleh Firhan Abdillah Mahbubi dkk, model LSTM dimanfaatkan untuk meramalkan harga saham nikel dari PT Resource Alam Indonesia Tbk. Data historis yang digunakan mencakup periode Januari 2021 hingga Mei 2024 yang kemudian dibagi menjadi dua bagian, seperti 80% untuk pelatihan dan 20% untuk pengujian. Berbagai hyperparameter disesuaikan untuk memaksimalkan akurasi model selama proses pelatihan. Hasil terbaik diperoleh saat menggunakan optimasi Adam dengan jumlah 150 epoch. Evaluasi performa model menunjukkan metrik prediksi yang cukup rendah dengan nilai *Mean Absolute Error* (MAE) sebesar 33,15, RMSE sebesar 48,14, *Mean Squared Error* (MSE) sebesar 2317,33, dan *Mean Absolute Percentage Error* (MAPE) sebesar 7,39 [4].

Penelitian yang dilakukan oleh Rendy Saputra dkk menggunakan data historis saham Tesla Inc. yang mencakup periode dari 21 Mei 2020 hingga 21 Mei 2024. Data yang dianalisis meliputi harga pembukaan, harga tertinggi, harga terendah, dan harga penutupan harian. Seluruh variabel masukan terlebih dahulu dinormalisasi dengan menggunakan teknik StandardScaler. Data kemudian dibagi menjadi 80% untuk pelatihan dan 20% untuk pengujian. Model prediksi yang dikembangkan mengandalkan dua lapisan LSTM yang masing-masing berisi 50 neuron dan diakhiri dengan satu lapisan dense. Proses pelatihan berlangsung selama 100 epoch dengan batch size sebesar 4 dan tanpa menerapkan teknik *shuffling*. Hasil evaluasi menunjukkan performa model yang cukup baik dengan nilai RMSE sebesar 0,042 dan MAPE sebesar 18,7% [5].

Dalam penelitian yang dilakukan oleh Muhammad Sahroni dkk, data saham PT Bank Syariah Indonesia yang diperoleh dari situs Yahoo Finance dengan fokus data pada harga penutupan harian yang telah disesuaikan (*adjusted close*) selama periode 31 Mei 2019 hingga 31 Mei 2024. Data tersebut dibagi menjadi dua bagian, yakni 80% untuk pelatihan dan 20% untuk pengujian. Sebelum pemodelan, data terlebih dahulu dinormalisasi menggunakan metode MinMaxScaler. Model yang digunakan adalah LSTM dengan parameter optimasi Adam. Struktur jaringan terdiri atas dua lapisan tersembunyi LSTM dengan fungsi aktivasi tanh dan sigmoid serta ditambahkan mekanisme dropout. Pengujian dilakukan dengan berbagai kombinasi parameter seperti jumlah neuron sebesar 50, ukuran batch masing-masing sebesar 4, 16, 64, 128, dan jumlah epoch masing-masing sebesar 50, 100, 150, 200. Pemilihan parameter terbaik dilakukan dengan pendekatan Grid Search dan cross-validation. Evaluasi model menggunakan metrik RMSE dan MAPE menunjukkan bahwa kombinasi batch size 4 dan epoch 200 memberikan hasil terbaik dengan nilai MAPE 1,72% dan RMSE 53,49. Secara keseluruhan model mampu memprediksi harga saham dengan tingkat akurasi sebesar 98,28% [6].

Dalam penelitian yang dilakukan oleh Bayu Dwi Handika dan Sugianto, metode LSTM diterapkan untuk melakukan prediksi harga saham dari dua perusahaan perbankan, yaitu PT Bank BTPN Tbk (BTPN) dan PT Bank Rakyat Indonesia (Persero) Tbk (BBRI). Proses pelatihan model dilakukan dengan membagi data historis menjadi 80% untuk pelatihan dan 20% untuk pengujian. Data yang digunakan terlebih dahulu dinormalisasi menggunakan teknik Min-Max Scaler dan jumlah epoch yang digunakan adalah 66 untuk model saham BTPN serta 55 untuk model saham BBRI. Hasil dari pemodelan ini menunjukkan tingkat akurasi yang baik dengan nilai Mean Absolute Error (MAE) masing-masing sebesar 0,01444 untuk saham BTPN dan 0,01406 untuk saham BBRI [7].

Alrafiful Rahman dkk dalam penelitiannya menganalisis data historis pergerakan harga saham harian BNI selama periode 16 April 2001 hingga 6 Januari 2023. Fokus utama penelitian ini adalah pada harga penutupan saham di mana data dibagi menjadi dua bagian, seperti 80% digunakan untuk pelatihan model dan 20% untuk pengujian. Model LSTM diterapkan dengan optimasi Adam dalam beberapa konfigurasi epoch, yaitu pada 25, 50, 75, dan 100. Hasil terbaik diperoleh saat model dijalankan dengan 100 epoch dan ukuran batch 64 yang menghasilkan nilai koefisien determinasi (R2) sebesar 0,9928 dan MAPE sebesar 1,53% [8].

Dalam penelitian yang dilakukan oleh Edwin Setiawan Nugraha dkk dengan memanfaatkan data harga penutupan saham PT Astra International Tbk yang diambil dari situs Yahoo Finance dalam periode antara 2 Januari 2015 hingga 30 Desember 2020. Proses awal pengolahan data mencakup pembersihan data, pembagian dataset menjadi 80% untuk pelatihan dan 20% untuk pengujian, serta normalisasi menggunakan metode Min-Max Scaling. Proses pemodelan dilakukan dengan bantuan algoritma optimasi Adam dan menggunakan fungsi *mean squared error*. Sebanyak dua belas model dievaluasi dengan berbagai kombinasi hyperparameter termasuk variasi jumlah epoch (50, 100, 200, dan 400) dan ukuran batch (4, 16, dan 64). Dari seluruh model yang diuji, model dengan konfigurasi batch size 4 dan 50 epoch terbukti paling optimal. Performa model tersebut diukur menggunakan metrik RMSE sebesar 151.910, MSE sebesar 23.076.561, MAE sebesar 118.128, dan MAPE sebesar 2,3% [9].

Tatas Handharu Sworo dan Arief Hermawan melakukan penelitian yang berfokus pada prediksi harga saham BRI dengan memanfaatkan data historis harga penutupan dari Januari 2006 hingga Desember 2023. Model LSTM digunakan untuk mempelajari pola pergerakan harga saham dari data historis tersebut. Setelah proses pelatihan selesai, performa model dievaluasi menggunakan sejumlah metrik evaluasi dilakukan dan hasil pengujian model memiliki kinerja yang baik yang ditunjukkan dengan nilai MSE sebesar 0,000279, MAE sebesar 0,0133, dan RMSE sebesar 0,0167 pada data pelatihan [10].

Ni Putu Noviyanti Kusuma dalam penelitiannya memanfaatkan data harga penutupan saham yang telah disesuaikan dari 17 emiten *blue chip* yang tergabung dalam indeks IDX30 pada periode 2019 hingga 2022. Penerapan teknik normalisasi min-max scaling digunakan untuk menyesuaikan skala data. Setelah melalui proses preprocessing, data dibagi menjadi dua bagian, yakni 80% untuk pelatihan dan 20% untuk pengujian. Model prediksi kemudian dikembangkan menggunakan algoritma LSTM. Akurasi hasil prediksi dievaluasi menggunakan metode MAPE dengan hasil rata-rata sebesar 2,6% yang menandakan bahwa model ini memiliki performa yang sangat baik dalam memprediksi harga saham [11].

Arnes Anandita dan Tri Wahyuningsih dalam penelitiannya memanfaatkan data harga penutupan dari indeks saham syariah JII70 yang dikumpulkan selama periode 1 Januari 2020 hingga 30 Januari 2024. Data ini kemudian dipisahkan menjadi dua bagian, yaitu 90% digunakan untuk pelatihan dan 10% untuk pengujian. Sebelum pelatihan dimulai, data distandarisasi untuk meningkatkan stabilitas dan akurasi model. Model prediksi yang dibangun menggunakan arsitektur LSTM dengan konfigurasi satu masukan dan satu keluaran serta dilengkapi dengan 50 unit pada hidden layer. Proses pelatihan berlangsung selama 500 epoch dengan menggunakan optimisasi Adam serta menerapkan batas gradien sebesar 1 dan learning rate sebesar 0,005. Akurasi model diuji menggunakan metrik RMSE dan hasilnya menunjukkan nilai RMSE sebesar 6,8993 yang menandakan bahwa model mampu memprediksi pergerakan indeks saham syariah secara cukup akurat [12].

Abdul Rosyd dkk menggunakan data historis saham BCA yang diambil dari situs Yahoo Finance dengan mencakup periode 1 Januari 2020 hingga 30 Oktober 2023. Proses awal dimulai dengan tahap preprocessingdata dengan fokus analisis deret waktu ditujukan pada feature Low. Tahapan berikutnya adalah pembagian 80% data sebagai data latih dan normalisasi data menggunakan metode Min-Max Scaling. Setelah transformasi selesai, maka dilakukan tahap pemodelan data dengan membangun arsitektur model LSTM Sequential yang terdiri dari dua lapisan LSTM masing-masing berisi 50 neuron serta dua lapisan dense dengan 25 dan 1 neuron. Proses uji model menggunakan optimasi Adam dan *mean squared error* sebagai fungsi loss model. Selama pelatihan, model disesuaikan dengan iterasi terhadap data latih, pengaturan jumlah epoch dan batch size, serta penyesuaian bobot berdasarkan hasil prediksi. Model yang telah dilatih kemudian digunakan untuk melakukan prediksi terhadap data baru dan performanya dievaluasi menggunakan metrik RMSE, MAPE, dan MSE. Hasil evaluasi menunjukkan bahwa model ini bekerja dengan baik dengan nilai RMSE sebesar 40,85, MAPE sebesar 0,71%, dan MSE sebesar 6.662,76 [13].

Penelitian yang dilakukan oleh Zhenglin Li dkk menggunakan data historis harga penutupan saham dari Apple, Google, Microsoft, dan Amazon yang diambil dari Yahoo Finance sejak tanggal 1 Januari 2012. Data ini kemudian dibagi menjadi dua bagian, seperti 95% digunakan untuk melatih model dan 5% sisanya untuk menguji performanya. Model yang digunakan berbasis LSTM dengan arsitektur dua lapisan LSTM yang masing-masing memiliki 128 dan 64 unit serta dilanjutkan dengan dua lapisan dense berukuran 25 dan 1 unit. Model ini dibangun dengan menggunakan algoritma optimisasi Adam dan fungsi lossMSE. Proses pelatihan dijalankan selama satu epoch dengan ukuran batch sebesar satu. Untuk mengevaluasi hasil prediksi model pada data pengujian menggunakan metrik RMSE yang menghasilkan nilai 18,89 yang mencerminkan tingkat akurasi data uji model dalam memprediksi harga saham [14].

Penelitian yang dilakukan oleh Eka Patriya dkk menggunakan data harga penutupan IHSG dari 10 Juli 2008 hingga 1 Maret 2023. Sebelum dimasukkan ke dalam model LSTM, data menjalani proses preprocessing seperti normalisasi, penentuan jumlah sampel berdasarkan time step sebesar 50, serta pembagian data menjadi 70% untuk pelatihan dan 30% untuk pengujian. Model LSTM dibangun dengan pendekatan sequential yang terdiri atas beberapa lapisan LSTM yang dilanjutkan dengan dense layer dan output layer untuk menghasilkan satu nilai prediksi. Evaluasi model dilakukan dengan menggunakan metrik MSE, sedangkan proses pelatihannya mengandalkan algoritma optimasi Adam. Model diuji dengan berbagai konfigurasi hyperparameter, seperti jumlah epoch (30, 35, 40, 45, dan 50) dan ukuran batch sebesar 32. Selain itu, performa model juga dianalisis melalui beberapa rasio pembagian data, yaitu 7:3, 8:2, dan 9:1 dengan pengukuran kinerja menggunakan RMSE. Hasil terbaik diperoleh saat model dilatih dengan 50 epoch dan rasio data 7:3 yang menghasilkan nilai RMSE sebesar 63,672 [15].

Dalam penelitian yang dilakukan oleh Juan Syidi Prasetyo dkk, data yang digunakan berasal dari harga penutupan harian saham ANTM dan MDKA diambil dari situs Yahoo Finance dengan periode waktu sejak 1 Januari 2017. Model prediksi LSTM terdiri dari beberapa lapisan yang diawali dengan lapisan pertama berisi 50 neuron dan diikuti oleh lapisan kedua serta dilanjutkan dengan lapisan dense ketiga yang memiliki 25 neuron. Lapisan keluaran model terdiri dari satu neuron untuk menghasilkan prediksi akhir. Model ini menggunakan parameter optimasi Adam. Dari hasil evaluasi kinerja model diperoleh nilai MAPE sebesar 3,08% untuk saham MDKA dan 2,17% untuk saham ANTM yang menunjukkan tingkat kesalahan prediksi yang relatif rendah [16].

I Ketut Agung Enriko dkk dalam penelitiannya menggunakan data saham TLKM dengan periode waktu dari 1 Januari 2019 hingga 11 Januari 2023. Data tersebut diperoleh melalui platform Yahoo Finance. Fokus utama dari penelitian ini adalah harga saham pembukaan. Dataset dibagi dengan proporsi 75% untuk pelatihan dan 25% untuk pengujian. Penerapan algoritma LSTM dilakukan dalam beberapa tahap yang mencakup penggunaan optimasi Adam, SGD, dan RMSprop dengan variasi jumlah epoch yang berbeda, yakni 25, 50, 75, dan 100. Pengukuran akurasi menggunakan metrik MAPE dan akurasi tertinggi tercatat pada optimasi Adam dengan 100 epoch dengan menghasilkan akurasi sebesar 98,59%. Sebaliknya, akurasi terendah tercatat pada optimasi RMSprop dengan 25 epoch yang hanya mencapai 97,61%. Pada optimasi SGD, akurasi tertinggi tercatat pada 50 epoch, sementara akurasi tertinggi tercatat pada 100 epoch pada optimasi RMSprop. Selanjutnya model menggunakan fungsi loss MSE. Nilai loss terendah ditemukan pada optimasi Adam dan RMSprop dengan 100 epoch yang menghasilkan nilai loss 0,0011. Sementara itu, nilai loss tertinggi sebesar 0,003656 ditemukan pada optimasi SGD dengan 25 epoch [17].

Berdasarkan berbagai penelitian sebelumnya menunjukkan bahwa metode LSTM telah menunjukkan performa yang baik dalam menganalisis dan memprediksi data time series termasuk harga saham. Namun, setiap penelitian memiliki perbedaan dalam hal pendekatan dan objek yang diteliti. Oleh karena itu, penelitian ini berupaya memberikan kontribusi ilmu dan pengetahuan terkait penerapan metode LSTM kepada penulis terhadap penelitian pada saham Hapag-Lloyd AG. Penelitian diharapkan dapat memperluas pemahaman dan memberikan wawasan baru terhadap penerapan LSTM dalam analisis data *time series*.

2.2 Landasan Teori

2.2.1 Saham

Di dalam dunia investasi terdapat banyak instrumen yang dapat dipilih oleh investor perorangan, seperti deposito, valuta asing, reksadana, tanah, logam, dan saham. Semua instrumen tersebut mempunyai kelebihan dan kekurangan pada masing-masing perspektif investor. Tetapi, salah satu instrumen investasi menarik yang dapat dilakukan di berbagai negara ialah saham.

Saham merupakan surat berharga yang menjadi tanda kepemilikan seseorang atau badang terhadap suatu perusahaan. Biasanya perusahaan dapat mengeluarkan dua jenis saham yang terdiri atas saham biasa dan preferen. Saham biasa adalah pemilik asli dari perusahaan yang menanggung risiko dan mendapatkan profit. Pemegang saham biasa mempunyai hak suara dalam RUPS (Rapat Umum Pemegang Saham) dan turut serta adlam menentukan kebijakan perusahaan. Apabila perusahaan dibubarkan, maka pemegang saham biasa akan membagi sisa aset perusahaan setelah dikurangi bagian pemegang saham preferen.

Saham preferen memeroleh hak istimewa dalam pembayaran dividen yang tetap setiap tahunnya dibandingkan saham biasa. Hak dividen pemegang saham preferen akan dikumpulkan apabila perusahaan pada tahun tesebut tidak mampu membagikan dividen. Pemegang saham preferen juga akan mendapatkan pembayaran dari sisa-sisa aset perusahaan sebelum pemegang saham biasa jika perusahaan dibubarkan. Sebagai konsekuensinya, pemegang saham preferen mempunyai hak suara terbatas dalam menentukan kebijakan perusahaan.

Bursa saham terpengaruh pada faktor-faktor ekonomi dan non-ekonomi dari dalam dan luar negeri. Perdagangan saham antara pembeli dan penjual saham hanya dapat dilakukan di bursa efek melalui perusahaan sekuritas. Harga saham dapat bergerak secara dinamis dan tidak ada jaminan apabila suatu perusahaan memiliki laba terkontrol atau pembagian dividen yang besar, maka harga sahamnya akan stabil naik. Adapun karena suatu hal, harga saham pada perusahaan dapat terus-menerus menurun meskipun labanya besar [18].

Investor mempunyai karakteristik dalam menentukan kategori saham untuk memperhitungkan profit yang berpotensi diraih. Kategori *blue chip* merupakan saham perusahaan yang mempunyai reputasi baik, catatan pertumbuhan profit stabil dari tahun ke tahun, dan dapat memberikan dividen secara stabil. Beberapa perusahaan tersebut telah diakui dalam pengelolaan secara profesional dalam menghasilkan produk/jasa berkualitas. Kelompok *growth stock* dapat diartikan kepada perusahaan yang menampilkan pertumbuhan pendapatan dan laba yang lebih tinggi dari rata-rata sektornya. Perusahaan *growth stock* cenderung menggunakan labanya untuk ekspansi dan sedikit bahkan tidak mengalokasikan pada dividen.

Terdapat saham lain yang termasuk dalam *defensive stock* di mana perusahaan dapat bertahan dalam keadaan ekonomi yang tidak stabil. Ketika kondisi perekonomian sudah membaik, perusahaan tersebut dapat mempertahankan labanya. Kategori *cyclical stock* mendapatkan momentum pada musim-musim tertentu. Beberapa perusahaan dapat memiliki musim berbeda yang bergantung pada agenda yang dilakukan oleh beberapa kelompok masyarakat, seperti keluarga yang memiliki anak dalam memperlengkap kebutuhan tahun ajaran baru, persiapan hari-hari raya keagamaan, atau kebijakan terhadap naik atau turunnya suku bunga [19].

Ciri khas yang dimiliki saham biasanya dilakukan berdasarkan sifat kepemilikan atas nama dan untuk investasi jangka panjang. Saham dengan mudah dapat dipindahtangankan melalui proses jual beli pada bursa efek di suatu negara. Potensi keuntungan yang dapat diraih investor saham salah satunya dividen sebagai sejumlah pengembalian yang dilandaskan pada jumlah kepemilikan atas saham perusahaan bagi pemegang saham. Keuntungan saham lainnya dapat diperoleh dari *capital gain*, yaitu hasil penjualan saham dari selisih nilai jual yang lebih tinggi dari nilai belinya. Keunikan pada keuntungan saham juga dapat dijaminkan ke bank sebagai agunan untuk memeroleh kredit.

Berinvestasi saham tidak terhindarkan dari risiko yang bernama *capital loss* sebagai kerugian hasil penjualan dari selisih nilai jual yang lebih rendah dari nilai belinya. Risiko likuidasi dapat terjadi jika perusahaan bubar dan pemegang saham memiliki klaim terakhir terhadap aset perusahaan setelah semua kewajiban perusahaan telah dibayar. Saham juga dapat di-*delisting* dari pencatatan bursa sehingga tidak dapat diperdagangkan lagi.

Pasar pada bursa terbagi menjadi tiga, seperti pasar reguler, pasar negosiasi, dan pasar tunai. Pasar reguler merupakan pasar dengan pembentukan harga saham dilakukan dari tawar menawar terus-menerus secara lelang. Untuk pembentukan harga dengan negosiasi langsung antara sekuritas pembeli dan penjual dilakukan pada pasar negosiasi. Dan pasar tunai terjadi dengan pembentukan harga pada sekuritas yang tidak melunasi kewajiban penyelesaian transaksi di pasar reguler dan negosiasi.

Tempat melakukan pembelian saham dapat dilakukan pada pasar perdana di mana saham suatu perusahaan ditawarkan kepada masyarakat untuk pertama kalinya. Investor hanya bisa membeli saham pada mekanisme yang bernama *Initial Public Offering* (IPO) dan menjualnya di pasar sekunder. Pasar sekunder ialah tempat jual beli saham yang dapat dilakukan di hari perdagangan setelah perusahaan dikatakan tercatat di bursa. Sekuritas akan menjadi jembatan dalam pembelian dan penjualan saham [20].

2.2.2 Hapag-Lloyd AG

Hapag-Lloyd Aktiengesellschaft (AG) beroperasi pada aktivitas pengiriman global dengan menyediakan berbagai jenis kontainer untuk memenuhi berbagai kebutuhan kargo. Pada tahun 2025, Hapag-Lloyd menawarkan armada dengan kapasitas kapal sebanyak 2,4 juta TEU (*Twenty-foot Equivalent Unit*) dan kontainer sebanyak 3,7 juta TEU yang menjadikan salah satu armada kontainer berpendingin terbesar di dunia.

Perjalanan Hapag-Lloyd dimulai dari abad ke-19 pada saat perusahaan pendirinya yang bernama Hamburg-Amerikanische Packetfahrt-Actien-Gesellschaft (Hapag) and North German Lloyd berlayar untuk pertama kalinya membawa kargo umum dan penumpang ke New York. Dengan perubahan yang sangat dinamis, berkembanglah perusahaan pelayaran peti kemas ternama dan aktif seperti saat ini.

“*We care*. *We move*. *We deliver*.” merupakan nilai-nilai perusahaan Hapag-Lloyd AG yang menjadi inti dari semua hal yang perusahaan lakukan. Nilai-nilai ini menceritakan kisah tentang asal-usul, pendefinisian, dan bimbingan dalam perjalanan perusahaan ke depan. Nilai-nilai tersebut juga mengarahkan cara perusahaan dalam berbisnis, inspirasi, motivasi, dan membantu perusahaan menemukan jalan yang benar [21].

2.2.3 Pengembangan Sistem *Prototyping*

2.2.4 *Data Mining*

2.2.5 *Long Short-Term Memory*

2.2.6 *Cross-Industry Standard Process for Data Mining*

2.2.7 *Time Series Forecasting*

2.2.8 Streamlit

BAB III  
METODE PENELITIAN

3.1 Metode *Prototyping*

3.2 Alur Kerja CRISP-DM

3.3 Kebutuhan Sistem

3.4 Perancangan Sistem

3.4.1 Use Case

3.4.2 Activity

3.4.3 Sequence

3.5 Perancangan Antar Muka

BAB IV  
HASIL PENELITIAN DAN PEMBAHASAN

4.1 Implementasi Sistem

4.2 Implementasi Antarmuka

4.3 Pengujian Sistem

BAB V  
PENUTUP

5.1 Kesimpulan

5.2 Saran

DAFTAR PUSTAKA

LAMPIRAN

**Lampiran 1. Riwayat Hidup**

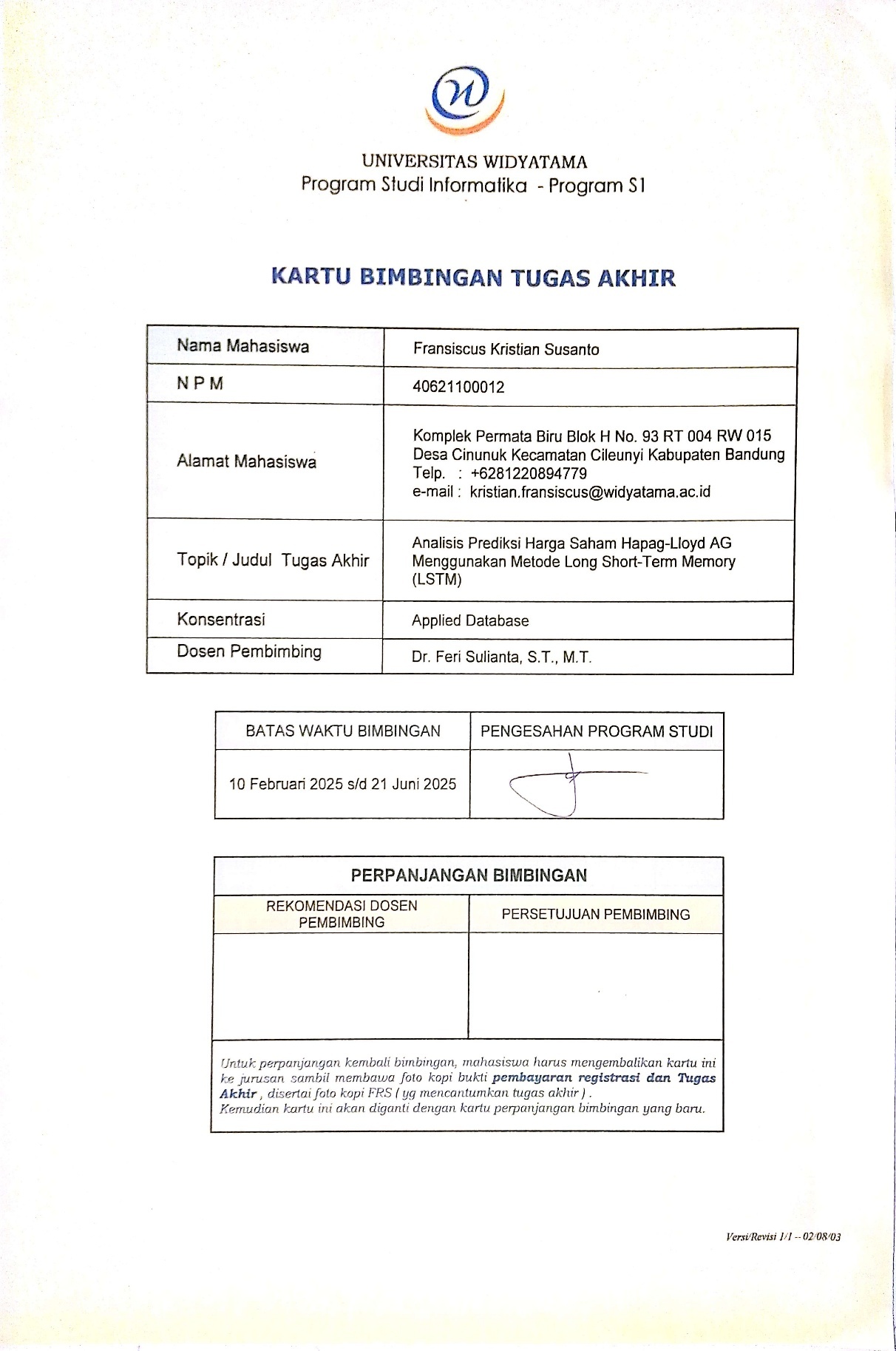
1. Identitas Diri

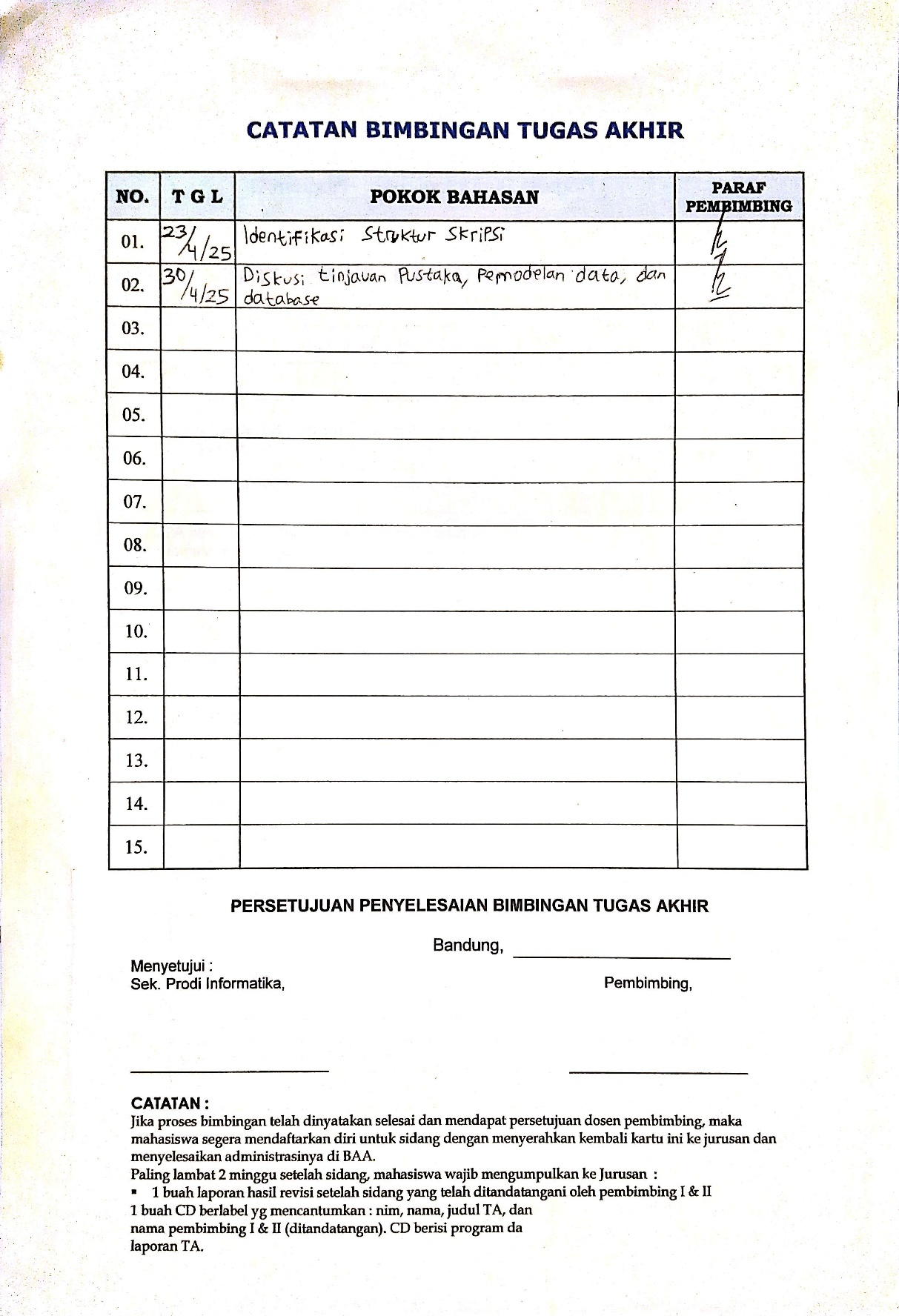
|  |  |  |  |
| --- | --- | --- | --- |
| 1 | Nama Lengkap | : |  |
| 2 | Tempat dan Tanggal Lahir | : |  |
| 3 | Jenis Kelamin | : |  |
| 4 | Program Studi | : |  |
| 5 | NIM | : |  |
| 6 | Alamat | : |  |
| 7 | Kewarganegaraan | : |  |
| 8 | Agama | : |  |
| 9 | E-Mail | : |  |
| 10 | No HP | : |  |

1. Riwayat Pendidikan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **SD** | **SMP** | **SMA** | **S1** |
| Nama Instansi |  |  |  |  |
| Jurusan |  |  |  |  |
| Tahun Masuk - Lulus |  |  |  |  |

**Lampiran 2. Kartu Bimbingan Tugas Akhir**





Lampiran 3. Source code halaman utama

import os

import io

import time

import math

import platform

import datetime

import psutil

import matplotlib

import sklearn

import streamlit as st

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import yfinance as yf

import tensorflow as tf

from sklearn.preprocessing import MinMaxScaler

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_percentage\_error

from utils.train\_predict import get\_frequency\_code, get\_time\_step, create\_dataset, build\_and\_train\_model, show\_prediction\_results, get\_future\_trading\_dates

from utils.model\_utils import save\_info\_model, find\_model\_file, load\_info\_model, delete\_old\_model

from utils.viz\_utils import dataset\_information\_summary, generate\_lstm\_model\_config, model\_architecture\_summary, highlight\_rows

st.set\_page\_config(page\_title="Prediksi Harga Saham", page\_icon="assets/favicon.ico", layout="wide")

st.header("Analisis Prediksi Harga Saham Menggunakan Metode LSTM", divider="gray")

if "logged\_in" not in st.session\_state:

    st.session\_state.logged\_in = False

if "username" not in st.session\_state:

    st.session\_state.username = ""

if st.session\_state.logged\_in:

    st.sidebar.header("Autentikasi")

    st.sidebar.success(f"Nama akun: {st.session\_state.username}")

    is\_admin = True

    if st.sidebar.button("Keluar"):

        st.session\_state.logged\_in = False

        st.session\_state.username = ""

        st.success("Keluar berhasil!")

        st.markdown("""

            <meta http-equiv="refresh" content="3; url=/">

        """, unsafe\_allow\_html=True)

        st.stop()

else:

    is\_admin = False

st.sidebar.header("Pengaturan Model")

ticker = st.sidebar.text\_input(

    "Masukkan Ticker Saham",

    "HLAG.DE",

    help="Ticker adalah kode unik yang digunakan untuk mewakili nama perusahaan pada saham."

)

start\_date = st.sidebar.date\_input("Tanggal Awal", datetime.date(1980, 1, 1))

end\_date = st.sidebar.date\_input("Tanggal Akhir", datetime.date.today())

freq = st.sidebar.selectbox(

    "Frekuensi Data",

    options=["Harian", "Mingguan", "Bulanan"],

    help="Frekuensi data merupakan frekuensi pengambilan data harga saham."

)

model\_option = st.sidebar.radio(

    "Metode Pemodelan",

    ["Latih model baru", "Gunakan model dari database"],

    help="Pemilihan metode `Latih model baru` disarankan untuk prediksi terkini. Default ticker pada metode Gunakan model dari database adalah ticker `HLAG.DE`"

)

tune\_model = st.sidebar.checkbox(

    "Aktifkan Model Tuning",

    value=True,

    help="Mengombinasikan parameter model tuning secara otomatis untuk analisis prediksi yang mendalam."

)

if st.sidebar.button("Mulai Prediksi"):

    st.subheader("0. System and Library Information")

    st.markdown("#### 0.1 Sistem")

    device\_name = "GPU" if tf.config.list\_physical\_devices('GPU') else "CPU"

    gpus = tf.config.list\_physical\_devices('GPU')

    cpu\_info = platform.processor()

    ram\_info = psutil.virtual\_memory()

    components = [

        "Sistem Operasi",

        "Versi Python",

        "Perangkat Komputasi",

        "Total RAM (GB)"

    ]

    info\_values = [

        f"{platform.system()} {platform.release()}",

        platform.python\_version(),

        device\_name,

        round(ram\_info.total / (1024 \*\* 3), 2)

    ]

    if cpu\_info and cpu\_info.strip():

        components.insert(3, "Informasi CPU")

        info\_values.insert(3, cpu\_info)

    if gpus:

        components.insert(3, "Jumlah GPU")

        info\_values.insert(3, str(len(gpus)))

        components.insert(4, "Nama GPU")

        info\_values.insert(4, ", ".join([gpu.name for gpu in gpus]))

    system\_info = {

        "Komponen": components,

        "Informasi": info\_values

    }

    st.dataframe(pd.DataFrame(system\_info))

    st.markdown("#### 0.2 Library")

    env\_info = {

        "Streamlit": st.\_\_version\_\_,

        "TensorFlow": tf.\_\_version\_\_,

        "NumPy": np.\_\_version\_\_,

        "Pandas": pd.\_\_version\_\_,

        "Sklearn": sklearn.\_\_version\_\_,

        "Matplotlib": matplotlib.\_\_version\_\_,

        "Seaborn": sns.\_\_version\_\_

    }

    env\_df = pd.DataFrame(list(env\_info.items()), columns=["Library", "Versi"])

    st.dataframe(env\_df)

    st.subheader("1. Business Understanding")

    st.markdown(

        """

        <div style='text-align: justify'>

            Dalam dunia investasi dan pasar modal, kemampuan untuk memprediksi harga saham secara akurat

            sangat penting bagi pengambilan keputusan yang tepat dan strategis. Oleh karena itu, dibutuhkan

            metode yang mampu menangkap pola waktu (time series) secara efektif. Long Short-Term Memory (LSTM)

            dirancang untuk mengenali pola dalam data berurutan dan memiliki keunggulan dalam mengatasi masalah

            long-term dependencies. Dengan menerapkan LSTM, perusahaan atau investor dapat memperoleh prediksi

            harga saham yang mendukung perencanaan dan manajemen risiko investasi.

        </div>

        """,

        unsafe\_allow\_html=True

    )

    st.subheader("2. Data Understanding")

    with st.spinner("Mengambil data dari Yahoo Finance..."):

        if start\_date >= end\_date:

            st.sidebar.error("Tanggal awal harus lebih kecil dari tanggal akhir.")

            st.stop()

        freq\_code = get\_frequency\_code(freq)

        data = yf.download(ticker, start=start\_date, end=end\_date + pd.Timedelta(days=1), interval=freq\_code, auto\_adjust=True)

        if data.empty:

            st.error(f"Data dengan ticker {ticker} tidak ditemukan atau gagal diunduh.")

        else:

            actual\_start\_date = data.index.min().to\_pydatetime().date()

            actual\_end\_date = data.index.max().to\_pydatetime().date()

            st.write(

                f"Data dengan ticker \*\*{ticker}\*\* dari tanggal "

                f"\*\*{actual\_start\_date.strftime('%d %b, %Y')}\*\* sampai "

                f"\*\*{actual\_end\_date.strftime('%d %b, %Y')}\*\* berhasil diunduh dari Yahoo Finance."

                f"Dataset dapat dilihat pada tautan [ini](https://finance.yahoo.com/quote/{ticker}/history)."

            )

    dataset\_information\_summary(data, "Data mentah", expanded=True)

    st.subheader("3. Data Preparation")

    st.markdown("#### 3.1 Data Transformation")

    with st.expander("Mengganti multiIndex dengan indeks biasa."):

        if isinstance(data.columns, pd.MultiIndex):

            data.columns = data.columns.get\_level\_values(0)

            data.columns.name = None

        st.write(data.head())

    with st.expander("Mengubah indeks Date menjadi feature dan mengonversi tipe datanya ke datetime."):

        data.reset\_index(inplace=True)

        data["Date"] = pd.to\_datetime(data["Date"])

        st.write(data.head())

    st.markdown("#### 3.2 Data Filtering")

    st.write("Mengambil feature Date dan Close.")

    df = data[["Date", "Close"]].copy()

    df["Close"] = df["Close"].round(2)

    dataset\_information\_summary(df, "Data yang difilter", expanded=True)

    st.subheader("3.3 Data Cleaning")

    st.markdown("##### 3.3.1 Missing Values")

    with st.expander("Memeriksa dan menangani nilai yang hilang pada feature Close."):

        missing\_before = df[["Close"]].isnull().sum()

        if df["Close"].isnull().sum() > 0:

            df["Close"] = df["Close"].fillna(method="ffill")

        missing\_after = df[["Close"]].isnull().sum()

        missing\_data = {

            "Feature": ["Close"],

            "Missing Values": missing\_before.values,

            "Keterangan": [

                "Tidak ada missing value" if missing\_before["Close"] == 0 else "Terdapat missing value"

            ]

        }

        if missing\_before["Close"] > 0:

            missing\_data["Missing sesudah Ditangani"] = missing\_after.values

        missing\_table = pd.DataFrame(missing\_data)

        st.write(missing\_table)

    st.markdown("##### 3.3.2 Negative Values")

    with st.expander("Memeriksa nilai yang negatif pada feature Close."):

        negative\_table = pd.DataFrame({

            "Feature": ["Close"],

            "Negative Values": [(df["Close"] < 0).sum()],

            "Keterangan": [

                "Tidak ada nilai negatif" if (df["Close"] < 0).sum() == 0 else "Terdapat nilai negatif"

            ]

        })

        st.write(negative\_table)

    st.markdown("#### 3.4 Exploratory Data Analysis (EDA)")

    st.write("Ringkasan statistik untuk feature Close.")

    st.write(df["Close"].describe())

    df["Year"] = df["Date"].dt.year

    year\_counts = df["Year"].value\_counts().sort\_index()

    st.write("Distribusi jumlah data per tahun.")

    fig, ax = plt.subplots(figsize=(10, 5))

    ax.bar(year\_counts.index.astype(str), year\_counts.values, color="skyblue")

    ax.set\_xlabel("Tahun", fontsize=12)

    ax.set\_ylabel("Jumlah Data", fontsize=12)

    ax.set\_title("Distribusi Jumlah Data per Tahun", fontsize=14)

    ax.grid(axis="y", linestyle="--", alpha=0.7)

    st.pyplot(fig)

    st.write("Grafik harga saham.")

    fig, ax = plt.subplots(figsize=(12, 5))

    ax.plot(df["Date"], df["Close"], color="blue")

    ax.set\_title("Grafik Harga Saham", fontsize=14)

    ax.set\_xlabel("Tanggal", fontsize=12)

    ax.set\_ylabel("Harga", fontsize=12)

    ax.grid(True)

    st.pyplot(fig)

    st.markdown("#### 3.5 Normalization")

    scaler = MinMaxScaler()

    df["Close Scaled"] = scaler.fit\_transform(df[["Close"]])

    df\_normalized = pd.DataFrame(df[["Date", "Close Scaled"]])

    df\_normalized = df\_normalized.rename(columns={"Date": "Tanggal", "Close Scaled": "Close yang Diskalasikan"})

    dataset\_information\_summary(df\_normalized, "Data yang telah dinormalisasi tetapi belum disusun sebagai suatu sequence (urutan)", expanded=False)

    st.markdown("#### 3.6 Data Windowing")

    time\_step = get\_time\_step(freq)

    data\_scaled = df["Close Scaled"].values.reshape(-1, 1)

    X\_seq, y\_seq = create\_dataset(data\_scaled, time\_step)

    df\_seq = pd.DataFrame({

        "X (sequence)": [x.flatten().round(4).tolist() for x in X\_seq],

        "y (target)": y\_seq.flatten().round(4)

    })

    dataset\_information\_summary(df\_seq, "Data yang telah dinormalisasi dan disusun sebagai sequence untuk input (X) dan target (y)", expanded=False)

    st.markdown("#### 3.7 Data Splitting")

    time\_step = get\_time\_step(freq)

    X, y = create\_dataset(data\_scaled, time\_step)

    split = int(len(X)\*0.8)

    X\_train, X\_test = X[:split], X[split:]

    y\_train, y\_test = y[:split], y[split:]

    X\_train = X\_train.reshape(X\_train.shape[0], X\_train.shape[1], 1)

    X\_test = X\_test.reshape(X\_test.shape[0], X\_test.shape[1], 1)

    full\_data = np.concatenate([y\_train, y\_test])

    full\_data\_rescaled = scaler.inverse\_transform(full\_data.reshape(-1, 1))

    plot\_dates = df["Date"].iloc[time\_step:].reset\_index(drop=True)

    fig, ax = plt.subplots(figsize=(12, 5))

    ax.plot(plot\_dates, full\_data\_rescaled, label="Harga", color="blue")

    split\_date = plot\_dates.iloc[split]

    ax.axvline(split\_date, color="red", linestyle="--", label="Train/Test Split")

    ax.fill\_between(plot\_dates[:split], full\_data\_rescaled[:split, 0], color="lightblue", alpha=0.5, label="Train")

    ax.fill\_between(plot\_dates[split:], full\_data\_rescaled[split:, 0], color="orange", alpha=0.5, label="Test")

    ax.set\_title("Pembagian Data Train dan Test")

    ax.set\_xlabel("Tanggal")

    ax.set\_ylabel("Harga")

    ax.legend()

    ax.grid(True)

    st.pyplot(fig)

    df\_X\_and\_y\_train = pd.DataFrame({

        "X\_train (sequence)": X\_train[i].flatten().round(4).tolist(),

        "y\_train": round(float(y\_train[i]), 4)

    } for i in range(len(X\_train)))

    dataset\_information\_summary(df\_X\_and\_y\_train, "Kelompok data X\_train dan y\_train", expanded=False)

    df\_X\_and\_y\_test = pd.DataFrame({

        "X\_test (sequence)": X\_test[i].flatten().round(4).tolist(),

        "y\_test": round(float(y\_test[i]), 4)

    } for i in range(len(X\_test)))

    dataset\_information\_summary(df\_X\_and\_y\_test, "Kelompok data X\_test dan y\_test", expanded=False)

    st.subheader("4. Modeling")

    if model\_option == "Gunakan model dari database":

        model\_path = find\_model\_file(freq, ticker, model\_type="baseline")

        if model\_path:

            st.success("Model baseline telah ditemukan.")

            params = load\_info\_model(freq=freq, info\_type="metadata", ticker=ticker, model\_type="baseline")

            if params is None:

                st.error("Metadata model baseline tidak ditemukan.")

                st.stop()

            else:

                time\_step = params["time\_step"]

                epochs = params["epochs"]

                batch\_size = params["batch\_size"]

                rmse = params["rmse"]

                mape = params["mape"]

                duration = params["duration"]

            X, y = create\_dataset(data\_scaled, time\_step)

            split = int(len(X) \* 0.8)

            X\_train, X\_test = X[:split], X[split:]

            y\_train, y\_test = y[:split], y[split:]

            X\_train = X\_train.reshape(X\_train.shape[0], X\_train.shape[1], 1)

            X\_test = X\_test.reshape(X\_test.shape[0], X\_test.shape[1], 1)

            model = load\_info\_model(info\_type="model", ticker=ticker, model\_path=model\_path)

            start\_time = time.time()

            y\_pred = model.predict(X\_test)

            duration = time.time() - start\_time

            history = None

            time\_step = time\_step

            epochs = epochs

            batch\_size = batch\_size

        else:

            st.error("Model baseline tidak ditemukan.")

            st.stop()

    elif model\_option == "Latih model baru":

        with st.spinner("Melatih model dan melakukan prediksi..."):

            model, history, duration, epochs, batch\_size = build\_and\_train\_model(X\_train, y\_train, X\_test, y\_test, time\_step)

    st.markdown("#### 4.1 Model Selection")

    generate\_lstm\_model\_config(

        model=model,

        time\_step=time\_step,

        epochs=epochs,

        batch\_size=batch\_size,

        title="4.1.1 Model Configuration"

    )

    st.markdown("##### 4.1.2 Model Architecture Summary")

    model\_architecture\_summary(model)

    st.markdown("#### 4.2 Model Training")

    fig, ax = plt.subplots(figsize=(8, 4))

    if history is None:

        model = load\_info\_model(info\_type="model", ticker=ticker, model\_path=model\_path)

        history\_dict = load\_info\_model(info\_type="history", ticker=ticker, model\_path=model\_path)

        history = type("History", (), {"history": history\_dict}) if history\_dict else None

    if history is not None:

        ax.plot(history.history["loss"], label="Training Loss", color="blue")

        if "val\_loss" in history.history:

            ax.plot(history.history["val\_loss"], label="Validation Loss", color="orange")

    else:

        st.info("History tidak ditemukan atau gagal dimuat dari file.")

    ax.set\_title("Visualisasi Training Loss")

    ax.set\_xlabel("Epochs")

    ax.set\_ylabel("Loss")

    ax.grid(True)

    ax.legend()

    st.pyplot(fig)

    y\_pred = model.predict(X\_test)

    y\_pred\_rescaled = scaler.inverse\_transform(y\_pred)

    y\_test\_rescaled = scaler.inverse\_transform(y\_test.reshape(-1, 1))

    st.markdown("#### 4.3 Prediction Results")

    plot\_dates\_test = df["Date"].iloc[time\_step + len(y\_train):]

    df\_result, rmse, mape = show\_prediction\_results(

        y\_true\_rescaled=y\_test\_rescaled,

        y\_pred\_rescaled=y\_pred\_rescaled,

        plot\_dates=plot\_dates\_test,

        title="Hasil Prediksi"

    )

    baseline\_tipe = "Baseline (dari Database)" if model\_option == "Gunakan model dari database" else "Baseline"

    baseline\_results = [{

        "time\_step": time\_step,

        "epochs": epochs,

        "batch\_size": batch\_size,

        "rmse": rmse,

        "mape": mape,

        "model": model,

        "history": history,

        "duration": duration,

        "tipe": baseline\_tipe

    }]

    if model\_option == "Latih model baru":

        if st.session\_state.username == "admin":

            metadata = {

                "time\_step": time\_step,

                "epochs": epochs,

                "batch\_size": batch\_size,

                "rmse": rmse,

                "mape": mape,

                "duration": duration,

            }

            files\_deleted = delete\_old\_model(freq, ticker, model\_type="baseline")

            save\_info\_model(model, freq, ticker=ticker, model\_type="baseline", history=history, metadata=metadata)

            if files\_deleted:

                st.success(f"Model baseline lama berhasil dihapus. Model, histori training, dan metadata parameter baseline yang baru telah berhasil disimpan.")

            else:

                st.success(f"Model, histori training, dan metadata parameter baseline telah berhasil disimpan.")

    if tune\_model:

        st.markdown("#### 4.4 Model Results")

        st.write("Hasil model baseline.")

    else:

        st.subheader("5. Evaluation")

        st.markdown("#### 5.1 Model Evaluation")

        st.write("Hasil evaluasi model.")

    baseline\_tipe = "Baseline (dari Database)" if model\_option == "Gunakan model dari database" else "Baseline"

    baseline\_table = pd.DataFrame([{

        "Tipe Model": baseline\_tipe,

        "Time Step": time\_step,

        "Epoch": epochs,

        "Batch Size": batch\_size,

        "RMSE": round(rmse, 4),

        "MAPE (%)": round(mape \* 100, 2),

        "Durasi (detik)": round(duration, 2)

    }])

    baseline\_table = (

        baseline\_table

        .style

        .format({

            "RMSE": "{:.4f}",

            "MAPE (%)": "{:.2f}",

            "Durasi (detik)": "{:.2f}"

        })

    )

    st.dataframe(baseline\_table)

    if tune\_model:

        st.markdown("#### 4.5 Model Tuning")

        if model\_option == "Gunakan model dari database":

            model\_path = find\_model\_file(freq, ticker, model\_type="best tuning")

            if model\_path:

                st.success("Model best tuning telah ditemukan.")

                params = load\_info\_model(freq=freq, info\_type="metadata", ticker=ticker, model\_type="best tuning")

                if params is None:

                    st.error("Metadata model best tuning tidak ditemukan.")

                    st.stop()

                else:

                    best\_model = load\_info\_model(info\_type="model", ticker=ticker, model\_path=model\_path)

                    time\_step = params["time\_step"]

                    epochs = params["epochs"]

                    batch\_size = params["batch\_size"]

                    rmse = params["rmse"]

                    mape = params["mape"]

                    duration = params["duration"]

                    best = {

                        "model": best\_model,

                        "history": None,

                        "time\_step": time\_step,

                        "epochs": epochs,

                        "batch\_size": batch\_size,

                        "rmse": rmse,

                        "mape": mape,

                        "duration": duration,

                        "tipe": "Tuning (dari Database)"

                    }

                st.markdown("##### 4.5.1 Best Model Tuning (from Database)")

                X, y = create\_dataset(data\_scaled, time\_step)

                split = int(len(X)\*0.8)

                X\_train\_tune, X\_test\_tune = X[:split], X[split:]

                y\_train\_tune, y\_test\_tune = y[:split], y[split:]

                X\_train\_tune = X\_train\_tune.reshape(X\_train\_tune.shape[0], X\_train\_tune.shape[1], 1)

                X\_test\_tune = X\_test\_tune.reshape(X\_test\_tune.shape[0], X\_test\_tune.shape[1], 1)

                best\_model = load\_info\_model(info\_type="model", ticker=ticker, model\_path=model\_path)

                start\_time = time.time()

                y\_pred = best\_model.predict(X\_test\_tune)

                duration = time.time() - start\_time

                y\_pred\_rescaled = scaler.inverse\_transform(y\_pred)

                y\_test\_rescaled = scaler.inverse\_transform(y\_test\_tune.reshape(-1, 1))

                plot\_dates\_best = df["Date"].iloc[time\_step + len(y\_train\_tune):]

                df\_tuning\_result = pd.DataFrame({

                    "Date": plot\_dates\_best.reset\_index(drop=True),

                    "Actual": y\_test\_rescaled.flatten(),

                    "Predicted": y\_pred\_rescaled.flatten()

                })

                rmse\_best = math.sqrt(mean\_squared\_error(df\_tuning\_result["Actual"], df\_tuning\_result["Predicted"]))

                mape\_best = mean\_absolute\_percentage\_error(df\_tuning\_result["Actual"], df\_tuning\_result["Predicted"])

                tuning\_results = []

                tuning\_results.append({

                    "time\_step": time\_step,

                    "epochs": epochs,

                    "batch\_size": batch\_size,

                    "rmse": rmse\_best,

                    "mape": mape\_best,

                    "model": best\_model,

                    "history": None,

                    "duration": duration,

                    "tipe": "Tuning (dari Database)"

                })

                best = tuning\_results[0]

            else:

                st.error("Model best tuning tidak ditemukan.")

                st.stop()

        else:

            st.markdown("##### 4.5.1 Hyperparameter Tuning")

            if freq == "Harian":

                time\_steps = [30, 60, 90]

            elif freq == "Mingguan":

                time\_steps = [4, 8, 12]

            else:

                time\_steps = [12, 24, 36]

            epochs\_list = [50, 75, 100]

            batch\_sizes = [16, 32]

            total\_iter = len(time\_steps) \* len(epochs\_list) \* len(batch\_sizes)

            progress\_bar = st.progress(0)

            status\_text = st.empty()

            tuning\_results = []

            iter\_count = 0

            for ts in time\_steps:

                for ep in epochs\_list:

                    for bs in batch\_sizes:

                        iter\_count += 1

                        status\_text.text(f"Melatih model dengan time step = {ts}, epoch = {ep}, dan batch size = {bs} ({iter\_count}/{total\_iter})")

                        X, y = create\_dataset(data\_scaled, ts)

                        split = int(len(X)\*0.8)

                        X\_train\_tune, X\_test\_tune = X[:split], X[split:]

                        y\_train\_tune, y\_test\_tune = y[:split], y[split:]

                        X\_train\_tune = X\_train\_tune.reshape(X\_train\_tune.shape[0], X\_train\_tune.shape[1], 1)

                        X\_test\_tune = X\_test\_tune.reshape(X\_test\_tune.shape[0], X\_test\_tune.shape[1], 1)

                        model\_temp, history\_temp, duration\_temp, \_, \_ = build\_and\_train\_model(X\_train\_tune, y\_train\_tune, X\_test\_tune, y\_test\_tune, ts, epochs=ep, batch\_size=bs)

                        pred = model\_temp.predict(X\_test\_tune)

                        pred = scaler.inverse\_transform(pred)

                        actual = scaler.inverse\_transform(y\_test\_tune.reshape(-1, 1))

                        rmse = math.sqrt(mean\_squared\_error(actual, pred))

                        mape = mean\_absolute\_percentage\_error(actual, pred)

                        tuning\_results.append({

                            "time\_step": ts,

                            "epochs": ep,

                            "batch\_size": bs,

                            "rmse": rmse,

                            "mape": mape,

                            "model": model\_temp,

                            "history": history\_temp,

                            "duration": duration\_temp,

                            "tipe": "Tuning"

                        })

                        progress\_bar.progress(iter\_count / total\_iter)

        best = sorted(tuning\_results, key=lambda x: x["rmse"])[0]

        if st.session\_state.username == "admin":

            metadata\_best = {

                "time\_step": best["time\_step"],

                "epochs": best["epochs"],

                "batch\_size": best["batch\_size"],

                "rmse": best["rmse"],

                "mape": best["mape"],

                "duration": best["duration"],

            }

            files\_deleted = delete\_old\_model(freq, ticker, model\_type="best tuning")

            save\_info\_model(best["model"], freq, ticker=ticker, model\_type="best tuning", history=best["history"], metadata=metadata\_best)

            if files\_deleted:

                st.success(f"Model best tuning lama berhasil dihapus. Model, histori training, dan metadata parameter best tuning yang baru telah berhasil disimpan.")

            else:

                st.success(f"Model, histori training, dan metadata parameter best tuning telah berhasil disimpan.")

        st.success(f"Model terbaik menggunakan parameter time step = {best['time\_step']}, epoch = {best['epochs']}, dan batch size = {best['batch\_size']} dengan metrik evaluasi RMSE sebesar {best['rmse']:.4f} dan MAPE sebesar {best['mape']:.2%}")

        st.markdown("#### 4.6 Best Model Selection")

        best\_model = best["model"]

        best\_history = best["history"]

        best\_time\_step = best["time\_step"]

        best\_epochs = best["epochs"]

        best\_batch\_size = best["batch\_size"]

        best\_duration = best["duration"]

        generate\_lstm\_model\_config(

            model=best["model"],

            time\_step=best["time\_step"],

            epochs=best["epochs"],

            batch\_size=best["batch\_size"],

            duration=best["duration"],

            title="4.6.1 Best Model Configuration"

        )

        st.markdown("##### 4.6.2 Best Model Architecture Summary")

        model\_architecture\_summary(best\_model)

        st.markdown("#### 4.7 Best Model Training")

        fig, ax = plt.subplots(figsize=(8, 4))

        if best["history"] is None:

            best\_model = load\_info\_model(info\_type="model", ticker=ticker, model\_path=model\_path)

            history\_dict = load\_info\_model(info\_type="history", ticker=ticker, model\_path=model\_path)

            best["history"] = type("History", (), {"history": history\_dict}) if history\_dict else None

        if best["history"] is not None:

            ax.plot(best["history"].history["loss"], label="Training Loss", color="blue")

            if "val\_loss" in best["history"].history:

                ax.plot(best["history"].history["val\_loss"], label="Validation Loss", color="orange")

        else:

            st.info("History tidak ditemukan atau gagal dimuat dari file.")

        ax.set\_title("Visualisasi Training Loss pada Model Terbaik")

        ax.set\_xlabel("Epochs")

        ax.set\_ylabel("Loss")

        ax.grid(True)

        ax.legend()

        st.pyplot(fig)

        st.markdown("#### 4.8 Best Model Prediction Results")

        X, y = create\_dataset(data\_scaled, best["time\_step"])

        split = int(len(X) \* 0.8)

        X\_train\_best\_model, X\_test\_best\_model = X[:split], X[split:]

        y\_train\_best\_model, y\_test\_best\_model = y[:split], y[split:]

        X\_train\_best\_model = X\_train\_best\_model.reshape(X\_train\_best\_model.shape[0], X\_train\_best\_model.shape[1], 1)

        X\_test\_best\_model = X\_test\_best\_model.reshape(X\_test\_best\_model.shape[0], X\_test\_best\_model.shape[1], 1)

        y\_pred\_best = best["model"].predict(X\_test\_best\_model)

        y\_pred\_best\_rescaled = scaler.inverse\_transform(y\_pred\_best)

        y\_test\_best\_model\_rescaled = scaler.inverse\_transform(y\_test\_best\_model.reshape(-1, 1))

        plot\_dates\_best = df["Date"].iloc[best["time\_step"] + len(y\_train\_best\_model):]

        df\_best\_result, rmse\_best, mape\_best = show\_prediction\_results(

            y\_true\_rescaled=y\_test\_best\_model\_rescaled,

            y\_pred\_rescaled=y\_pred\_best\_rescaled,

            plot\_dates=plot\_dates\_best,

            title="Hasil Prediksi Model Terbaik"

        )

        st.markdown("#### 4.9 Best Model Results")

        st.write("Hasil model tuning.")

        best\_tuning\_table = pd.DataFrame([{

            "Tipe Model": "Tuning terbaik",

            "Time Step": best["time\_step"],

            "Epoch": best["epochs"],

            "Batch Size": best["batch\_size"],

            "RMSE": round(rmse\_best, 4),

            "MAPE (%)": round(mape\_best \* 100, 2),

            "Durasi (detik)": round(best['duration'], 2)

        }])

        best\_tuning\_table = (

            best\_tuning\_table

            .style

            .format({

                "RMSE": "{:.4f}",

                "MAPE (%)": "{:.2f}",

                "Durasi (detik)": "{:.2f}"

            })

        )

        st.dataframe(best\_tuning\_table)

        st.subheader("5. Evaluation")

        st.markdown("#### 5.1 Final Model Evaluation")

        st.write("Hasil evaluasi model terbaik berdasarkan RMSE terendah.")

        all\_results = baseline\_results + tuning\_results

        model\_table = pd.DataFrame([{

            "Tipe Model": r["tipe"],

            "Time Step": r["time\_step"],

            "Epoch": r["epochs"],

            "Batch Size": r["batch\_size"],

            "RMSE": round(r["rmse"], 4),

            "MAPE (%)": round(r['mape'] \* 100, 2),

            "Durasi (detik)": round(r['duration'], 2)

        } for r in all\_results])

        model\_table = model\_table.sort\_values(by="RMSE").reset\_index(drop=True)

        min\_rmse\_value = model\_table["RMSE"].min()

        model\_table = (

            model\_table

            .style

            .apply(lambda row: highlight\_rows(row, min\_rmse\_value), axis=1)

            .format({

                "RMSE": "{:.4f}",

                "MAPE (%)": "{:.2f}",

                "Durasi (detik)": "{:.2f}"

            })

        )

        st.dataframe(model\_table, use\_container\_width=True)

    st.markdown("#### 5.2 Forecasting")

    unit = {"Harian": "Hari", "Mingguan": "Minggu", "Bulanan": "Bulan"}[freq]

    st.info(

        "Forecasting ini bersifat estimasi dan tidak merepresentasikan kejadian nyata secara pasti. "

        "Model menghasilkan proyeksi berdasarkan pola historis dan bukan kejadian masa depan yang pasti.\n\n"

        f"Jumlah waktu sebesar \*\*10 {unit.lower()} ke depan\*\* dipilih karena merupakan angka psikologis "

        "umum yang digunakan untuk memberikan gambaran ringkas. Angka ini cukup untuk menunjukkan tren "

        "yang mungkin terjadi."

    )

    n\_future = 10

    future\_model = best["model"] if tune\_model else model

    future\_time\_step = best["time\_step"] if tune\_model else time\_step

    last\_input = data\_scaled[-future\_time\_step:]

    temp\_input = list(last\_input.flatten())

    st.write(f"Model yang digunakan adalah {'hasil tuning' if tune\_model else 'baseline'} untuk memprediksi {n\_future} {unit.lower()} ke depan.")

    progress\_bar = st.progress(0)

    status\_text = st.empty()

    predictions = []

    for i in range(n\_future):

        status\_text.text(f"Memproses iterasi ke-{i+1} dari {n\_future}")

        x\_input = np.array(temp\_input[-future\_time\_step:]).reshape(1, future\_time\_step, 1)

        yhat = future\_model.predict(x\_input, verbose=0)

        temp\_input.append(yhat[0][0])

        predictions.append(yhat[0][0])

        progress\_bar.progress((i + 1) / n\_future)

    start\_date = df["Date"].max()

    future\_dates = get\_future\_trading\_dates(start\_date, n\_future)

    future\_prices = scaler.inverse\_transform(np.array(predictions).reshape(-1, 1))

    df\_future = pd.DataFrame({"Tanggal": future\_dates, "Harga yang Diprediksi": future\_prices.flatten()})

    fig, ax = plt.subplots(figsize=(10, 5))

    ax.plot(df\_future["Tanggal"], df\_future["Harga yang Diprediksi"], marker="o", linestyle="-")

    ax.set\_title(f"Prediksi Harga Saham Masa Depan dalam {n\_future} {unit} ke Depan", fontsize=14)

    ax.set\_xlabel("Tanggal", fontsize=12)

    ax.set\_ylabel("Harga yang Diprediksi", fontsize=12)

    ax.grid(True)

    fig.autofmt\_xdate()

    st.pyplot(fig)

    st.write(df\_future)

else:

    st.info("Silakan isi parameter di sidebar, lalu tekan 'Mulai Prediksi'")