

BAB II

KAJIAN PUSTAKA

A. Kajian Teoritis

1. Saham

Saham dapat didefinisikan sebagai salah satu instrumen di pasar modal yang menjadi tanda bukti penyertaan modal seseorang atau badan usaha pada suatu perusahaan Perseroan Terbatas. Sebagai bukti kepemilikan, investor atau pemegang saham memiliki hak untuk mendapatkan bagian dari keuntungan perusahaan, yang dikenal sebagai dividen, serta berpotensi memperoleh *capital gain*, yaitu keuntungan dari selisih harga jual saham yang lebih tinggi daripada harga belinya (Otoritas Jasa Keuangan, 2023). Dewasa ini, aktivitas transaksi jual beli saham telah mengalami perubahan drastis; jika dahulu dilakukan manual, kini investor dapat melakukannya secara digital melalui aplikasi atau platform yang disediakan perusahaan sekuritas (Selvia, R. & Zaman, B. 2025).

Dalam melakukan transaksi jual beli saham, seorang investor atau *trader* memerlukan dasar analisis yang kuat agar aktivitasnya tidak jatuh ke arah spekulasi atau perjudian. Setidaknya terdapat tiga jenis analisis yang umum digunakan di pasar modal Indonesia, yaitu analisis fundamental yang berfokus pada rasio keuangan perusahaan, analisis teknikal yang menggunakan grafik dan data historis harga, serta analisis bandarmologi yang berfokus pada pergerakan broker atau bandar.

Pemahaman akan analisis ini penting bagi para pelaku pasar, khususnya *trader* pemula, agar dapat memiliki dasar yang kuat dalam mengambil keputusan untuk mencapai tujuan memperoleh keuntungan (Hudaya et al., 2023).

Saham adalah bukti kepemilikan modal pada suatu perusahaan yang memberikan hak atas dividen dan potensi *capital gain*. Saat ini, transaksi saham dilakukan secara digital, dan untuk menghindari spekulasi, investor perlu menggunakan analisis fundamental, teknikal, atau bandarmologi dalam pengambilan keputusan.

2. Flask

Flask adalah sebuah *micro web framework* yang dibangun menggunakan bahasa pemrograman Python (Fahriza, 2024). Flask berfungsi sebagai kerangka kerja (framework) dan antarmuka untuk membangun aplikasi berbasis web (Wijayanto & Susetyo, 2022). Tujuan utama dari Flask adalah untuk membantu pengembang membangun aplikasi web yang terstruktur dan mudah dikelola, serta mempercepat proses pembangunan dengan menyediakan *library* dan kumpulan kode program yang siap digunakan tanpa harus membuatnya dari awal.

Flask tergolong dalam kategori *microframework* karena sifatnya yang ringan dan tidak memerlukan *library* atau *tools* spesifik dalam penggunaannya. Keunggulan ini membuatnya cocok untuk diimplementasikan pada program yang memiliki keterbatasan dari segi

energi dan memori , serta dapat mengurangi penggunaan *resource* memori. Selain itu, Flask dinilai efektif karena fiturnya yang sederhana, fleksibel, dan mudah disesuaikan dengan kebutuhan sistem. Salah satu fitur unggulan dari Flask adalah *template engine* Jinja2, yang menyederhanakan proses pengiriman data dari *back-end* untuk ditampilkan di *front-end* (Walingkas et al., 2023).

Flask merupakan *micro web framework* berbasis Python yang ringan dan efisien untuk membangun aplikasi web. Framework ini menyediakan struktur dasar dan *library* yang siap pakai (seperti *template engine* Jinja2) tanpa memerlukan sumber daya memori yang besar, sehingga mempercepat proses pengembangan.

3. Python

Python adalah bahasa pemrograman tingkat tinggi yang dinamis dan populer karena fokus pada keterbacaan kode, serta sangat mendukung pengembangan kecerdasan buatan melalui *package* seperti Tensorflow dan Keras. Penerapannya kini meluas ke ranah statistik dan ekonometrik, di mana penggunaan pustaka seperti Pandas dan Statsmodels memungkinkan eksplorasi data serta uji statistik yang kemampuannya sebanding dengan perangkat lunak lain seperti R. Selain itu, Python juga mempermudah pemodelan matematika yang kompleks, seperti fungsi dua variabel pada permukaan ruang, dengan memanfaatkan modul Matplotlib dan NumPy untuk menghasilkan

visualisasi grafik yang lebih akurat dan efisien dibandingkan metode manual (Anam et al., 2023).

Sebagai simpulan, Python menawarkan solusi pemrograman dengan sintaks yang sederhana namun sangat ampuh untuk menyelesaikan permasalahan kompleks di berbagai bidang ilmu (Surbakti et al., 2024). Bahasa ini terbukti efektif sebagai alternatif perangkat lunak statistika dalam menyelesaikan kasus seperti *path analysis*, di mana hasil yang diperoleh setara dengan *software* R namun dengan penerapan kode yang lebih simpel. Lebih jauh lagi, integrasi Python dalam pembelajaran memungkinkan pengguna memahami karakteristik matematis secara visual dengan lebih baik sekaligus meningkatkan efisiensi waktu dan tenaga dalam proses perhitungan (Kaestria & Himmah, 2023).

Python merupakan bahasa pemrograman tingkat tinggi yang populer dan dinamis, dirancang agar kode mudah dibaca dan dipahami. Bahasa ini sangat relevan untuk pengembangan kecerdasan buatan (*machine learning*) karena didukung oleh ekosistem *library* yang kuat seperti TensorFlow dan Keras.

4. Agile

Metode Agile adalah pendekatan pengembangan perangkat lunak yang memprioritaskan efisiensi dan responsivitas terhadap perubahan, yang sering diimplementasikan melalui kerangka kerja Scrum dengan siklus pendek bernama *sprint*. Literatur terkini mengklasifikasikan Agile ke dalam dua perspektif utama: *agile-as-a-tool* yang berfokus

pada perbaikan proses teknis melalui teori kontingensi, dan *agile-as-a-culture* yang menekankan pola pikir fleksibel serta perilaku organisasi secara menyeluruh. Secara praktis, pendekatan ini memungkinkan pengembangan sistem menjadi lebih adaptif karena berfokus pada pengiriman fitur-fitur yang memiliki nilai tambah tinggi secara bertahap dalam setiap iterasi, sehingga tim dapat merespons perubahan kebutuhan bisnis dan pengguna dengan cepat (Magistretti & Trabucchi, 2025).

Sebagai simpulan, penerapan metode Agile menawarkan solusi komprehensif untuk mengelola proyek di tengah ketidakpastian dengan menyeimbangkan efisiensi proses dan fleksibilitas strategis (Badiwibowo Atim., 2024). Metode ini terbukti mampu menghasilkan tingkat keberhasilan sistem yang tinggi, bahkan mencapai hasil sempurna dalam pengujian fungsional, karena adanya mekanisme pengujian berkelanjutan dan integrasi umpan balik yang cepat selama pengembangan. Dengan demikian, Agile tidak hanya berfungsi sebagai metode teknis semata, tetapi juga sebagai fondasi budaya yang mendorong kolaborasi tim yang kuat dan inovasi yang berkelanjutan (Susilo & Azimah, 2023).

Agile adalah metode pengembangan perangkat lunak yang mengutamakan efisiensi dan fleksibilitas melalui pendekatan iteratif dan inkremental (seperti Scrum). Metode ini membagi pengembangan ke

dalam siklus pendek (*sprint*) untuk menghasilkan fungsionalitas sistem secara bertahap dan memastikan hasil akhir sesuai kebutuhan pengguna.

5. Hyperparameter Tuning

Hyperparameter tuning merupakan sebuah proses esensial dalam machine learning yang bertujuan untuk menemukan kombinasi nilai hyperparameter terbaik guna menghasilkan model dengan performa optimal (Nugraha & Sasongko, 2022). Proses ini penting karena sebagian besar algoritma machine learning memiliki hyperparameter, dan pengaturan default dari hyperparameter tersebut tidak dapat menjamin kinerja terbaik dari sebuah model (Putri et al., 2023). Kinerja prediktif sebuah model sangat dipengaruhi oleh nilai hyperparameter yang digunakan untuk melatihnya, sehingga optimasi ini menjadi langkah krusial untuk meningkatkan performa model prediksi, terutama pada model deep learning yang memiliki banyak konfigurasi (Utama et al., 2022).

Merupakan proses krusial dalam *machine learning* untuk mencari kombinasi nilai parameter terbaik guna mengoptimalkan performa model. Penyetelan ini diperlukan karena pengaturan *default* algoritma sering kali tidak menghasilkan kinerja prediktif yang maksimal, terutama pada model *deep learning*.

6. Grid Search

Grid Search adalah salah satu teknik optimasi hyperparameter yang populer dan mapan (Ahmad et al., 2022). Metode ini bekerja sebagai pendekatan *brute-force* yang secara sistematis mengevaluasi semua kemungkinan kombinasi nilai hyperparameter dalam ruang pencarian yang telah ditentukan sebelumnya. Dalam praktiknya, sekumpulan nilai ditentukan untuk setiap hyperparameter, kemudian model akan dilatih dan dievaluasi untuk setiap kombinasi yang ada; pada akhirnya, kombinasi yang memberikan performa paling sukses atau optimal akan dipilih. Meskipun pendekatan ini menjamin eksplorasi penuh atas seluruh ruang pencarian dan menawarkan kesederhanaan, biaya komputasinya dapat tumbuh secara eksponensial seiring bertambahnya jumlah hyperparameter, sehingga sering dianggap tidak praktis untuk model yang kompleks (Dimgba & Andreas, 2024).

Sebagai simpulan, penggunaan algoritma *Grid Search* terbukti lebih unggul dalam meningkatkan kinerja dan akurasi model prediksi dibandingkan dengan pendekatan pemilihan parameter secara manual atau *trial and error* (Puteri et al., 2024).

Keunggulan ini dibuktikan melalui studi kasus prediksi harga saham syariah, di mana model BiLSTM yang dioptimasi dengan *Grid Search* mampu menghasilkan nilai *Mean Absolute Percentage Error* (MAPE) terendah sebesar 10,83%, yang dikategorikan sebagai hasil prediksi yang akurat. Kemampuan algoritma ini dalam menjangkau seluruh

kombinasi parameter memberikan dampak signifikan terhadap kemampuan model dalam melakukan generalisasi data dan menghasilkan luaran yang presisi.

7. UML (Unified Modeling Language)

Unified Modeling Language (UML) adalah bahasa spesifikasi standar yang esensial dalam rekayasa perangkat lunak, berfungsi untuk mendokumentasikan, merinci, dan membangun sistem, terutama yang berorientasi objek (Fitriani & Puspitaningrum, 2023). Sebagai alat komunikasi visual, UML menyediakan notasi dan beragam diagram standar seperti *Use Case*, *Activity*, *Sequence*, dan *Class Diagram* yang memfasilitasi pengembang dalam menganalisis dan merancang kebutuhan sistem, pengguna, serta struktur basis data aplikasi (Akbar et al., 2023).

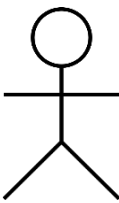
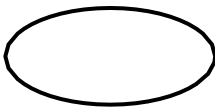
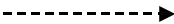
Penggunaan UML secara efektif membantu memastikan pemahaman yang seragam dan mempermudah seluruh tahapan implementasi serta pengembangan aplikasi perangkat lunak. Berikut adalah beberapa diagram pada UML :

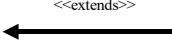
a) *Use Case Diagram*

Use Case Diagram adalah diagram *UML* yang menggambarkan fungsionalitas yang diharapkan dari sebuah sistem dan merepresentasikan alur kerjanya (Komara et al., 2025). Diagram ini secara visual merepresentasikan interaksi antara pengguna

(aktor) dengan sistem untuk mencapai tujuan tertentu, sehingga memberikan gambaran umum tingkat tinggi mengenai fungsi sistem dari perspektif pengguna. *Use Case Diagram* terdiri dari aktor dan interaksi, yang menjelaskan interaksi tipikal antara pengguna sistem dan prosedur terpisah melalui sebuah cerita bagaimana sistem tersebut digunakan (Fajar et al., 2023). Simbol-simbol *Use Case* dapat dilihat pada Tabel 2.1.

Tabel 2.1 Simbol-simbol Use Case Diagram

Simbol	Nama	Deskripsi
	Aktor	Merepresentasikan pengguna atau sistem eksternal yang berinteraksi dengan sistem yang sedang dimodelkan.
	<i>Use Case</i>	Merepresentasikan fungsionalitas atau tindakan yang dilakukan sistem dari sudut pandang pengguna.
	Asosiasi	Menghubungkan aktor dengan use case yang dilakukannya. Menunjukkan interaksi antara aktor dan sistem.
	Generalisasi	Menunjukkan spesialisasi aktor untuk mendapatkan partisipasi dengan use case.
	<i>Include</i>	Menunjukkan bahwa <i>use case</i> menyertakan fungsi dari <i>use case</i> lain. <i>Use case</i> yang di-

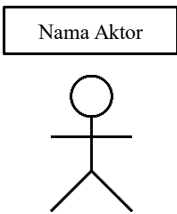
		include harus selesai agar use case yang menyertakannya dapat selesai.
Simbol	Nama	Deskripsi
	<i>Extend</i>	Menunjukkan bahwa suatu use case memperluas atau menambahkan fungsionalitas pada use case lain. Use case yang di-extend bersifat opsional dan dapat terjadi dalam kondisi tertentu.

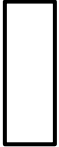
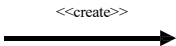
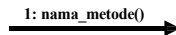
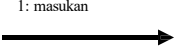
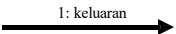
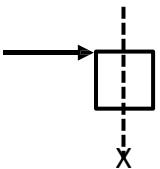
Sumber: (Noviantoro et al., 2022)

b) *Sequence Diagram*

Sequence Diagram merupakan jenis diagram *UML* yang memvisualisasikan interaksi antar objek berbeda dalam sebuah sistem dari waktu ke waktu. Diagram ini fokus pada urutan pertukaran pesan antar objek untuk mencapai tugas atau perilaku tertentu. *Sequence Diagram* mendeskripsikan perilaku objek dalam sebuah *use case* dengan menjelaskan masa hidup objek tersebut serta pesan yang dikirim dan diterima antar objek. Simbol-simbol *Sequence Diagram* dapat dilihat pada Tabel 2.2.

Tabel 2. 2 Simbol-simbol *Sequence Diagram*

Simbol	Nama	Deskripsi
	<i>Actor</i>	Menunjukkan pengguna atau sistem eksternal yang berinteraksi dengan sistem.

	<i>Life line</i> (Garis hidup)	Menunjukkan keberadaan objek selama interaksi.
	<i>Activation</i>	Menunjukkan periode waktu ketika objek sedang aktif dan melakukan operasi.
	Pesan tipe <i>create</i>	Merupakan pernyataan satu objek membuat objek lain.
	Pesan tipe <i>call</i>	Merupakan pernyataan satu objek memanggil metode atau operasi pada objek lain atau diri sendiri.
	Pesan tipe <i>send</i>	Merupakan pernyataan bahwa objek menjalankan metode atau operasi yang memberi hasil suatu pengembalian atau keluaran ke objek tertentu.
	Pesan tipe <i>return</i>	Merupakan pernyataan bahwa objek menjalankan metode atau operasi yang memberi hasil suatu pengembalian atau keluaran ke objek tertentu.
	Pesan tipe <i>destroy</i>	Merupakan pernyataan bahwa satu objek mengakhiri hidup dari objek lain, jika ada create lebih baik ada destroy.

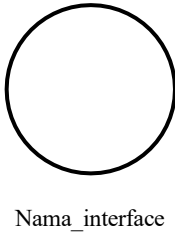
Sumber: (Noviantoro et al., 2022)

c) *Class Diagram*

Class Diagram adalah diagram *UML* yang menjelaskan beberapa kelas dan hubungan antar kelas dalam sebuah sistem. Diagram ini

secara visual merepresentasikan struktur statis dari sebuah sistem dengan menunjukkan kelas-kelasnya, atribut-atributnya, metode-metodenya, serta hubungan antar kelas tersebut. *Class Diagram* merupakan alat fundamental dalam desain berorientasi objek yang menyediakan cetak biru komponen sistem dan koneksinya. Simbol-simbol *Class Diagram* dapat dilihat pada Tabel 2.3.

Tabel 2.3 Simbol-simbol Class Diagram

Simbol	Nama	Deskripsi
Nama_kelas +attribute +operasi	<i>Class</i>	Merupakan kelas yang ada pada struktur sistem. Memiliki atribut dan operasi dalam kelas.
	<i>Interface</i>	Merupakan kemiripan dengan kelas tetapi memiliki metode yang di deklarasikan tanpa isi dan tanpa atribut kelas.
	<i>Association</i>	Merupakan relasi antarkelas (umum), biasanya dilengkapi dengan multiplicity.

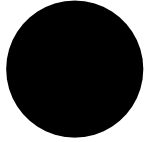
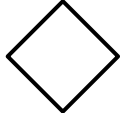
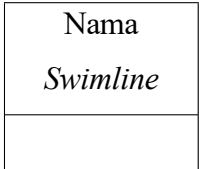
	<i>Directed association</i>	Merupakan relasi antara kelas bermakna satu kelas digunakan oleh kelas yang lain, biasanya dilengkapi dengan multiplicity.
	<i>Generalisasi</i>	Merupakan relasi antara kelas bermakna generalisasi spesialisasi (umum ke khusus).
	<i>Dependency</i>	Merupakan relasi kebergantungan (dependency) antara kelas.
	<i>Aggregation</i>	Merupakan relasi antara kelas bermakna semua-bagian (whole-part).

Sumber: (Noviantoro et al., 2022)

d) *Activity Diagram*

Activity Diagram adalah model analisis yang digunakan untuk mendeskripsikan sebuah proses atau aktivitas dalam sistem. Diagram *UML* ini secara visual merepresentasikan alur kontrol dan aktivitas dalam suatu sistem atau proses. Pada dasarnya, *Activity Diagram* adalah sebuah *flowchart* yang berfokus pada urutan aksi dan keputusan yang terlibat dalam menyelesaikan suatu tugas atau proses. Simbol-simbol *Activity Diagram* dapat dilihat pada Tabel 2.4.

Tabel 2. 4 Simbol-simbol *Activity Diagram*

Simbol	Nama	Deskripsi
	Status awal/akhir	Merupakan status awal atau akhir keadaan dari sistem, setiap diagram aktivitas memiliki satu status awal.
Simbol	Nama	Deskripsi
	Aktivitas	Merupakan kegiatan yang dilakukan sistem, sering dimulai dengan kata kerja.
	<i>Decision</i>	Merupakan hubungan percabangan untuk keputusan aktivitas yang memiliki lebih dari satu pilihan.
	<i>Join</i>	Merupakan hubungan penggabungan jika satu atau lebih aktivitas menjadi satu.
	<i>Swimlane</i>	Merupakan yang memisahkan organisasi bisnis. Memiliki tanggung jawab untuk aktivitas yang terjadi.

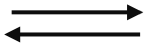
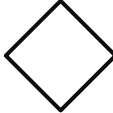
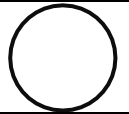
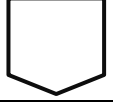
Sumber: (Noviantoro et al., 2022)

8. Flowchart

Flowchart adalah alat pengorganisasian visual yang terstruktur dan sekuensial, dirancang untuk memecah proses kompleks menjadi langkah-langkah logis yang mudah dikelola (Pradana et al., 2024). Alat ini berfungsi sebagai panduan sistematis, misalnya dalam proses menulis, dengan membantu pengguna untuk bertukar pikiran, dan mengatur ide-ide secara efektif dari awal hingga akhir. Dengan

menggabungkan manfaat pembelajaran visual dan pengembangan keterampilan yang sistematis, *flowchart* dapat meningkatkan pemahaman, integrasi pengetahuan, serta mempermudah pengerjaan materi yang rumit (Zhang et al., 2023).

Tabel 2. 5 Simbol-simbol Flowchart

Simbol	Nama	Deskripsi
	<i>Flow Direction</i>	Untuk menunjukkan garis alir proses dan menghubungkan simbol satu dengan simbol lain.
	<i>Terminator</i>	Untuk permulaan (<i>start</i>) dan akhir (<i>stop</i>) dari suatu proses.
	<i>Input/Output</i>	Untuk menggambarkan proses masukan (<i>input</i>) dan keluaran (<i>output</i>) pada sistem.
	<i>Process</i>	Untuk menggambarkan proses pengolahan yang ada pada sistem.
	<i>Disk Storage</i>	Untuk menyatakan masukan dan keluaran yang berasal dari disk.
 	<i>Decision</i>	Untuk memilih proses atau keputusan berdasarkan kondisi yang ada.
	<i>Connector on page</i>	Untuk menghubungkan atau menyambungkan proses dalam satu lembar kerja yang sama.
	<i>Connector off page</i>	Untuk menghubungkan atau menyambungkan proses pada lembar kerja yang berbeda.
	<i>Document</i>	Untuk input berasal dari dokumen dalam bentuk kertas atau output yang perlu dicetak diatas kertas.
	<i>Preparation</i>	Untuk mempersiapkan penyimpanan di dalam storege.

9. Black Box Testing

Blackbox testing adalah metodologi pengujian perangkat lunak di mana program yang dikembangkan dijalankan dan dilakukan upaya untuk memasukkan data melalui setiap antarmuka tanpa perlu mengetahui struktur internal atau cara kerja kode di dalamnya (Apriyanti & Bernanda, 2023). Tujuan utama dari pengujian ini adalah untuk memeriksa secara ketat perilaku program dan memastikan bahwa semua halaman dan perintah beroperasi sesuai dengan rancangan yang diusulkan. Pengujian blackbox berperan penting dalam memastikan kemampuan sistem untuk mengenali dan menyelesaikan masalah, menjamin fungsionalitas sistem yang mulus, serta menghilangkan potensi anomali atau kesalahan seperti perbedaan fungsional dan masalah kinerja sistem (Hafidz & Ikhwan, 2024).

Sebagai simpulan, pengujian *black box* menawarkan solusi praktis dan efisien untuk memvalidasi keandalan sistem dengan memastikan bahwa semua perintah beroperasi sesuai rancangan tanpa mempedulikan kompleksitas internalnya (Aghababaeian et al., 2023).

Efektivitas metode ini semakin terbukti dalam sistem cerdas, di mana pendekatan *black box* berbasis diversitas mampu mengungguli kriteria cakupan *white box* dalam hal kemampuan mendeteksi kesalahan (*fault-revealing capability*) dan efisiensi waktu komputasi. Hal ini menjadikan metode ini sangat relevan untuk diterapkan pada

skenario di mana informasi internal model terbatas atau ketika efisiensi sumber daya menjadi prioritas utama

10. Website

Website merupakan sekumpulan halaman digital yang memuat beragam informasi seperti teks, gambar, dan video yang saling terintegrasi melalui jaringan internet (Kurniasari, 2023). Keberadaan website memungkinkan pengguna untuk mengakses data secara cepat melalui perangkat peramban sehingga dapat menunjang efisiensi dalam penyampaian maupun pelayanan informasi (To Suli, 2023).

Sebagai simpulan, website adalah platform digital yang berfungsi sebagai antarmuka interaktif untuk menyajikan data secara visual bagi pengguna. Dalam penelitian ini, website digunakan sebagai media utama bagi investor untuk mengunggah data historis, menginisiasi pelatihan model GRU, serta memantau hasil klasifikasi arah tren harga saham secara dinamis (To Suli, 2023).

B. Kajian Empiris

Berikut ini merupakan penelitian yang relevan untuk dijadikan acuan dalam penelitian ini:

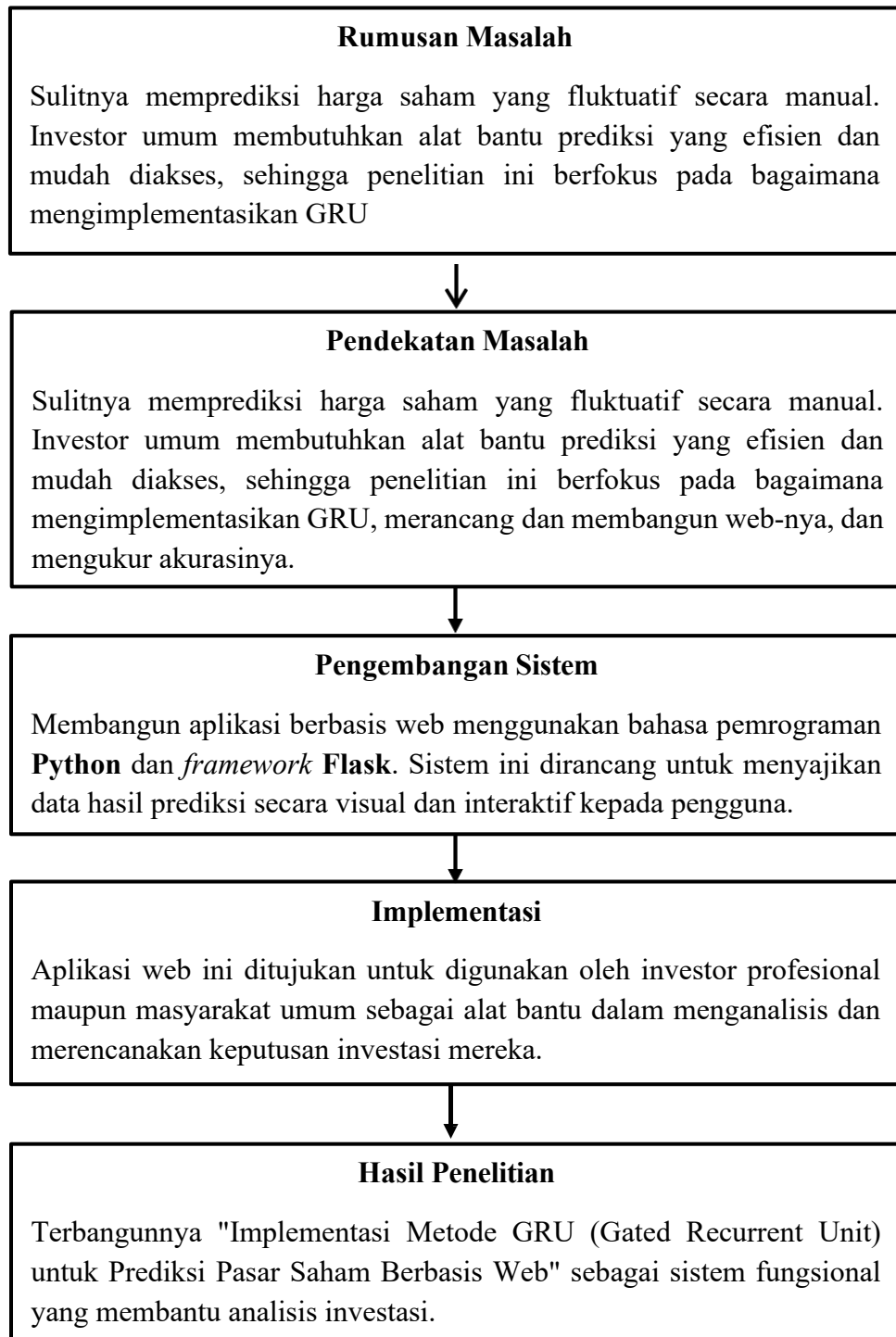
1. Penelitian yang dilakukan oleh Hudaya et al. (2023) berfokus pada pelatihan analisis teknikal dan pengenalan analisis bandarmologi untuk *trader* saham. Penelitian tersebut mengidentifikasi analisis teknikal dan bandarmologi sebagai alat utama untuk *trading*.

Perbedaan mendasarnya adalah, penelitian ini tidak menggunakan analisis teknikal tradisional, melainkan mengimplementasikan model *deep learning* spesifik yaitu Gated Recurrent Unit (GRU) untuk melakukan prediksi data deret waktu harga saham.

2. Penelitian oleh Selvia, R. & Zaman, B. (2025) mengkaji faktor-faktor yang memengaruhi *minat* berinvestasi di pasar modal di kalangan mahasiswa, seperti edukasi, pengetahuan, dan kemampuan finansial. Penelitian tersebut berfokus pada aspek perilaku investor. Sedangkan, penelitian ini berfokus pada pengembangan teknis sebuah sistem, yaitu merancang dan mengimplementasikan aplikasi berbasis web yang menggunakan metode GRU untuk memprediksi harga saham secara langsung.
3. Penelitian yang dilakukan oleh Ahmad et al. (2022) menerapkan teknik optimasi *hyperparameter* GridSearchCV untuk meningkatkan akurasi berbagai model *machine learning* (seperti LR, KNN, dan GBC) dalam prediksi penyakit jantung. Meskipun sama-sama membahas optimasi model, penelitian ini memiliki fokus domain yang berbeda, yaitu prediksi pasar saham, dan menerapkan model *deep learning* Gated Recurrent Unit (GRU), bukan model *machine learning* klasik yang digunakan dalam penelitian tersebut.

C. Kerangka Berfikir

Berdasarkan latar belakang dan kajian teori diatas maka kerangka berfikir penelitian dapat dilihat pada gambar 2.1 sebagai berikut:



Gambar 2. 1 Kerangka berfikir