

BAB II

LANDASAN TEORI

2.1 Teori yang digunakan

2.1.1 Saham

Saham adalah salah satu instrumen pasar keuangan yang paling populer. Menerbitkan saham adalah salah satu opsi perusahaan saat memutuskan untuk pendanaan perusahaan. Di sisi lain, saham merupakan sarana investasi yang banyak dipilih karena dapat memberikan keuntungan menarik [13].

Saham dapat diartikan sebagai penyertaan modal seseorang atau pihak (badan perusahaan) dalam suatu perusahaan atau perseroan terbatas (PT). Dengan penyertaan modal tersebut, maka pihak tersebut mempunyai hak atas penghasilan perusahaan, klaim atas aset dan hak untuk ikut serta dalam Rapat Umum Pemegang Saham (RUPS) [14].

Berikut merupakan keuntungan dan risiko dari saham [13] :

Keuntungan :

- *Capital Gain*

Capital gain merupakan suatu kondisi dimana pemegang saham dapat menjual dengan harga lebih tinggi dari harga beli. Contohnya ketika seseorang membeli saham A dengan harga per saham Rp 2.000 dan menjual saham A tersebut dengan harga Rp 2.500 per saham, sehingga dapat dilihat bahwa pemegang saham mendapat capital gain sebesar Rp 500 untuk setiap saham yang dijual [13].

- Dividen

Dividen adalah pembagian keuntungan yang diberikan perusahaan dan berasal dari keuntungan yang diperoleh perusahaan. Pemegang saham bisa mendapatkan dividen ketika orang tersebut telah memegang saham tersebut dalam kurun waktu yang relatif lama yaitu hingga kepemilikan saham tersebut berada pada periode dimana diakui sebagai pemegang saham yang berhak mendapatkan dividen. Dividen akan diberikan setelah mendapatkan persetujuan dari pemegang saham dalam RUPS [13].

Risiko :

- Likuidasi

Ketika seseorang membeli saham dari perusahaan tertentu dan perusahaan tersebut dinyatakan bangkrut oleh pengadilan atau dibubarkan. hak klaim pemegang saham merupakan prioritas terakhir setelah seluruh kewajiban dari perusahaan telah dilunasi (dari hasil penjualan kekayaan perusahaan. Jika masih terdapat sisa dari hasil penjualan tersebut, maka sisa tersebut akan dibagi secara proporsional kepada semua pemegang saham perusahaan tersebut. Tetapi jika tidak ada sisa kekayaan perusahaan, maka pemegang saham tidak akan mendapat hasil likuidasi tersebut [13].

- *Capital Loss*

Capital Loss Merupakan kebalikan dari *Capital Gain*, kondisi tersebut dapat terjadi dimana pemegang saham menjual saham lebih rendah dari harga beli. Sebagai contoh ketika seseorang membeli saham B dengan harga Rp 3.000 per saham dan menjual saham B dengan harga Rp 2.200 per saham. dikarenakan pemegang saham tersebut tidak ingin mengambil

resiko harga saham yang terus turun maka saham dijual dengan harga Rp 2.200 tersebut, sehingga mengalami kerugian sebesar Rp 800 per saham [13] .

2.1.2 *Market Capitalization*

Market capitalization adalah nilai pasar dari saham perusahaan yang diperdagangkan secara publik, yang didapat dari mengalikan harga saham saat ini dengan jumlah saham yang beredar. *Market capitalization* dapat menunjukkan ukuran, potensi, dan risiko perusahaan di pasar saham, serta berguna untuk membandingkan perusahaan dalam industri atau sektor yang sama. *Market capitalization* tidak mencerminkan nilai aset perusahaan, tetapi tergantung pada jumlah saham dan harga saham yang berfluktuasi [15].

2.1.3 *Website*

Website merupakan sebuah media yang mempunyai banyak halaman yang saling terhubung satu sama lain (*hyperlink*), website berfungsi dalam memberikan informasi berupa teks, gambar, suara, video, dan animasi ataupun gabungan dari semuanya. Website memiliki karakteristik utama halaman yang saling terhubung, dan dilengkapi dengan domain yang berfungsi sebagai alamat (url) atau *world wide web* (www) dan juga hosting sebagai media yang dapat menyimpan banyak data. *Website* dapat diakses menggunakan jaringan internet melalui browser, seperti chrome, opera, mozilla firefox, dan sebagainya [16].

Website dapat dibangun menggunakan *localhost*, sehingga *website* dapat dirancang, dibangun, dan dimodifikasi tanpa harus menggunakan jaringan internet. Terdapat beberapa aplikasi yang dibutuhkan dalam pembangunan sebuah website hingga publikasi ke internet, diantaranya adalah *database* (MySQL, Oracle, dan lain - lain), dan lain – lain. *Website* dapat dibangun menggunakan bahasa

pemrograman seperti PHP, HTML, Javascript, CSS, Python, dan lain – lain [16].

Website terbagi menjadi dua macam yaitu [17]:

1. Web Statis

Web statis adalah sebuah *website* yang Ketika ingin merubah konten pada *website* tersebut, maka *source code website* tersebut harus di edit secara manual. Data pada *website* statis masih belum tersimpan dalam database. *Website* statis dibangun menggunakan tag HTML [17].

2. Web Dinamis

Web dinamis merupakan sebuah *website* yang sangat mudah di ubah konten di dalamnya tanpa harus mengubah *source code* yang ada, hal tersebut dapat dilakukan karena konten *website* tersimpan di dalam database. *Website* Dinamis biasanya dibangun tidak hanya memakai HTML tetapi dibangun juga dengan bahasa pemrograman *server side* seperti PHP, ASP, JSP, dan Lainnya [17].

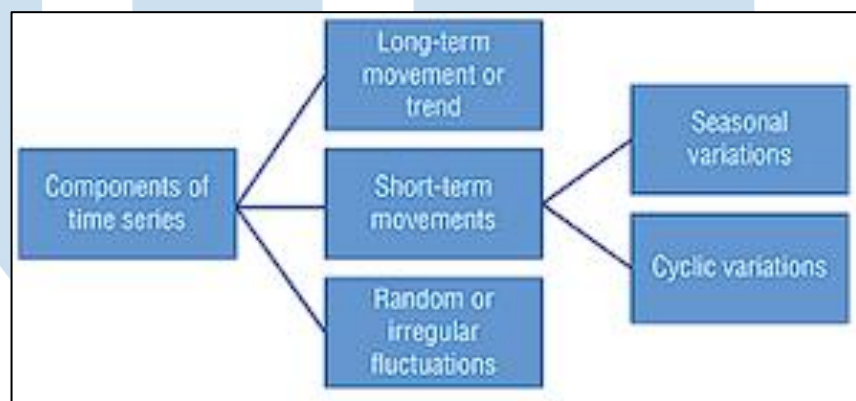
2.1.4 Data time series

Data *time series* merupakan tipe data yang mengukur bagaimana suatu data berubah dari waktu ke waktu. Dalam dataset *time series*, kolom waktu bukan mewakili sebuah variabel, namun kolom waktu adalah struktur utama yang dapat digunakan untuk mengurutkan kumpulan data. Struktur *temporal* tersebut membuat masalah *time series* lebih menantang karena para *data scientist* perlu menerapkan Teknik pengolahan data tertentu dan melakukan *feature engineering* untuk mengatasi data *time series* tersebut [18].

Data *time series* dapat digunakan dalam membuat sebuah prediksi. Prediksi menggunakan data *time series* melibatkan penggunaan model *machine learning*, model dilatih dengan

historical time series data, dan hasil latih akan digunakan untuk memprediksi hasil yang akan terjadi di masa depan. Data *historical* yang berbeda serta fenomena yang sedang terjadi dapat mempengaruhi hasil prediksi [18].

Dapat dilihat pada Gambar 2.1 terdapat 4 komponen kategori utama dari data *time series*: *long-terms movement or trend*, *seasonal short term movements*, *cyclic short term movements*, dan *random or irregular fluctuations* [18].



Gambar 2.1 Komponen data *time series*

Sumber : [18]

Berikut penjelasan mengenai 4 komponen tersebut

- *Long-Term movement or trend*

Long-Term movement or trend mengacu pada keseluruhan pergerakan dari nilai *time series* untuk meningkat atau menurun pada interval waktu yang berkepanjangan. Merupakan hal yang umum jika *trend* berubah arah, tren dapat meningkat, menurun, atau tetap stabil pada momen yang berbeda. Namun jika dilihat secara keseluruhan, terdapat tren yang lebih menonjol daripada tren yang lain. Contoh dari *Long-Term movement or trend* adalah jumlah populasi suatu negara dalam beberapa decade [18].

- *Short-Term movement*

- *Seasonal variations*

Seasonal variations merupakan Perubahan pada data yang dipengaruhi oleh fenomena tertentu yang berkaitan dengan musim. Musim merupakan suatu periode yang diketahui secara pasti. *Seasonal variation* biasanya berkaitan dengan faktor – faktor seperti cuaca, kejadian alam, iklim, atau kegiatan manusia tertentu yang terjadi pada waktu tertentu dalam waktu setahun [18].

- *Cyclical variations*

Cyclical variations merupakan variasi yang terjadi saat suatu data menunjukkan fluktuasi naik, turun, maupun tetap yang tidak memiliki periode yang tetap, meskipun durasi fluktuasi tersebut dapat berlangsung selama lebih dari setahun. Contohnya adalah siklus bisnis, yang mencakup periode naik turun pada *gross domestic product*. Siklus bisnis biasanya terjadi selama beberapa tahun, namun durasi siklus bisnis tidak dapat diketahui secara pasti [18].

- *Random or Irregular fluctuations*

Random or Irregular fluctuations merupakan elemen terakhir yang menyebabkan variasi dalam data *time series*. Fluktuasi ini tidak dapat dikontrol, tidak dapat diprediksi, dan tidak menentu, seperti perang, wabah, gempa bumi, banjir, maupun bencana alam lainnya [18].

2.1.5 *Deep Learning*

Deep Learning merupakan sebuah teknik komputer yang digunakan untuk mengekstrak dan mentransformasi data menggunakan beberapa lapisan (*layers*) *neural networks*. Setiap lapisan (*layers*) tersebut menerima *input* yang diberikan dari lapisan (*layers*) sebelumnya dan secara bertahap menyempurnakannya. Lapisan – lapisan tersebut dilatih menggunakan algoritma yang dapat mengurangi atau meminimalisir error (kesalahan) dan

meningkatkan akurasi. Melalui tahap ini, *network* tersebut dapat belajar untuk menyelesaikan tugas tertentu [19].

Deep Learning memiliki kekuatan, fleksibilitas, dan kemudahan sehingga dapat diaplikasikan dalam berbagai bidang. Bidang–bidang tersebut antara lain meliputi ilmu fisika, ilmu sosial, seni, *finance*, kedokteran, penelitian ilmiah, dan lain–lain. Berikut merupakan beberapa contoh pemanfaatan *deep learning* di berbagai bidang [19]:

- *Natural language processing (NLP)*
NLP dapat menjawab pertanyaan, melakukan *speech recognition*, mengklasifikasi dokumen, dan lain-lain.
- *Computer vision*
Pemanfaatan pada *computer vision* antara lain membaca rambu lalu lintas, *face recognition*, sistem autopilot pada kendaraan.
- *Recommendation systems*
Recommendation systems mencakup *web search*, rekomendasi produk, *home page layout*. Games
- *Financial Forecasting*
Pemanfaatan dalam *Financial Forecasting* dapat dilakukan seperti memprediksi harga saham menggunakan data *time series* seperti yang akan dibahas pada penelitian ini [19].

2.1.6 Multi-Step Forecasting

Multi-step forecasting merupakan merujuk pada proses memprediksi nilai di masa depan dari suatu data time series. *Multi-step forecasting* menggunakan data masa lalu sebagai input untuk memprediksi nilai untuk beberapa langkah waktu di masa depan. metode ini memiliki tantangan dimana semakin jauh langkah waktu prediksi yang dilakukan, maka semakin sulit bagi model yang dilatih untuk menghasilkan prediksi yang akurat [20] .

2.2 Framework dan Algoritma yang digunakan

2.2.1 Flask

Flask merupakan *framework open-source* yang tergolong kecil dan *lightweight*, *framework* Flask cukup kecil hingga dapat disebut juga dengan “*micro-framework*”. Flask dirancang sebagai *framework* yang dapat diperluas menggunakan *extension*, sehingga memiliki fleksibilitas yang tinggi [21]. *Framework* flask ditulis dengan menggunakan bahasa pemrograman python [22]. Flask bergantung pada tiga toolkit eksternal, yaitu Werkzeug WSGI toolkit, Jinja *template engine* dan Click CLI *toolkit*.

2.2.2 Bootstrap

Bootstrap merupakan *framework* pengembangan website bersifat *open-source* dan gratis yang menyediakan kumpulan syntax untuk membangun *user interface website*. *Framework* yang digunakan berbentuk *template* desain berbasis HTML dan CSS untuk membuat berbagai komponen dalam *website* seperti *form*, *button*, *NavBar*, serta berbagai komponen *website* lainnya. Bootstrap dapat dengan mudah digunakan karena tidak membutuhkan penulisan kode yang Panjang dan rumit sehingga dapat memudahkan *developer* dalam mengembangkan tampilan *website* dengan cepat [23] .

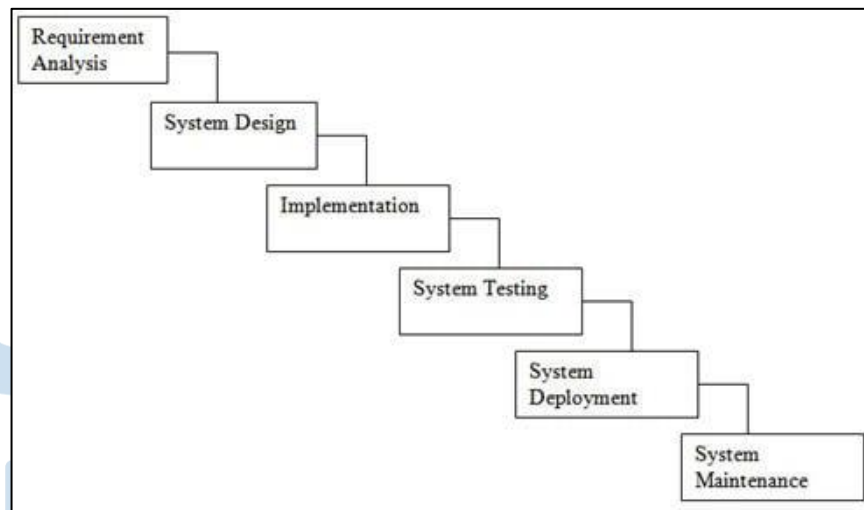
2.2.3 SQLite database

SQLite merupakan database yang dikembangkan oleh D. Richard Hipp, yang, mulai dikembangkan pada tanggal 09 Mei 2000. SQLite dapat mengimplementasikan *self-contained*, *serverless*, *zero-configuration*, *transactional SQL database*. SQLite adalah proyek bersifat *public domain* yang dapat digunakan dalam berbagai kebutuhan secara gratis. Tidak seperti *database SQL* lainnya, SQLite tidak memiliki *separate server process*. SQLite membaca dan menulis data secara langsung ke *file disk*. SQLite merupakan *cross-platform database file format* sehingga *database*

dapat dengan mudah di copy (salin) ke berbagai sistem. SQLite merupakan *library* yang *compact* dengan ukuran yang relatif kecil, yaitu kurang dari 750 KiB (kibibyte) tergantung pada pengaturan optimasi compiler dan target *platform* [24] .

2.2.4 Metode *waterfall*

Model *waterfall* adalah model pengembangan perangkat lunak yang paling umum digunakan. Model pengembangan ini mengikuti urutan linear dari tahap awal pengembangan sistem, yaitu tahap perencanaan, hingga tahap akhir pengembangan sistem, yaitu tahap pemeliharaan. Tahap-tahap selanjutnya tidak dapat dimulai sebelum tahap sebelumnya selesai, dan tidak memungkinkan untuk kembali atau mengulangi tahap sebelumnya. Model SDLC air terjun (*waterfall*) juga dikenal sebagai model sekuensial linier atau alur hidup klasik (*classic life cycle*). Model *waterfall* mengikuti pendekatan alur hidup perangkat lunak secara sekuensial atau terurut, dimulai dari analisis, desain, implementasi, pengujian, dan pemeliharaan (*maintenance*) [25].



Gambar 2.2 Metode *waterfall*

sumber [25]

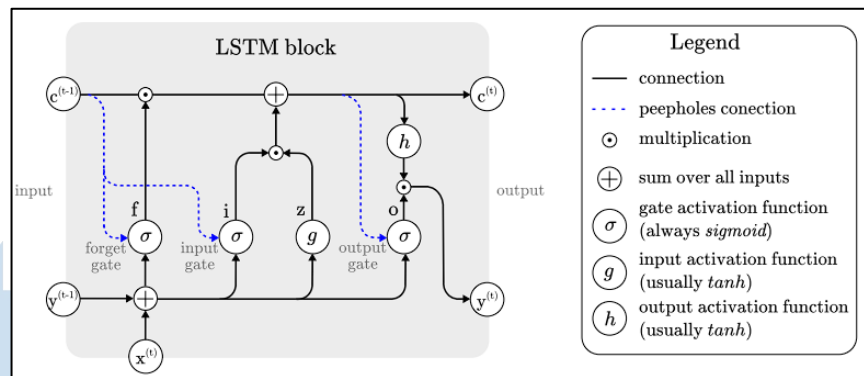
2.2.5 Metode *Prototyping*

Metode *prototyping* merupakan metode pengembangan sistem yang menggunakan prototype untuk mendeskripsikan sistem, sehingga *system owner* atau pengguna aplikasi dapat memiliki gambaran mengenai pengembangan sistem yang sedang dilakukan. Metode *prototyping* merupakan metode pengembangan yang dilakukan secara berulang. Setiap *prototype* akan diperiksa dan dievaluasi oleh pengguna. Permintaan pengguna mengenai perbaikan, perubahan, maupun penambahan yang terkait pada *prototype* saat ini harus dipertimbangkan oleh tim *developer* pada pembentukan *prototype* berikutnya [26].

2.2.6 *Long Short-Term Memory (LSTM)*

Long Short-Term Memory atau LSTM merupakan algoritma dari RNN yang telah dimodifikasi. Algoritma RNN dimodifikasi dengan menambahkan *memory cell* yang memiliki kemampuan untuk menyimpan informasi dalam jangka waktu yang lama [27]. LSTM sangat cocok digunakan untuk *sequence data*. LSTM mampu melakukan prediksi, klasifikasi, dan menghasilkan *sequence data*. *Sequence data* adalah data yang terdiri dari nilai atau kejadian yang terjadi secara berurutan, seperti data historis saham yang termasuk dalam data time series [28].

LSTM terdiri dari sekumpulan *sub-networks* yang terhubung secara berulang, yang dikenal sebagai *memory blocks*. Gagasan dibalik *memory blocks* adalah untuk mempertahankan keadaannya dari waktu ke waktu sembari mengatur aliran informasi informasi melalui *non-linear gating units*. Dapat dilihat gambar 2.3 merupakan arsitektur algoritma LSTM, yang melibatkan *gates*, *Input Signal* $x^{(t)}$ dan *output* $y^{(t)}$, *activation functions*, dan *peephole connections*. *Output* dari blok tersebut akan secara berulang terhubung Kembali ke blok *input* dan ke semua *gates* [28].



Gambar 2.3 Arsitektur LSTM
Sumber : [28]

Blok *Input*, pada tahap ini blok *input* diperbarui dengan menggabungkan saat ini $x^{(t)}$ dengan *Output* dari unit LSTM $y^{(t-1)}$ pada iterasi terakhir. Hal tersebut dapat dilakukan dengan rumus berikut [28]:

$$z^{(t)} = g(W_z x^{(t)} + R_z y^{(t-1)} + b_z)$$

(1)

W_z dan R_z merupakan bobot yang terkait dengan input saat ini $x^{(t)}$ dan *Output* pada iterasi sebelumnya $y^{(t-1)}$, sedangkan b_z merupakan vektor bobot bias. Nilai dari b_z disesuaikan selama proses *training* menggunakan *backpropagation* meminimalkan kesalahan antara *output* prediksi dengan nilai sebenarnya. g merupakan *sigmoid function*.

Dalam LSTM terdapat 3 *gate* yaitu *input gate*, *forget gate*, dan *output gate*. Berikut merupakan penjelasan mengenai tahapan LSTM beserta rumusnya.

Input gate, pada tahap ini *input gate* diperbarui dengan menggabungkan *input* saat ini $x^{(t)}$, *Output* dari unit LSTM $y^{(t-1)}$ dan nilai *cell state* $c^{(t-1)}$ pada iterasi terakhir. Tahap ini :

$$i^{(t)} = \sigma (W_i x^{(t)} + R_i y^{(t-1)} + p_i \odot c^{(t-1)} + b_i) \quad (2)$$

Simbol \odot menunjukkan perkalian dua vektor, W_i , R_i dan p_i adalah bobot yang terkait dengan $x^{(t)}$, $y^{(t-1)}$, dan $c^{(t-1)}$, sementara b_i mewakili vektor bias yang terkait dengan komponen ini.

Pada tahap sebelumnya, LSTM layer menentukan informasi mana yang harus dipertahankan dalam jaringan cell states $c^{(t)}$. Hal tersebut meliputi pemilihan kandidat nilai $z^{(t)}$ yang berpotensi dimasukkan ke cell states, dan activation values $i^{(t)}$ pada input gates.

Forget gate, pada tahap ini unit LSTM menentukan informasi mana yang harus di buang dari cell states $c^{(t-1)}$ sebelumnya. Oleh sebab itu, activation values $f^{(t)}$ dari forget gates pada saat langkah t dihitung berdasarkan input saat ini $x^{(t)}$, outputs $y^{(t-1)}$ dan nilai cell state $c^{(t-1)}$ dari langkah sebelumnya ($t - 1$), dari peephole connections, dan bias b_f dari forget gates. Hal tersebut dapat dilakukan seperti berikut [28]:

$$f^{(t)} = \sigma (W_f x^{(t)} + R_f y^{(t-1)} + p_f \odot c^{(t-1)} + b_f) \quad (3)$$

W_f, R_f , dan p_f merupakan bobot yang berkaitan dengan $x^{(t)}$, $y^{(t-1)}$ dan $c^{(t-1)}$, sedangkan b_f merupakan vektor bobot bias.

Cell, tahap ini menghitung cell value, yang menggabungkan blok input $z^{(t)}$, input gate $i^{(t)}$, dengan cell value sebelumnya. Rumus dari tahap tersebut adalah sebagai berikut :

$$c^{(t)} = z^{(t)} \odot i^{(t)} + c^{(t-1)} \odot f^{(t)} \quad (4)$$

Output gate, tahap ini menghitung output gate yang akan mengontrol bagaimana memory cell dihasilkan sebagai output dari LSTM. Hal tersebut dilakukan dengan menggabungkan input saat

ini $x^{(t)}$, *Output* $y^{(t-1)}$ dan *cell value* $c^{(t-1)}$ pada iterasi sebelumnya. Rumus dari tahap tersebut adalah sebagai berikut :

$$o^{(t)} = \sigma (W_o x^{(t)} + R_o y^{(t-1)} + p_o \odot c^{(t)} + b_o) \quad (5)$$

W_o, R_o dan p_o merupakan bobot yang terkait dengan $x^{(t)}, y^{(t-1)}$ dan $c^{(t-1)}$, sedangkan b_o merupakan vektor bobot bias.

Block output, block output digunakan sebagai input iterasi LSTM berikutnya. Tahap ini menggabungkan *cell value* $c^{(t)}$ saat ini dengan *output gate value* saat ini. Tahap tersebut dapat dirumuskan sebagai berikut :

$$y(t) = g(c^{(t)}) \odot o^{(t)} \quad (6)$$

pada langkah-langkah di atas, σ , g , dan h adalah fungsi aktivasi non-linear. Fungsi sigmoid logistik $\sigma(x) = 1 / 1+e^{-x}$ digunakan sebagai fungsi aktivasi gate, sedangkan fungsi tangen hiperbolik $g(x) = h(x) = \tanh(x)$ sering digunakan sebagai fungsi aktivasi input dan output blok [28].

LSTM dipilih sebagai algoritma untuk memprediksi harga saham karena LSTM memiliki kemampuan untuk mempelajari pola – pola kompleks pada data *time series*. Kemampuan LSTM dalam mempelajari pola – pola kompleks tersebut karena terdapat mekanisme *gating*, sehingga model dapat memilih informasi yang relevan dan mengabaikan informasi yang tidak relevan.

2.2.7 Performance Metrics

Pada persamaan yang akan disajikan, diketahui bahwa O_i adalah nilai observasi atau nilai aktual, S_i adalah nilai prediksi yang dihasilkan, dan n adalah jumlah data point yang tersedia untuk dianalisis

1) MSE

mean squared error (MSE) merupakan selisih antara nilai aktual dan nilai prediksi. Rumus untuk menghitung MSE adalah sebagai berikut [29]:

$$\text{MSE} = \frac{1}{n} \sum_i^n (S_i - O_i)^2 \quad (7)$$

2) RMSE

Root mean squared error (RMSE) adalah akar kuadrat dari rata-rata kuadrat dari semua kesalahan. Penggunaan RMSE sangat umum, dan dianggap sebagai pengukuran kesalahan yang sangat baik digunakan untuk prediksi numerik. Rumus untuk menghitung RMSE adalah sebagai berikut [30]:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_1^n (S_i - O_i)^2} \quad (8)$$

3) MAE

MAE dihitung dengan mengambil rata-rata dari selisih absolut antara nilai prediksi dan nilai aktual. Rumus untuk menghitung MAE adalah sebagai berikut [29]:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |S_i - O_i| \quad (9)$$

4) MAPE

Mean Absolute Percentage Error (MAPE) merupakan metrik evaluasi yang umum digunakan untuk mengukur tingkat kesalahan relatif suatu model dalam memprediksi data, MAPE menampilkan persentase kesalahan dari suatu model, kemudian dikalikan dengan 100. Rumus untuk menghitung MAE adalah sebagai berikut [31]:

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^n \left| \frac{S_i - O_i}{O_i} \right| \times 100\%$$

(10)

Penilaian skala akurasi MAPE dapat dilihat pada tabel 2.1

Tabel 2.1 Penilaian skala akurasi MAPE
Sumber : [32]

MAPE	Forecast Accuracy
< 10%	<i>Highly accurate</i>
11% - 20%	<i>Good forecast</i>
21% - 50%	<i>Reasonable forecast</i>
> 51%	<i>Inaccurate forecast</i>

2.3 Tools yang digunakan

2.3.1 UML






UML atau dapat disebut juga *Unified Modeling Language* merupakan bahasa standar untuk menulis *software blueprints*. UML dapat digunakan untuk memvisualisasi menentukan, membangun, mendokumentasi komponen pada sebuah *software*. UML cocok digunakan untuk memodelkan sistem mulai dari sistem informasi sebuah perusahaan hingga aplikasi berbasis web. UML adalah sebuah bahasa dan merupakan bagian dari *software development method*. UML membantu *developer* dan *user* untuk memahami secara mendalam mengenai sistem yang akan dibuat. Visualisasi UML akan dibuat dalam bentuk diagram, antara lain [33]:

2.3.1.1 Use case diagram

Use case diagram sangat penting untuk memvisualisasikan, menetapkan dan mendokumentasikan *behavior* suatu element. *Use case* diagram bertujuan untuk menjelaskan struktur sistem mengenai hubungan antar elemen – elemen yang terdapat di dalamnya. *Use case diagram* memberikan gambaran visual mengenai bagaimana sistem akan berinteraksi dengan pengguna [33].

Pada *use case* diagram, terdapat aktor yang merupakan entitas yang berinteraksi dengan sistem, seperti pelanggan, pengguna, maupun sistem lainnya. Selain itu terdapat *use case* yang mewakili skenario atau kegiatan tertentu yang dapat digunakan pengguna dengan bantuan sistem [33].

Tabel 2.2 Tabel notasi *Use Case Diagram*
Sumber : [34]

Simbol	Notasi	Keterangan
 Actor	<i>Actor</i>	<i>Actor</i> dapat diartikan sebagai pengguna sistem yang berinteraksi dengan <i>use case</i> .
 Use Case	<i>Use Case</i>	<i>Use case</i> merupakan abstraksi dan interaksi antara sistem dan actor.
	<i>Association</i>	<i>Association</i> menunjukkan hubungan antara <i>actor</i> dan <i>use case</i> .
	<i>Dependency</i>	<i>Dependency</i> mengindikasikan adanya ketergantungan antara satu elemen dengan elemen lainnya.
	<i>Generalization</i>	<i>Generalization</i> menunjukkan adanya hubungan antara <i>child object</i> dan <i>parent object</i>




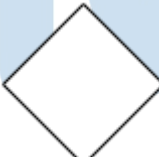
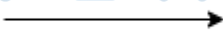
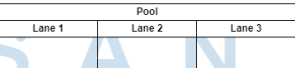
2.3.1.2 *Activity diagram*

Activity diagram berfokus pada kontrol alur dari sebuah aktivitas ke aktivitas lainnya. *Use case diagram* memberikan visualisasi dalam urutan Tindakan atau aktivitas yang dilakukan oleh aktor maupun objek. *Activity*

diagram dapat digunakan dalam berbagai macam proses seperti proses produksi, proses bisnis, dan proses pengembangan *software*. *Activity diagram* berfungsi untuk mengilustrasikan tampilan dinamis dari sebuah system [33].

Dalam *activity diagram*, terdapat beberapa elemen seperti persegi Panjang untuk mewakili aktivitas, panah untuk menghubungkan aktivitas – aktivitas tersebut, symbol horizontal dan vertikal yang digunakan untuk menunjukkan percabangan atau penyatuan alur kerja yang mewakili *fork* dan *join* [33].

Tabel 2.3 Tabel notasi *Activity Diagram*
Sumber : [35]

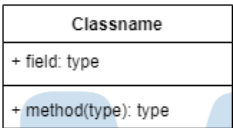


Simbol	Notasi	Keterangan
	<i>Activity</i>	<i>Activity</i> berisi deskripsi tentang interaksi atau kegiatan apa saja yang dilakukan suatu pihak.
	<i>Initial Node</i>	<i>Initial node</i> menunjukkan dimulainya suatu proses atau <i>activity</i> .
	<i>Final Node</i>	<i>Final node</i> menunjukkan berakhirnya suatu proses atau <i>activity</i> .
	<i>Decision</i>	<i>Decision</i> menunjukkan percabangan atau pemilihan kondisi yang mengarah pada jalur yang berbeda
	<i>Line Connector</i>	<i>Line Connector</i> menunjukkan koneksi pada <i>activity diagram</i> .
	<i>Swimlane</i>	<i>Swimlane</i> digunakan untuk mengorganisir dan membagi proses kerja di



Simbol	Notasi	Keterangan
		antara entitas atau aktor yang terlibat dalam sebuah sistem atau proses.

2.3.1.3 Class diagram

Class diagram menunjukkan sekumpulan kelas, *interface*, dan kolaborasi serta hubungannya. *Class* diagram digunakan untuk mengilustrasikan tampilan desain statis dari sebuah sistem. Diagram suatu sistem akan dibentuk berdasarkan kelas – kelas yang terkait dengan sistem tersebut [33].

Tabel 2.4 Tabel notasi *Class Diagram*
Sumber : [35]

Simbol	Notasi	Keterangan
	<i>Class</i>	<i>Class</i> dapat diartikan sebagai komponen utama dalam membangun sistem. Terdapat 3 komponen, yaitu nama <i>class</i> , properti atau atribut yang dimiliki, dan <i>method</i> yang dimiliki.
	<i>Association</i>	<i>Association</i> menunjukkan relasi umum antar kelas. <i>Association</i> biasanya disertai dengan <i>multiplicity</i> .
	<i>Dependency</i>	<i>Dependency</i> menunjukkan relasi antar kelas yang memiliki makna

Simbol	Notasi	Keterangan
		ketergantungan antar kelas.
	<i>Aggregation</i>	<i>Aggregation</i> menunjukkan bahwa sebuah <i>class</i> merupakan bagian dari <i>class</i> lainnya.
	<i>Generalization</i>	<i>Generalization</i> menunjukkan relasi <i>inheritance</i> dari <i>child class</i> ke <i>parent class</i> .

2.3.2 Figma

Figma adalah *software* desain berbasis *cloud* yang. Figma dapat digunakan membuat desain *user interface* maupun produk digital lainnya. Figma populer karena memungkinkan sebuah tim dalam berkolaborasi secara *real-time*. Selain itu Figma mampu digunakan dalam *browser website* tanpa harus diunduh. Figma tersedia dalam versi gratis dan berbayar dengan beberapa fitur yang tidak dapat digunakan pada versi gratis. Secara keseluruhan,, Figma menjadi *tools* yang populer bagi desainer UI/UX [36].

2.3.3 Visual Studio Code

Visual Studio Code merupakan *code editor* gratis dan *open-source* yang dikembangkan oleh Microsoft. Visual studio code memiliki konsep *source code editor* yang sederhana dengan *powerful developer tooling* seperti IntelliSense untuk menyelesaikan kode otomatis dan *debugging*. Visual studio code merupakan *code editor* yang ringan dan *powerfull* serta dapat di pasang di berbagai perangkat seperti windows, macOS, dan Linux. Selain itu, Visual Studio Code dapat digunakan untuk berbagai bahasa pemrograman seperti Python, C#, C++, PHP, Python, dan lain sebagainya [37].

2.3.4 Python

Python merupakan bahasa pemrograman yang sering digunakan dalam aplikasi web, data, pengembangan *software*, dan *Machine Learning* (ML). python digunakan oleh para *programmer* karena mudah dipelajari dan efisien serta dapat dijalankan di berbagai platform [38].

Python merupakan bahasa pemrograman tingkat tinggi yang sedang populer dan dianggap *powerfull*. Python populer karena meningkatnya kebutuhan di beberapa bidang seperti *Face Recognition*, *Artificial Intelligent*, *Face Recognition*, *Machine Learning*, dan bidang lainnya. Python dipercayai di perusahaan – perusahaan besar seperti Facebook, Instagram, Netflix, Google, dan perusahaan – perusahaan lainnya sebagai bagian dari aplikasi mereka. Python merupakan bahasa pemrograman “*Interpreter*”, yang berarti kode akan langsung dijalankan sesuai dengan perintah yang ditulis dalam bahasa pemrograman atau *scripting* tanpa sebelumnya diubah menjadi kode objek seperti *compiler* [38].

Python memiliki banyak modul maupun *library* yang mudah dipahami. Modul-modul tersebut dapat digunakan untuk mendukung kebutuhan di bidang *Data Science*, *Artificial Intelligent*, *Cyber Security*, ekonomi, *statistic*, dan berbagai kebutuhan lainnya. Berikut merupakan modul, *framework*, maupun *library* python yang sedang populer saat ini, antara lain [38]:

- Flask (*Web framework*)
- Django (*web framework*)
- OpenCV Python (digunakan untuk membuat aplikasi *computer vision*)
- Scipy dan Scikit (membuat aplikasi *machine learning* dan *Artificial intelligence*)
- Matplotlib (digunakan untuk membuat grafik keperluan *scientific*)

- Tensorflow (digunakan untuk membuat aplikasi yang ditenagai *deep learning*)

2.3.5 API (yfinance)

Application Programming Interface (API) merupakan *interface* memungkinkan suatu aplikasi dapat terhubung dengan aplikasi lainnya. API dapat dikatakan juga sebagai perantara antar berbagai aplikasi yang berbeda, baik dari *platform* yang sama maupun lintas *platform*. Sebuah *website* dapat memiliki fitur yang lebih lengkap jika menggunakan API [39].

API digunakan untuk menyelesaikan kebutuhan dalam pertukaran informasi dengan penyedia data yang memiliki informasi atau kemampuan untuk menyelesaikan masalah – masalah tertentu. Oleh sebab itu, *developer* tidak perlu menyelesaikan masalah tersebut sendiri, seperti Ketika sebuah aplikasi ingin menampilkan sebuah *map* pada *website*, mereka tidak perlu membuat fitur map dari awal melainkan dapat langsung menggunakan API Google Maps [40]. Pada penelitian ini API *yahoo finance* digunakan untuk mengambil data secara otomatis melalui *library* yfinance. Yahoo finance merupakan situs web yang menyediakan informasi maupun berita mengenai data saham, keuangan, *crypto*, dan *currencies*. Data saham yang terdapat pada *website yahoo finance* dapat diakses dan diunduh dengan gratis [41].

2.3.6 SQLitestudio

SQLitestudio adalah proyek yang dibangun oleh Pawel Salawa dan dimulai pada tahun 2007. Aplikasi tersebut dibangun menggunakan bahasa pemrograman C++. SQLitestudio merupakan aplikasi *desktop open source* yang gratis dan digunakan untuk *browsing* dan menyunting file *database* SQLite. Beberapa fitur yang disediakan SQLiteStudio adalah *advanced SQL code editor*, *encrypted databases*, *custom SQL functions*, *SQL & DDL* [42].

2.3.7 User Acceptance Testing (UAT)

UAT merupakan singkatan dari *User Acceptance Testing* dan sering ditujukan ke pengguna akhir untuk melakukan *software testing* yang dilakukan sebelum sebuah sistem diperkenalkan ke sebuah organisasi. UAT memiliki tujuan utama untuk memastikan sistem baru berjalan sesuai dengan fungsinya dan telah memenuhi persyaratan. UAT diuji oleh pengguna untuk mengetahui apakah sistem tersebut memenuhi kebutuhan pengguna. Tanpa UAT, sebuah sistem yang telah diuji oleh pengembang dan *tester* serta telah lulus seluruh tes, masih dapat gagal jika tidak memenuhi kebutuhan pengguna [43].

2.4 Penelitian Terdahulu

Tabel 2.5 Tabel Penelitian Terdahulu

<i>Authors</i>	<i>Title</i>	<i>Journal</i>	<i>Methods</i>	<i>Result</i>	<i>Suggestion</i>
Gourav Bathla	<i>Stock Price prediction using LSTM and SVR</i> (2020) [10]	<i>2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC)</i>	LSTM, SVR	Pada penelitian ini dibahas mengenai perbandingan antara algoritma LSTM dengan SVR. Data yang digunakan merupakan data saham dari tahun 2015 hingga 2020. Hasil penelitian menunjukkan nilai MAPE pada indeks saham NSE menggunakan LSTM dan SVR sebesar 0.86 dan 1.44, sehingga dapat disimpulkan bahwa LSTM lebih unggul jika dibandingkan dengan SVR.	Pada penelitian selanjutnya, <i>performance metrics</i> yang berbeda seperti RMSE dan MSE akan digunakan untuk mengevaluasi akurasi prediksi.
Sima Siami-Namini, Neda Tavvakol, Akbar Siami Namin	<i>A Comparison of ARIMA and LSTM in Forecasting Time Series</i> (2018) [11]	<i>2018 17th IEEE International Conference on Machine Learning and Applications</i>	ARIMA, LSTM	membahas tentang perbandingan algoritma ARIMA dengan LSTM, penelitian menggunakan data <i>time series</i> bulanan dari Januari 1983 hingga Agustus 2018 yang diambil melalui <i>website yahoo finance</i> . Data bulanan yang digunakan tersebut antara lain Nikkei 225 Index (N225), S&P 500 Commodity price index (GSPC), Hang Seng Index (HIS), Dow Jones	Pada penelitian selanjutnya dapat membandingkan beberapa algoritma <i>machine learning</i> lainnya.

<i>Authors</i>	<i>Title</i>	<i>Journal</i>	<i>Methods</i>	<i>Result</i>	<i>Suggestion</i>
				industrial average index (DJ), dan NASDAQ composite index (IXIC). penelitian tersebut menemukan bahwa akurasi prediksi LSTM mengungguli akurasi prediksi ARIMA dengan nilai rata rata RMSE LSTM sebesar 0.936 dan ARIMA sebesar 5.999.	
Adil MOGHARA ,Mhamed HAMICHE	<i>Stock Market Prediction Using LSTM Recurrent Neural Network</i> (2020) [44]	<i>International Workshop on Statistical Methods and Artificial Intelligence (IWSMAI 2020)</i>	LSTM	pada penelitian ini data saham yang digunakan adalah GOOGL dan NKE yang diambil melalui yahoo finance. Model menggunakan 80% data untuk training dan 20% data untuk testing. prediksi dengan menggunakan 25 epochs menghasilkan Loss 0.0001 pada saham GOOGL dan Loss 0.0016 pada saham NKE. hasil pengukuran menunjukkan bahwa model yang dibuat mampu melacak perubahan harga untuk kedua saham tersebut.	untuk penelitian selanjutnya dapat menggunakan data yang lebih banyak dan lebih baik agar mampu memaksimalkan akurasi prediksi.
Samyak Meshram , Suraj Narsale , Sahil Sangamner , Kiran Kadam , Prof. Vrushali Paithankar	<i>Stock Prediction Webapp Using Python</i> (2022) [12]	<i>International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)</i> Volume 2, Issue 7, May 2022	LSTM, Streamlit, Prototyping	Hasil penelitian ini adalah website untuk memprediksi harga saham menggunakan framework streamlit , algoritma LSTM dan Regression, serta Prototype sebagai metode pengembangan aplikasi. Penelitian ini menyimpulkan bahwa Machine learning dapat digunakan untuk memprediksi pasar saham dengan akurasi dan efisiensi yang bagus.	prediksi pasar saham dapat ditingkatkan lagi dengan menggunakan dataset yang lebih besar. hal tersebut dapat meningkatkan akurasi dari model prediksi yang dibuat.
Kevin Johan, Julio C. Young, Seng Hansun	<i>LSTM-RNN Automotive Stock Price Prediction</i> (2019) [45]	<i>International Journal Of Scientific & Technology Research</i> Volume 8, Issue	LSTM-RNN, ADAM	penelitian ini menghasilkan prediksi menggunakan LSTM-RNN. Penelitian ini menggunakan 5 data saham otomotif yaitu	saran untuk penelitian selanjutnya adalah menggunakan teknik normalisasi Decimal Scaling, menggunakan metode z-score sebagai

<i>Authors</i>	<i>Title</i>	<i>Journal</i>	<i>Methods</i>	<i>Result</i>	<i>Suggestion</i>
		09, September 2019		saham Honda, Mitsubishi, Toyota, Suzuki, dan Daihatsu. Melalui penelitian tersebut didapatkan tingkat akurasi tertinggi dari prediksi masing masing saham adalah 99% untuk saham Honda, 98% untuk saham Mitsubishi, 97% untuk saham Toyota, 99% untuk saham Suzuki, dan 98% untuk saham Daihatsu.	evaluasi, dan melakukan testing tambahan pada parameter ukuran batch dalam proses training.
Seng Hansun, Vincent Charles, Tatiana Gherman	<i>The role of the mass vaccination programme in combating the COVID-19 pandemic: An LSTM-based analysis of COVID-19 confirmed cases</i> (2023) [46]	<i>Heliyon</i> (2023)	LSTM, MAE, MAPE, RMSE	Penelitian ini bertujuan untuk memprediksi jumlah kasus COVID-19 di masa depan untuk sepuluh negara dengan jumlah vaksinasi terbanyak di dunia. Terdapat dua skenario pada penelitian ini, yaitu "All time" dan "Before Vaccination". Skor rata-rata MAPE untuk skenario "All time" dan "Before Vaccination" masing-masing adalah 5,977% dan 10,388%. program vaksinasi massal memiliki efek positif dalam mengurangi dan mengendalikan penyebaran penyakit COVID-19 di negara-negara tersebut, kecuali pada negara India dan Meksiko	Melakukan penelitian yang lebih mendalam untuk mengidentifikasi dan mengatasi hambatan dalam program vaksinasi massal sangat penting. Dengan demikian, sehingga dapat memahami mengapa beberapa negara, seperti India dan Meksiko dalam penelitian ini, tidak menunjukkan efek positif yang sama dari program vaksinasi dan mendapatkan wawasan yang berharga untuk meningkatkan upaya vaksinasi di masa depan. Selain itu, penelitian mendatang dapat mengeksplorasi penggunaan model prediksi yang lebih canggih yang mempertimbangkan faktor tambahan, seperti demografi, pola mobilitas, dan tingkat cakupan vaksinasi. Hal ini akan meningkatkan akurasi prediksi dan memberikan wawasan yang lebih komprehensif bagi para pengambil keputusan.

Dari Tabel 2.5, dapat dilihat terdapat beberapa penelitian yang telah dilakukan. Penelitian pertama dengan judul "Stock Price prediction using

LSTM and SVR” [10] dan penelitian kedua dengan judul “*A Comparison of ARIMA and LSTM in Forecasting Time Series*” [11] memiliki kesimpulan bahwa algoritma memiliki akurasi yang lebih baik jika dibandingkan dengan algoritma SVR dan LSTM, kesimpulan tersebut menjadi landasan dalam pemilihan algoritma LSTM untuk memprediksi harga saham. Selain itu, terdapat beberapa kesamaan antara topik – topik tersebut dengan penelitian ini. Penelitian ketiga dengan judul “*Stock Market Prediction Using LSTM Recurrent Neural Network*” memiliki kesamaan algoritma dalam melakukan prediksi saham sehingga dapat dijadikan referensi [44]. Pada penelitian keempat dengan judul “*Stock Prediction Webapp Using Python*” memiliki kesamaan algoritma yang menggunakan LSTM dan metode prototyping, sehingga memiliki kemiripan dengan penelitian ini. oleh sebab itu penelitian tersebut dapat dijadikan acuan dalam melakukan pengembangan website [12]. Penelitian kelima dengan judul “*LSTM-RNN Automotive Stock Price Prediction*” yang memiliki kesamaan algoritma dalam memprediksi yaitu algoritma LSTM, maka penelitian tersebut dapat dijadikan panduan dalam penelitian ini [45]. Pada penelitian ini akan dikembangkan website untuk memprediksi harga saham dengan inovasi data yang akan diperbarui setiap minggunya dengan mengambil data menggunakan library yfinance.

