DOKUMEN SPESIFIKASI PROYEK *BIG DATA*

Implementasi Arsitektur Big Data untuk Analisis Perdagangan Saham BEI 2020–2024

Kelompok 14

Kiwit Novitasari	121450126
Rayan Koemi Karuby	122450038
Safitri	122450071
Amalia Melani Putri	122450122

PROGRAM STUDI SAINS DATA

FAKULTAS SAINS

INSTITUT TEKNOLOGI SUMATERA

Mei 2025

Daftar Isi

1. Pendahuluan

Transformasi digital dalam sektor keuangan telah mendorong kebutuhan akan sistem yang mampu mengelola dan menganalisis data dalam skala besar dan kompleks. Bursa Efek Indonesia (BEI), sebagai penyelenggara perdagangan di pasar modal Indonesia, menyediakan berbagai solusi produk Data Pasar yang dikembangkan untuk memberikan informasi kepada publik agar dapat membuat keputusan yang tepat [1]

Data historis perdagangan saham mencakup berbagai atribut penting seperti harga pembukaan dan penutupan, volume transaksi, serta aktivitas investor lokal dan asing. Kompleksitas dan besarnya volume data ini menjadikan pendekatan konvensional tidak lagi memadai untuk keperluan analisis mendalam, deteksi pola, maupun visualisasi data secara real-time. Oleh karena itu, dibutuhkan implementasi teknologi Big Data yang mampu menangani karakteristik data volume, velocity, dan variety secara efisien [2].

Teknologi Big Data menyediakan solusi untuk menyimpan, mengolah, dan menganalisis data dalam volume besar, kecepatan tinggi, dan beragam format [3]. Dalam konteks ini, implementasi arsitektur big data menjadi sangat penting untuk memungkinkan integrasi data historis pasar saham ke dalam sistem yang dapat menangani kebutuhan komputasi paralel, penyimpanan terdistribusi, serta akses data yang efisien. Arsitektur big data dirancang untuk memfasilitasi penyimpanan terdistribusi, pemrosesan paralel, dan akses data yang cepat melalui penggunaan tools seperti Hadoop Distributed File System (HDFS), Apache Hive, Apache Spark, serta sistem manajemen metadata dan pipeline integrasi data. Implementasi arsitektur ini memungkinkan proses ingest, transformasi, dan analisis data historis secara terstruktur dan scalable.

Studi yang dilakukan oleh Fang dan Zhang [4], menunjukkan bahwa pemanfaatan big data dalam analisis keuangan dapat meningkatkan akurasi prediksi hingga 30% dibandingkan dengan metode tradisional. Mereka menemukan bahwa integrasi data struktural dan tidak terstruktur dari berbagai sumber seperti media sosial, sensor IoT, dan transaksi online memungkinkan pemahaman yang lebih komprehensif tentang tren pasar dan perilaku konsumen. Hal ini menegaskan pentingnya penerapan big data dalam sektor keuangan untuk meningkatkan kualitas analisis dan pengambilan keputusan.

Dengan mengacu pada dataset saham Indonesia tahun 2020–2024 yang mencakup 938 perusahaan terdaftar, proyek ini menjadi studi implementatif untuk membangun sistem pemrosesan data besar yang sesuai dengan kebutuhan analitik pasar keuangan modern. Pemanfaatan teknologi big data dalam konteks ini tidak hanya meningkatkan efisiensi pemrosesan data, tetapi juga membuka peluang untuk eksplorasi informasi yang lebih dalam dari data yang sebelumnya sulit dikelola.

1.1 Tujuan Dokumen

- Membangun arsitektur Big Data berbasis Hadoop yang dapat menyimpan dan memproses data historis secara terdistribusi dengan komponen HDFS, Hive, YARN, dan Spark.
- Mengimplementasikan pipeline pengolahan data dari proses ekstraksi data, transformasi, hingga penyimpanan menggunakan tools seperti Hadoop, Spark, dan Hive untuk kebutuhan analisis skala besar.
- Melakukan analisis historis data berbasis query dan visualisasi dengan Apache Superset

1.2 Lingkup Sistem

Sistem pengolahan data ini dibangun dengan arsitektur tiga lapisan: Bronze, Silver, dan Gold. Lapisan Bronze menyimpan data mentah dari sumber saham ke dalam HDFS sebagai data lake tanpa transformasi. Di lapisan Silver, data dibersihkan dan diproses menggunakan Spark dan Hive, lalu disimpan dalam format terstruktur di Hive. Lapisan Gold menyajikan data siap analisis melalui

koneksi Hive-SQL dan divisualisasikan menggunakan Apache Superset. Pendekatan ini memastikan alur data yang efisien dari akuisisi hingga visualisasi.

2. Deskripsi Umum

2.1 Perspektif Sistem

Sistem dimulai dari tahap *Data Source & Ingestion*, di mana data saham dikumpulkan dan dimasukkan ke dalam penyimpanan awal menggunakan HDFS sebagai *data lake*. Selanjutnya, pada *Serving Layer*, data mentah tersebut diproses secara batch menggunakan Apache Spark yang dijalankan oleh YARN sebagai mesin pemroses. Hasil pemrosesan disimpan dalam Hive sebagai data terstruktur yang siap untuk dianalisis. Pada tahap *Query Service*, pengguna dapat mengakses dan menjalankan query terhadap data di Hive melalui Spark-Hive Thrift Server dengan antarmuka SQL. Terakhir, data yang telah dianalisis divisualisasikan menggunakan Apache Superset untuk mendukung pemahaman dan pengambilan keputusan. Perspektif ini menggambarkan sistem yang terintegrasi, mulai dari ingesting data mentah hingga penyajian informasi visual secara efektif.

2.2 Fungsi Sistem Utama

Berikut fungsi sistem utama dalam bentuk poin berdasarkan lapisan Bronze, Silver, dan Gold:

- **Bronze Layer (Data Mentah)**
    - Menyimpan data saham mentah secara langsung ke dalam HDFS sebagai data lake.
    - Menyediakan sumber data lengkap tanpa proses transformasi untuk kebutuhan referensi dan pemulihan.
- **Silver Layer (Data Terproses)**
    - Melakukan pembersihan dan transformasi data menggunakan Apache Spark dan Hive.
    - Menghasilkan data terstruktur yang siap untuk dianalisis dan diproses lebih lanjut.
- **Gold Layer (Data Siap Analisis dan Visualisasi)**
    - Menyajikan data hasil agregasi dan ringkasan melalui query SQL di Hive.
    - Memfasilitasi visualisasi data menggunakan Apache Superset untuk mendukung pengambilan keputusan.

2.3 Karakteristik Pengguna

Karakteristik pengguna sistem ini mencakup beberapa kelompok utama. Pertama, analis data yang memiliki pemahaman tentang data saham dan kemampuan menjalankan query SQL untuk menggali wawasan dari data terstruktur. Mereka menggunakan hasil pemrosesan data untuk melakukan analisis tren dan pola pasar. Kedua, manajer atau pengambil keputusan yang membutuhkan visualisasi data yang mudah dipahami melalui dashboard Apache Superset agar dapat memantau kinerja saham dan membuat keputusan strategis berdasarkan analisis tersebut. Ketiga, data engineer yang bertanggung jawab mengelola pipeline data, memastikan proses pengambilan, transformasi dengan Spark, serta pemeliharaan Hive dan integrasi dengan sistem visualisasi berjalan lancar. Terakhir, pengguna teknis

lainnya yang memiliki kemampuan menggunakan antarmuka SQL melalui Spark-Hive Thrift Server untuk mengakses dan mengeksplorasi data secara mendalam.

3. Spesifikasi Proyek

3.1 Ringkasan

Sistem pengolahan data saham ini dirancang dengan arsitektur tiga lapisan, Bronze, Silver, dan Gold untuk mengelola data secara efisien dari tahap pengumpulan hingga visualisasi. Data mentah disimpan di lapisan Bronze menggunakan HDFS sebagai data lake, kemudian diproses dan dibersihkan di lapisan Silver dengan Spark dan Hive agar menjadi data terstruktur. Pada lapisan Gold, data siap dianalisis disajikan melalui Hive-SQL dan divisualisasikan menggunakan Apache Superset. Sistem ini memanfaatkan YARN sebagai mesin pemroses dan Spark-Hive Thrift Server untuk query, sehingga mendukung integrasi dan otomasi pipeline data secara batch. Dengan pendekatan ini, sistem mampu menyediakan informasi yang akurat dan mudah diakses untuk mendukung pengambilan keputusan berbasis data saham.

3.2 Metode Proyek

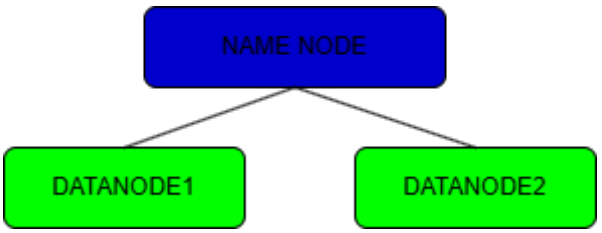
Metodologi pengembangan proyek ini mengacu pada pendekatan Lambda Architecture, yaitu sebuah pendekatan yang dirancang untuk menangani volume data besar dengan efisien melalui pemisahan proses pemrosesan batch, penyimpanan data terstruktur, dan akses cepat terhadap data untuk eksplorasi time-series. pendekatan metodologi dilakukan secara bertahap dan sistematis yang berfokus pada tiga arsitektur utama: pembangunan arsitektur Big Data, perancangan pipeline pengolahan data, serta analisis dan visualisasi data saham. Langkah awal dalam implementasi metodologi ini adalah pembangunan infrastruktur Big Data melalui penyusunan klaster Hadoop berskala kecil dengan konfigurasi terdistribusi. Klaster ini kemudian di-*deploy* menggunakan kontainerisasi untuk mempermudah eksekusi, replikasi lingkungan, dan pengelolaan dependensi perangkat lunak secara lokal. Pendekatan ini juga mendukung skalabilitas dan fleksibilitas dalam pengembangan sistem data skala besar. Beberapa komponen utama dari ekosistem Hadoop yang digunakan dalam klaster ini meliputi:

- HDFS (Hadoop Distributed File System) sebagai sistem penyimpanan terdistribusi untuk menyimpan data dalam bentuk blok yang tersebar di berbagai node.
- YARN (Yet Another Resource Negotiator) sebagai *resource manager* dan penjadwal pekerjaan (job scheduler), yang memungkinkan pemrosesan paralel dan efisien antar node.
- Hive, sebagai *Data Warehouse* yang mendukung query berbasis SQL (HiveQL) untuk memudahkan eksplorasi dan agregasi data oleh pengguna yang terbiasa dengan sintaks SQL.

Seluruh konfigurasi dilakukan secara terpusat, mencakup pengaturan core-site, hdfs-site, yarn-site, dan hive-site, sehingga klaster dapat bekerja secara sinkron dan optimal. Arsitektur sistem divisualisasikan dalam bentuk diagram yang menggambarkan hubungan antar node serta peran masing-masing komponen dalam sistem. Secara struktur, klaster Hadoop terdiri dari:

- Satu NameNode yang berfungsi sebagai pengelola metadata dan direktori sistem file.
- Dua DataNode yang digunakan untuk menyimpan blok data secara terdistribusi.

Masing-masing node dijalankan dalam sehingga komunikasi antar kontainer tetap stabil dan menyerupai sistem terdistribusi sesu kontainer berbeda dan dihubungkan melalui jaringan virtual yang dibuat menggunakan Docker Network,ngguhnya. Konfigurasi ini tidak hanya mempermudah pengujian dan pengembangan, tetapi juga merepresentasikan sistem Big Data modern yang fleksibel dan modular.



Gambar 1. Topologi Cluster Hadoop

3.3 Studi Kasus

Studi kasus ini berfokus pada implementasi arsitektur Big Data untuk menganalisis data historis dan perilaku perdagangan saham di Bursa Efek Indonesia (BEI) selama periode 2020–2024. Dengan menggunakan dataset "IDX Stock Summary 2020–2024" yang berisi data transaksi harian lebih dari 900 perusahaan, sistem ini dirancang untuk mengatasi tantangan pengelolaan data yang besar, kompleks, dan beragam. Arsitektur yang dibangun memanfaatkan ekosistem Hadoop, termasuk HDFS sebagai penyimpanan terdistribusi, YARN untuk manajemen sumber daya, Apache Spark untuk pemrosesan batch, serta Hive sebagai gudang data terstruktur. Pipeline data ini mengotomasi proses ingest, transformasi, penyimpanan, hingga analisis menggunakan query SQL dan visualisasi dengan Apache Superset. Studi ini menunjukkan bagaimana penerapan teknologi Big Data dapat meningkatkan efisiensi pengolahan data saham dan mendukung analisis mendalam untuk pengambilan keputusan di pasar modal

4. Metode Umum

Metode yang digunakan dalam proyek ini mengikuti pendekatan sistematis berbasis arsitektur Big Data dengan pipeline yang terstruktur mulai dari pengumpulan data, pemrosesan, hingga analisis dan visualisasi. Proses dilakukan secara batch menggunakan Hadoop dan Spark, serta penyimpanan terdistribusi menggunakan HDFS dan Hive. Metode ini dirancang untuk memastikan data yang besar dan beragam dapat diproses secara efisien dan scalable, dengan mempertimbangkan kebutuhan akses cepat dan analisis mendalam.

4.1 Analisis Kebutuhan

Tujuan:

1. Memahami masalah utama, yaitu kurangnya sistem yang mampu mengolah dan menganalisis data historis perdagangan saham Indonesia dalam skala besar secara efisien dan terstruktur.

2. Menentukan stakeholder, termasuk Bursa Efek Indonesia, analis keuangan, pengembang sistem, dan pengguna akhir yang membutuhkan laporan dan visualisasi data.

3. Mendefinisikan output utama berupa laporan analisis tren harga saham, volume perdagangan, serta visualisasi interaktif untuk mendukung pengambilan keputusan pasar modal.

4. Mengidentifikasi data yang dibutuhkan, seperti kode saham, harga pembukaan dan penutupan, volume transaksi, tanggal, dan sektor industri.

Tabel Kebutuhan Fungsional

No	Kebutuhan	Prioritas
1	Mengumpulkan data historis perdagangan saham dari sumber terpercaya (misal dataset IDX 2020–2024)	Tinggi
2	Menyimpan data mentah dalam sistem file terdistribusi (HDFS)	Tinggi
3	Membersihkan dan memvalidasi data (format tanggal, nilai null)	Tinggi
4	Mengubah format data menjadi efisien (Parquet/ORC) untuk komputasi lanjutan	Sedang

5	Mengelompokkan data berdasarkan kode saham dan periode waktu	Tinggi
6	Menghitung total volume dan tren harga saham per kode saham dan periode	Tinggi
7	Menyediakan query analitik melalui Hive untuk pengguna akhir	Tinggi
8	Menyediakan visualisasi sederhana (grafik tren harga dan volume per bulan)	Sedang
9	Menyusun pipeline batch processing berkala (mingguan/bulanan)	Sedang

Tabel Kebutuhan Non Fungsional

No.	Kebutuhan	Prioritas
1	Sistem dapat diskalakan horizontal jika volume data meningkat (Skalabilitas)	Tinggi
2	Sistem dirancang dengan ketersediaan tinggi dan minimal downtime	Tinggi
3	Akses data dibatasi berdasarkan peran pengguna (security dan kontrol akses).	Sedang
4	Pipeline data batch harus bisa recovery saat gagal atau crash (Reliabilitas).	Tinggi
6	Format data seperti Parquet/ORC digunakan untuk kompresi dan efisiensi (Efisiensi Penyimpanan)	Sedang

4.2 Perancangan Sistem: Arsitektur dan Desain (*System Design*)

Arsitektur Data

Tabel Arsitektur Medallion – Batch Processing dengan Data Lake

Lapisan	Deskripsi	Komponen Utama (Tools)	Format Data	Tujuan
Bronze	Menyimpan data mentah tanpa transformasi dari sumber asli	- HDFS	CSV / JSON / Raw	Arsip permanen data mentah dari sumber eksternal tanpa perubahan.
Silver	Menyimpan data yang telah dibersihkan dan ditransformasikan: validasi, parsing tanggal, penghapusan duplikat, dll.	- Apache Spark (ETL) - Apache Hive (DDL atau Data Definition Language) - Hadoop HDFS	Parquet	Data terstruktur siap dianalisis. Terintegrasi, bersih, dan efisien dalam penyimpanan.
Gold	Menyimpan agregasi dan data siap analitik, misalnya total volume perdagangan dan harga rata-rata saham per sektor/periode.	- Apache Hive (Analytic Queries) - Apache Spark (Aggregation) - Apache Superset	Parquet / ORC	Untuk kebutuhan laporan, query cepat, dan visualisasi ke end-user.

Desain Infrastruktur

Arsitektur Cluster Lokal (Hadoop Cluster)

Komponen	Jumlah Node	Peran dan Deskripsi
Hadoop Namenode	1	Master node untuk HDFS, mengelola metadata file dan koordinasi penyimpanan.
Hadoop Datanode	2	Menyimpan data dalam blok, diduplikasi untuk toleransi kesalahan dan ketersediaan data.
ResourceManager	1	Komponen YARN yang menjadwalkan job Spark/Hive.

NodeManager	2	Menjalankan dan mengelola eksekusi task pada masing-masing worker node.
Apache HiveServer2	1	Layanan query engine untuk Hive SQL.
Apache Spark Master	1	Koordinator eksekusi job Spark batch, mengatur distribusi tugas ke worker nodes.
Apache Spark Worker	2	Mengeksekusi job Spark (ETL, transformasi, agregasi).
Apache Hive Metastore	1	Metadata store untuk skema dan tabel Hive, memastikan konsistensi data di seluruh cluster
Ambari Server	1	Monitoring dan manajemen layanan Hadoop ecosystem.
Superset	1	Platform visualisasi data untuk menampilkan dashboard dan laporan dari data gold layer.

Tabel Daftar Teknologi Apache Projects yang digunakan

No	Teknologi	Kategori	Fungsi Utama
1	Hadoop HDFS	Storage	Menyimpan data mentah (bronze), data terproses (silver), dan data agregasi (gold) secara terdistribusi.
2	Hadoop YARN	Resource Management	Mengelola sumber daya cluster (CPU dan RAM) serta menjadwalkan eksekusi job Spark secara efisien.
3	Apache Hive	Query Engine / Metadata	Menyediakan interface SQL-like (HiveQL) dan manajemen metadata untuk kebutuhan analitik
4	Apache Spark	ETL / Analytics Engine	Menjalankan proses pembersihan data, transformasi, dan agregasi dalam batch processing
7	Apache Superset	BI / Visualisasi	Menyajikan visualisasi data hasil agregasi dari layer gold untuk mendukung pengambilan keputusan bisnis.

Orkestrasi Alur Kegiatan Sistem

Urutan	**Aktivitas**	**Layer**	**Tools**
1	Mengambil data dari sumber FTP/API kemudian upload ke HDFS	Bronze	Bash, curl, HDFS CLI
2	Menyimpan data mentah tanpa modifikasi	Bronze	HDFS
3	Melakukan validasi schema dan pembersihan data	Silver	Apache Spark, Hive Metastore
4	Mengkonversi format data menjadi Parquet dan menyimpan di layer refined	Silver	Apache Spark + HDFS
5	Melakukan agregasi dan perhitungan metrik ekspor berdasarkan kategori dan waktu	Gold	Spark SQL / Hive
6	Menyimpan hasil akhir agregasi dalam bentuk tabel analitik	Gold	HDFS, Apache Hive
7	Melakukan visualisasi dan eksplorasi data	Gold	Apache Superset, Jupyter Notebook

4.3 Implementasi (Implementation)

Lingkungan implementasi mencerminkan arsitektur batch processing berbasis data lake yang terdiri dari tiga layer (Bronze, Silver, Gold), dengan orkestrasi sederhana untuk pemrosesan berkala dan otomatisasi pipeline.

Lingkungan Implementasi

Komponen	Spesifikasi
OS	Ubuntu Server 22.04 (via Docker VM) – OS komputer: Windows 11
Cluster	Hadoop pseudo-distributed cluster dengan 2 node (1 master, 1 worker)
Orkestrasi	Shell Script + Crontab
Core Tools	Hadoop HDFS, Apache Spark, Apache Hive, Superset
Penyimpanan	HDFS sebagai distributed file system untuk semua layer data (bronze-gold)
Format Data	CSV (bronze), Parquet/ORC (silver & gold)

Instalasi dan Setup Cluster

a. Persiapan Cluster Lokal

1. Install Docker Desktop dan aktifkan WSL2 backend.
2. Jalankan container Hadoop cluster menggunakan docker-compose.
3. Konfigurasi layanan utama yang dijalankan:
    - 1 NameNode
    - 2 DataNode
    - 1 ResourceManager
    - 2 NodeManager
    - 1 Spark Master
    - 1 Spark Worker
    - 1 Hive Metastore + HiveServer2
    - 1 Superset

b. Struktur Docker-Compose

```
version: "3.7"
services:
  namenode:
    image: bde2020/hadoop-namenode:2.0.0-hadoop3.2.1-java8
    container_name: namenode
    environment:
      - CLUSTER_NAME=bigdata-cluster
    ports:
```

```
    - "9870:9870"
  volumes:
    - namenode_data:/hadoop/dfs/name

 datanode1:
  image: bde2020/hadoop-datanode:2.0.0-hadoop3.2.1-java8
  container_name: datanode1
  volumes:
    - datanode1_data:/hadoop/dfs/data

 spark-master:
  image: bitnami/spark:latest
  container_name: spark-master
  ports:
    - "8080:8080"

 spark-worker:
  image: bitnami/spark:latest
  container_name: spark-worker

 hive-metastore:
  image: bde2020/hive:2.3.2-postgresql-metastore
  container_name: hive-metastore

 superset:
  image: apache/superset
  container_name: superset
  ports:
    - "8088:8088"

 ambari:
  image: sequenceiq/ambari:2.7.4
  container_name: ambari
  ports:
    - "8089:8080"

volumes:
  namenode_data:
  datanode1_data:
```

Struktur Folder HDFS

Layer	Path HDFS	Format
Bronze	/data/bronze/	CSV, JSON
Silver	/data/silver/	Parquet
Gold	/data/gold/	ORC
Temp (untuk temporary)	/data/tmp/	-

Implementasi Pipeline Batch

1. **Bronze Layer (Ingestion)**

Dilakukan secara otomatis menggunakan cron dan bash script. Contoh:

```
curl -o saham.csv https://data.example/api/idx_stock_summary
hdfs dfs -put -f saham.csv /data/bronze/saham_$(date +%F).csv
```

2. **Silver Layer (Cleansing & Normalization)**

Pembersihan dan normalisasi data menggunakan Apache Spark:

```
from pyspark.sql import SparkSession

# Inisialisasi SparkSession
spark = SparkSession.builder.appName("CleanIDX").getOrCreate()

# Membaca data dari Bronze Layer (CSV saham)
df = spark.read.csv("/data/bronze/saham_*.csv", header=True)

# Pembersihan data: hapus duplikat dan isi nilai null dengan "N/A"
df_clean = df.dropDuplicates().na.fill("N/A")

# Simpan data bersih ke Silver Layer dalam format Parquet
df_clean.write.parquet("/data/silver/idx_clean.parquet", mode="overwrite")
```

3. **Gold Layer (Aggregation & Analytics)**

Agregasi nilai ekspor per negara dan bulan:

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import sum

# Inisialisasi SparkSession
spark = SparkSession.builder.appName("AggregateIDX").getOrCreate()

# Membaca data bersih dari Silver Layer
df = spark.read.parquet("/data/silver/idx_clean.parquet")

# Agregasi nilai transaksi saham berdasarkan kode saham dan bulan
agg = df.groupBy("Stock Code", "Last Trading
Date").agg(sum("Previous").alias("total_previous_close"))

# Simpan hasil agregasi ke Gold Layer
agg.write.parquet("/data/gold/idx_summary.parquet", mode="overwrite")
```

Implementasi Query dan Visualisasi

Hive Table DDL

```
CREATE EXTERNAL TABLE idx_summary (
  stock_code STRING,
  last_trading_date STRING,
  total_previous_close DOUBLE
)
STORED AS PARQUET
LOCATION '/data/gold/idx_summary.parquet';
```

Dashboard

Gunakan Apache Superset atau Jupyter + PyHive untuk eksplorasi data gold layer.

Monitoring dan Logging

Komponen	Fungsi
Spark UI	http://localhost:8080 untuk melihat job dan eksekusi Spark
Log Files	Tersimpan otomatis di /logs/ dalam container (dapat dikustomisasi)

Output Implementasi

Output	Lokasi
Data mentah	/data/bronze/
Data bersih (normalized)	/data/silver/
Ringkasan nilai	/data/gold/
Tabel Hive	Hive Metastore
Visualisasi ekspor	Superset / Jupyter Dashboard

4.4 Penerapan (*Deployment*)

Strategi Deployment

Tahapan	Deskripsi
1. Setup Environment	Instalasi Docker Desktop + Ubuntu VM
2. Build & Configure	Konfigurasi layanan HDFS, Spark, Hive, Superset.
3. Data Ingestion	Otomatisasi pengambilan data via curl dan cron ke Bronze layer
4. Spark Transformation	Pembersihan dan agregasi data dengan Apache Spark batch
5. Hive Integration	Buat tabel Hive dari hasil Spark Gold Layer
6. Visualisasi	Deploy dashboard di Apache Superset / Jupyter

Teknologi Deployment

Tools	Peran
Bash + Crontab	Menjadwalkan ingestion
Spark Submit	Menjalankan job ETL
Hive Metastore (MySQL/PostgreSQL)	Menyimpan metadata table Gold layer
Apache Superset	Menyediakan tampilan visualisas

Struktur File di Server

```
//opt/bigdata/
├── docker-compose.yml
├── spark_jobs/
```

```
│   ├── clean_saham.py        # Membersihkan data saham mentah (drop duplicates, handle null)
│   └── agg_saham.py          # Mengagregasi total harga penutupan sebelumnya per saham/tanggal
├── ingestion/
│   └── fetch_saham.sh        # Mengunduh data IDX dari sumber dan upload ke HDFS bronze
├── hive/
│   └── idx_summary.ddl       # Hive DDL untuk tabel agregasi saham (gold layer)
├── data/
│   ├── bronze/               # Data mentah hasil unduhan (CSV)
│   ├── silver/               # Data bersih hasil cleansing Spark (Parquet)
│   ├── gold/                 # Data agregasi hasil Spark (Parquet)
│   └── tmp/                  # Temp folder jika diperlukan proses antara
└── logs/
```

4.5 Pengujian (*Testing*)

Pengujian sistem dilakukan untuk memastikan bahwa seluruh proses dari ingest data, transformasi, hingga visualisasi berjalan tanpa kesalahan dan sesuai ekspektasi. Pendekatan pengujian dilakukan dalam beberapa level: unit, integrasi, kualitas data, performa, dan end-to-end.

Jenis Pengujian

Tipe Uji	Tujuan Uji
Uji Unit (Unit Test)	Memastikan bahwa setiap komponen seperti script ingestion atau Spark ETL dapat dijalankan dan menghasilkan output sesuai harapan
Uji Integrasi	Menilai integrasi antara sistem HDFS, Spark, Hive, dan dashboard visualisasi
Uji Kualitas Data	Mengecek konsistensi, completeness, dan keunikan data hasil transformasi
Uji Performa	Mengukur efisiensi waktu eksekusi dari job ingestion dan Spark job
Uji Ujung ke Ujung	Simulasi aliran penuh pipeline mulai dari pengambilan data hingga visualisasi

Kasus Uji (Test Cases)

No	Kasus Uji	Deskripsi	Status
1	Pengambilan data berhasil	File hasil unduhan tersedia di direktori /data/bronze/	✅
2	Transformasi pembersihan data sukses	Dataset Silver tidak memiliki nilai kosong/duplikat	✅
3	Agregasi Gold Layer selesai	File ringkasan agregat tersimpan di /data/gold/	✅
4	Query Hive berhasil	SELECT terhadap tabel memberikan hasil valid	✅
5	Visualisasi muncul di dashboard	Grafik berdasarkan tampil di Superset	✅

Tools Testing

Tools	Fungsi
Jupyter Notebook	Mengeksekusi Spark script secara manual untuk melihat hasil transformasi
Shell Script	Menjalankan ingestion secara terjadwal dan otomatis via crontab
Hive CLI	Melakukan query langsung ke Hive untuk visualisasi data
Superset	Melihat output akhir dan visualisasi
Log File (.log)	Audit log saat ingestion dan job Spark dijalankan

4.6 Analitik (*Analytics*)

Tahapan analitik dilakukan setelah data perdagangan saham dari Bursa Efek Indonesia (BEI) disimpan pada Gold Layer. Proses ini bertujuan untuk menggali wawasan strategis dan membangun model prediktif menggunakan Apache Spark MLlib. Eksperimen awal dilakukan melalui Jupyter Notebook, sementara hasil visualisasi dan evaluasi akhir dipublikasikan menggunakan Apache Superset. Metadata dan hasil olahan tersimpan dalam Hive Metastore.

Tujuan Analitik ML

Analitik difokuskan untuk menjawab dua pertanyaan utama berbasis data historis saham:

1. **Prediksi harga saham** atau indikator lain untuk periode mendatang.

Data Input untuk MLlib

1. Sumber: Tabel **Gold Layer** di Hive (hasil agregasi dan pembersihan).
2. Format: Parquet/ORC.
3. Kolom:
    - Stock Code
    - Ticker
    - Company Name
    - Previous
    - Open Price
    - First Trade
    - High
    - Low
    - Last Trading Date
    - DateTime

Metodologi Analitik (Pilih 1)

No	Analisis	Algoritma Spark MLlib	Tujuan
1	Prediksi Harga Penutupan	Linear Regression	Estimasi harga penutupan saham di hari berikutnya

Tahapan Analisis

1. **Load Data**

Baca data dari Gold Layer (Hive atau Parquet):

```
df = spark.read.parquet("/data/gold/idx_summary.parquet"
```

2. **Preprocessing**:
    - ✔ Normalisasi fitur numerik seperti Open Price, High, Low, Previous
    - ✔ Encoding fitur kategorikal: Stock Code, Company Name
    - ✔ Handling Missing: .na.fill() atau .dropna()
3. **Splitting Data**:

Pisahkan data untuk pelatihan dan pengujian:

```
train_df, test_df = df.randomSplit([0.8, 0.2], seed=42)
```

4. **Modeling**:

Contoh regresi prediksi harga:

```
from pyspark.ml.regression import RandomForestRegressor
model = RandomForestRegressor(featuresCol="features", labelCol="Previous")
pipeline = Pipeline(stages=[...])
trained_model = pipeline.fit(train_df)
```

5. **Evaluasi**:
   - ✔ Metrik: RMSE, MAE (regresi)
6. **Menyimpan Model**:

```
trained_model.save("/models/stock_predict_model")
```

7. **Inference & Hasil:**

Prediksi harga saham hari berikutnya:

```
predicted = trained_model.transform(test_df)
```

Output disimpan:

- Sebagai file di /data/gold/insight/
- Atau tabel Hive: prediksi_saham

4.7 Dataset

Dataset yang dimanfaatkan dalam tugas ini berjudul "IDX Stock Summary 2020 – 2024", yang diperoleh dari platform bersama yaitu Kaggle. Dataset ini menyajikan data historis mengenai aktivitas perdagangan saham di Bursa Efek Indonesia (BEI) selama kurun waktu 2020 – 2024. Dengan cakupan lebih dari 900 saham dari berbagai sektor industri, dataset ini dijadikan objek studi dalam analisis berskala besar tugas ini, khususnya di bidang ekonomi makro Indonesia.

Struktur data tersaji dalam bentuk tabel, di mana setiap baris menggambarkan catatan transaksi harian dari masing-masing saham. Kolom penting dalam dataset ini antara lain:

Kolom	Tipe Data	Deskripsi
stock_Code	String (varchar)	Kode saham unik tiap perusahaan
ticker	String (varchar)	Simbol saham, biasanya sama dengan stock code.
company_mame	String (text)	Nama lengkap perusahaan.
remarks	String (text)	Catatan tambahan; bisa berupa kode atau keterangan lain.
previous	Float	Harga penutupan sebelumnya, berupa nilai desimal.
open_price	Float	Harga pembukaan.

first_trade	Float	Harga transaksi pertama di hari itu.
high	Float	Harga tertinggi pada hari tersebut.
low	Float	Harga terendah pada hari tersebut.
last_tranding_ date	Date	Tanggal terakhir saham diperdagangkan.
date_time	Date/Tim estamp	Rentang waktu data

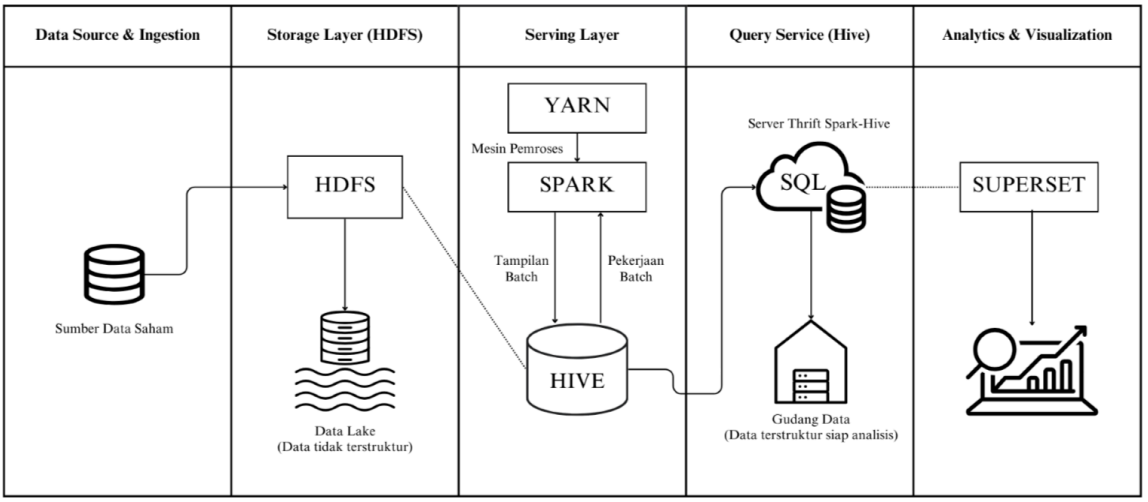
Deskripsi Data

Format Penyimpanan

Berikut format data setelah di ingest ke Hadoop setiap lapisannya

1. **Bronze Layer**: (Raw) CSV di HDFS /bronze/ekspor/2023-01.csv
2. **Silver Layer**: (Cleaned) Parquet /silver/ekspor/2023.parquet
3. **Gold Layer**: (Aggregated) ORC/Parquet dengan hasil agregasi per komoditas & negara

4.8 Aliran Data (*Pipeline*)



Gambar 2. Pipeline Big Data

Pipeline Big Data ini dirancang untuk memproses data saham historis secara batch menggunakan ekosistem Hadoop. Proses dimulai dari pengambilan data saham yang dimuat ke dalam HDFS sebagai data lake untuk menyimpan data tidak terstruktur. Apache YARN mengelola sumber daya pemrosesan, sementara Apache Spark menjalankan batch job terhadap data, dan hasilnya disimpan ke dalam Hive sebagai serving layer.

Data dalam Hive kemudian diakses melalui Spark-Hive Thrift Server menggunakan SQL, membentuk data warehouse yang siap dianalisis. Terakhir, Apache Superset digunakan untuk visualisasi, memungkinkan penyajian data dalam bentuk grafik dan dashboard interaktif.

5. Lampiran

5.1 Repositori Portofolio

https://github.com/sains-data/Analisis-Bursa-Saham-Indonesia

5.2 Lampiran Code

Silahkan isi file-file code yang digunakan di proyek.