NOVEMBER 2024

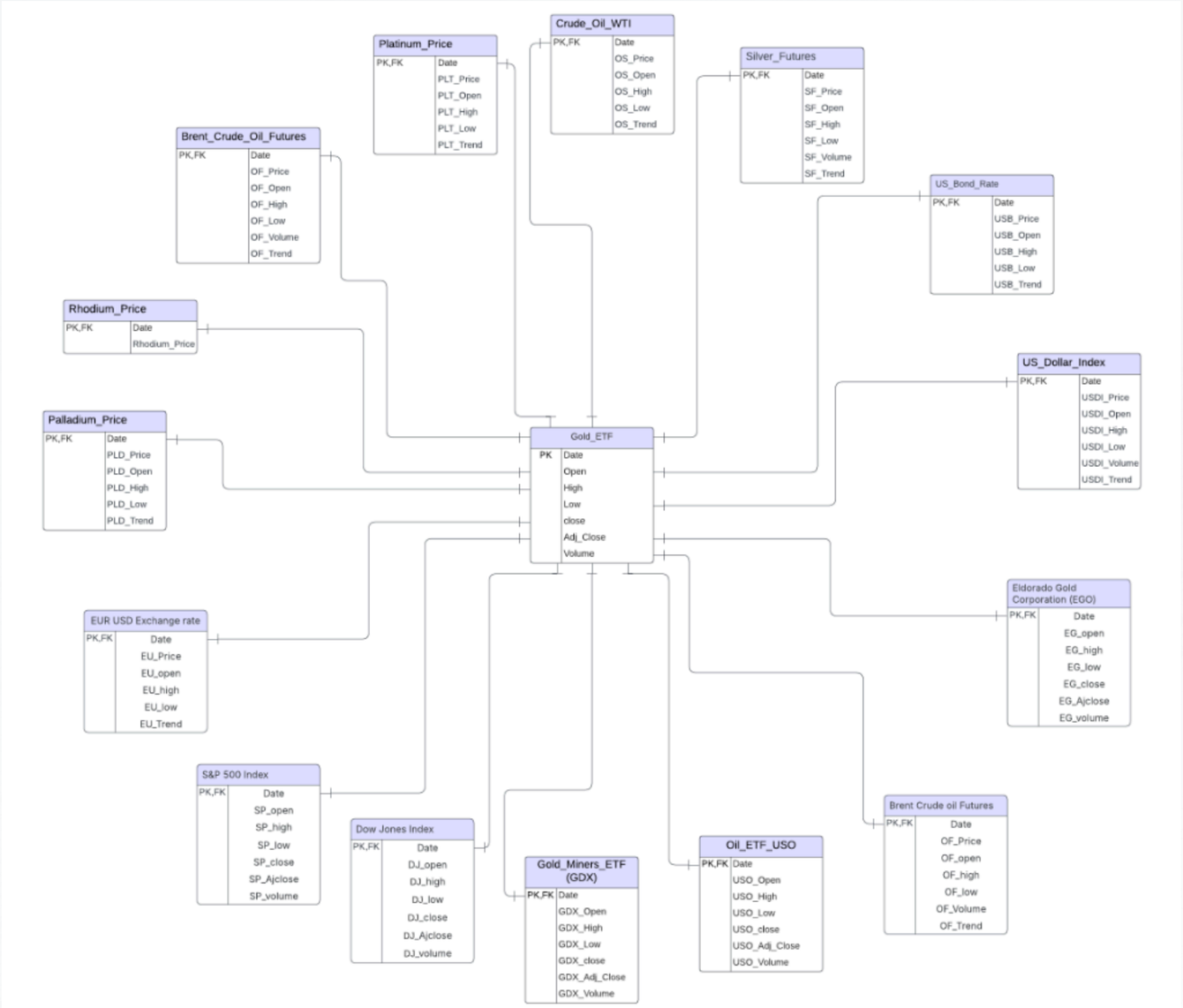
FINANCIAL DATA ANALYSIS - GROUP 8

LAB 3

Data Reliability Framework

1. **Data Governance**
   - Ensure data accuracy, ownership, and compliance with quality standards and regulations.
2. **Collection & Validation**
   - Use reliable sources, consistent formats, and validate with interpolation and outlier detection.
3. **Security & Access**
   - Protect data with role-based access, encryption, and regular security audits.
4. **Versioning & Backup**
   - Implement version control and follow backup protocols (3-2-1 rule) with regular recovery tests.
5. **Real-Time Updates & Documentation**
   - Enable continuous data updates, with clear documentation and metadata tracking.

Tables, Attributes and ER Diagram



LAB 4

DATA PIPELINE DESIGN PATTERN

To predict gold prices while incorporating world events, the **Lambda Architecture** is ideal.

It combines **batch processing for historical data** analysis with **real-time streaming for immediate response** to world events with the speed layer, enabling accurate, timely predictions.

Advantages of Lambda Architecture for Predicting Gold Prices:

- **Comprehensive Analysis**: Combines batch and real-time processing for insights from historical trends and quick reactions to new events.
- **Flexibility & Scalability**: It accommodates large datasets and high-velocity datastreams, ensuring efficient data handling at scale.
- **Accuracy & Completeness**: The dual system provides the ability to refine the predictions over time. Batch processing improves accuracy due to large data volumes, while real-time enhances timliness and relevance.

DATA COLLECTION

SOURCE 1: KAGGLE & GFG

Data Issues with Kaggle Dataset:

- **Inaccurate Prices**: Gold prices showed as 100-200 USD vs. verified 2600 USD.
- **Unrealistic Ratios**: Silver prices listed higher than gold, against market norms.
- **Missing Data**: Many gaps and zeroes for commodities like Rhodium.
- **Outdated Range**: Data only covers 2011-2019.
- **Limited Flexibility**: No real-time updates or customization options for dynamic analysis.

SOURCE 2: Web Scraping Yahoo Finance (yfinance api)

Web scraping ensures you collect up-to-date, accurate data directly from reliable sources (like Yahoo Finance).

The data was collected from multiple markets to analyze the relationships between various economic factors and gold prices. The sources include stock indices, commodity futures, exchange rates, and bond rates

DATA PRE-PROCSESSING

01. Handling Missing Values

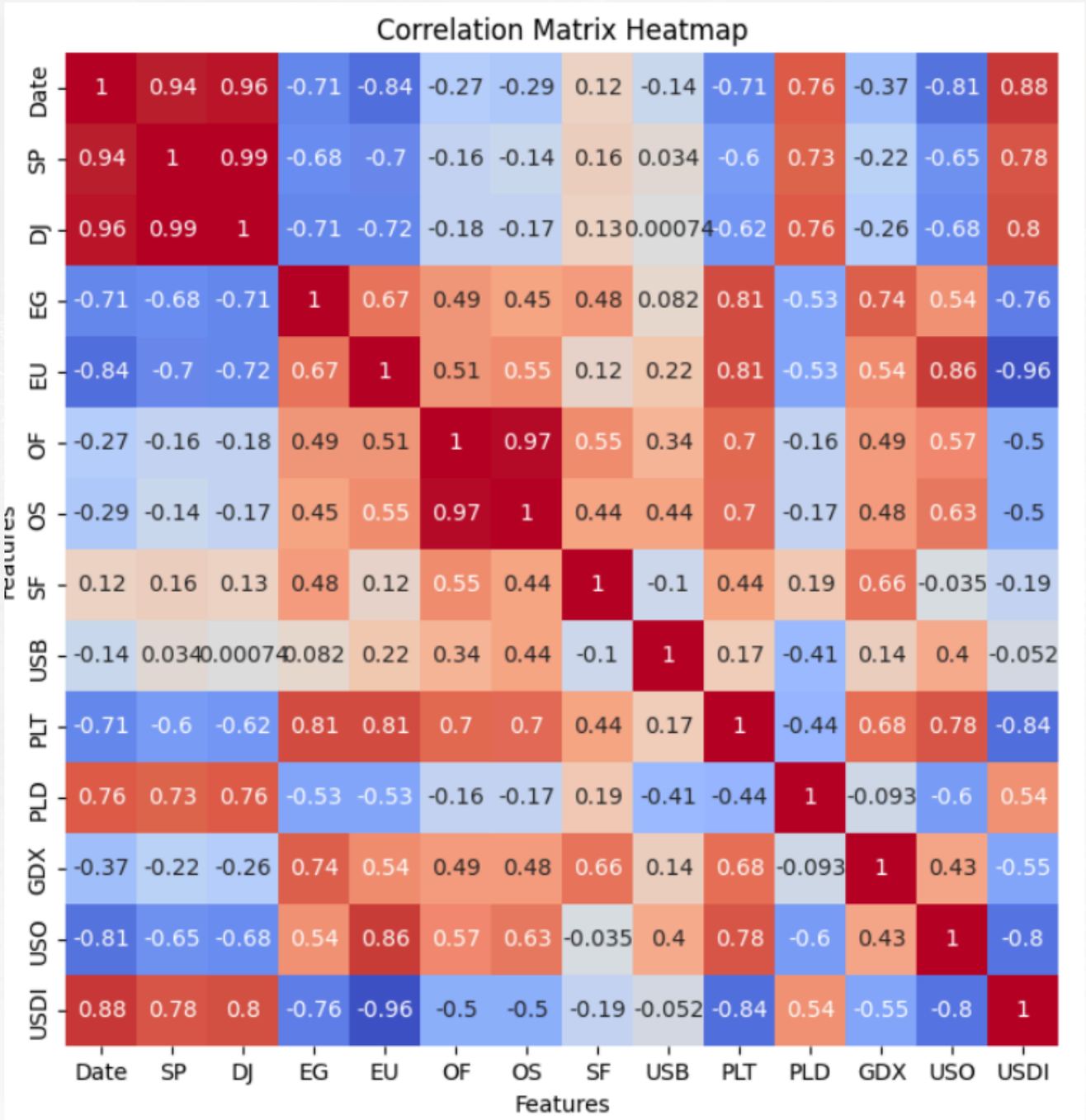
Checked for missing or null values using isna().

Any missing values are then addressed using the forward and backward fill method (ffill, bfill), where NaN values are replaced with the last valid data point

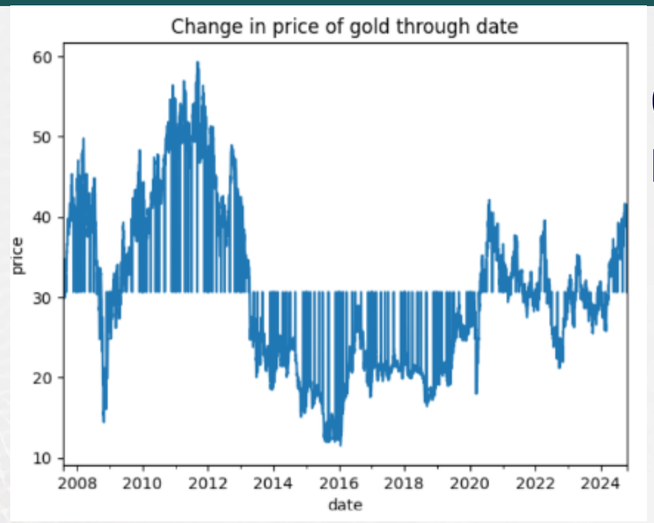
02. Correlation Analysis

A correlation matrix is calculated to understand the relationships between different features in the dataset.

This is visualized using a heatmap, allowing us to identify strong positive or negative correlations between variables, which is essential for feature selection and analysis.



DATA WRANGLING

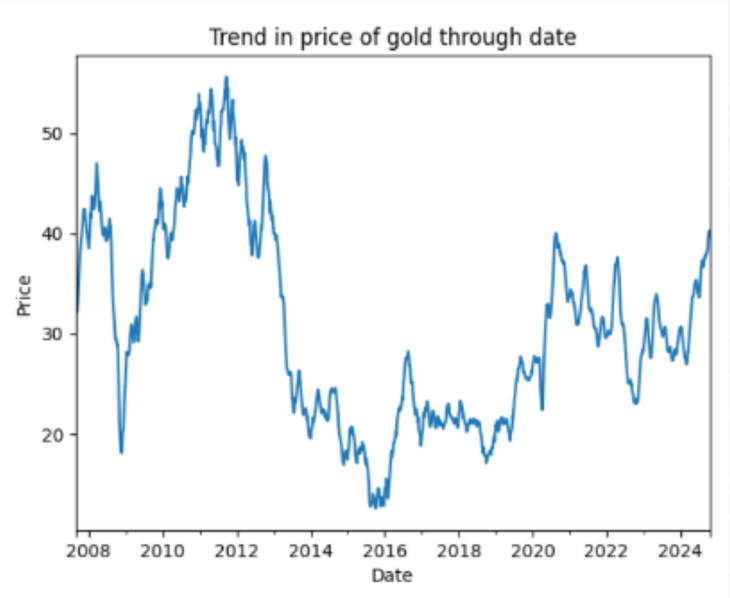


01. Change in Gold Price over time

A line plot was generated to visualize the change in the price of gold (GDX)
over time, helping to observe trends and fluctuations across the dataset.

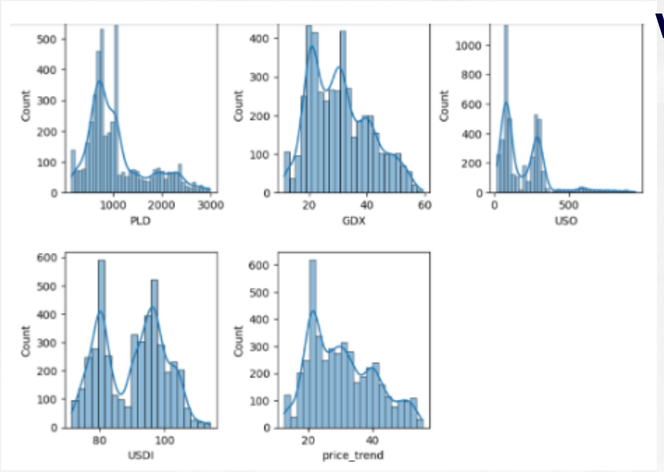
02. Overall price trend of Gold over time.

A rolling mean with a window size of 20 was applied to smooth out fluctuations in gold prices, and a line plot was generated to visualize the overall price trend over time



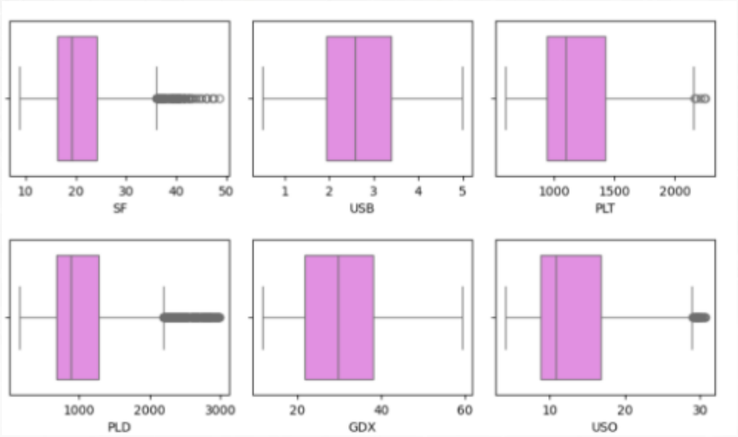
03. Visualise data distribution over various features

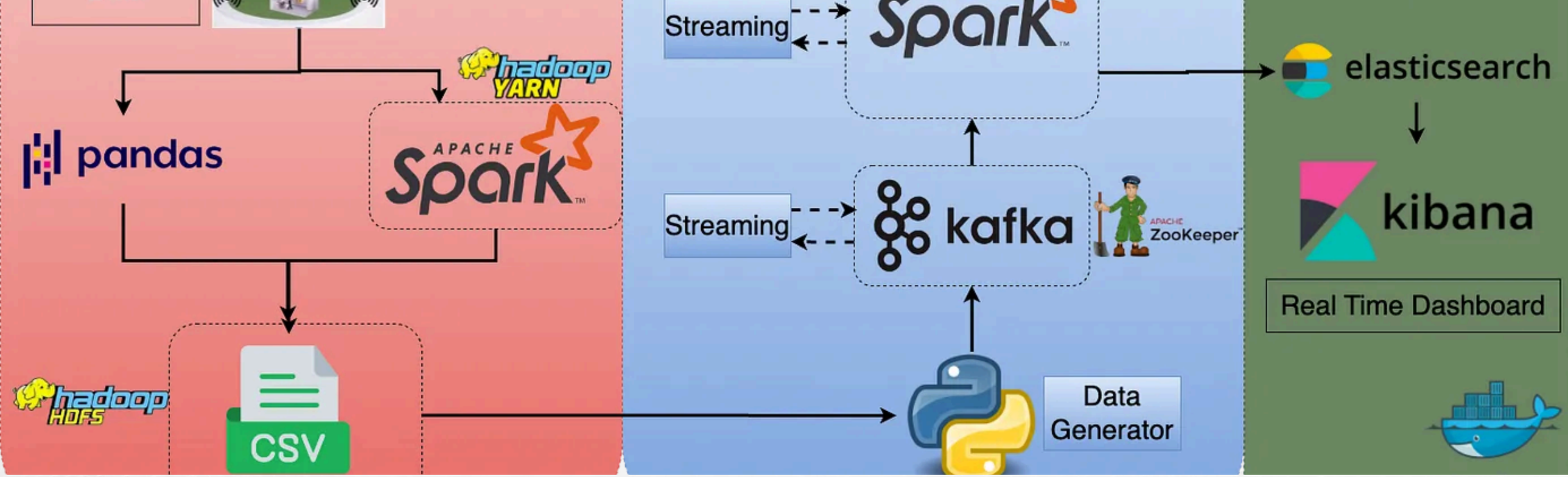
Histograms with KDE were plotted for each numeric column to visualize the distribution of the data across various features.



04. Outlier Detection

Box plots were then generated to visualize the distribution and detect potential outliers across the remaining numeric features.





BATCH LAYER

Ingests and processes historical data using Pandas; stores it in HDFS.

SPEED LAYER

We simulate real-time streaming data by sending records to a Kafka topic.
Spark Structured Streaming is employed to consume data from the Kafka topic in real-time.
The streaming data is then processed and written to Elasticsearch for indexing.
The processed data can be visualized using Kibana, which connects to Elasticsearch to display real-time dashboards.

SERVICE LAYER

It provides tools for querying, visualizing, and interacting with the processed data.
Kibana is used for visualization

DOCKER CONTAINERS

Kafka: Real-time data ingestion, Zookeeper:maintains configuration information, Spark: Batch and stream data processing, Elasticsearch: Search and analytics engine, Kibana: Data visualization interface.

We created a Docker Compose YAML file to specify all the services, networks, and volumes required for the application in a single file, simplifying the process of starting and stopping multiple containers with a single command

INITIAL SETUP USING PYSPARK

For the batch layer:

we took the pre processsed data and converted it into spark dataframe that has to be written on Elasticsearch.

For the Speed layer:

we have created a kafka topic for the real time data ingestion, where financial data (gold price) is taken at a fixed interval.

then a spark dataframe is made of those updated values.

VIEWS AND MATERIALISED VIEWS

Views in financial data management simplify querying by standardizing data across multiple tickers, ensuring users access only relevant columns without knowing each table's structure. They enable seamless aggregations for metrics, like moving averages, and create logical separation, allowing table changes without affecting end-user access to data.

1. **Daily Average Price View**: This view could show the average price per asset per day by calculating (Open + Close) / 2.
2. **Monthly Summary View**: A view that calculates monthly aggregates like average closing price, highest and lowest values, and total volume.
3. **Volatility View**: Financial analysts often assess volatility, which can be calculated by measuring the daily percentage change. A view can compute this directly, saving users from writing complex formulas

Materialized views in financial data management boost performance by pre-computing metrics, reducing redundant calculations, and allowing incremental updates to stay current. They also aid in data archiving and versioning, supporting efficient historical analysis without full recalculations

The following materialized views have been created:
1. **Yearly Summary Materialized View**: Create a materialized view for yearly summaries that aggregates data by year (e.g., average open, close, high, and low values).
2. **Moving Averages Materialized View**: For financial analysis, moving averages (direction of trend) are commonly used. A materialized view can store the 7-day and 30-day moving averages of the closing prices for each ticker.
3. **Volume Trend Materialized View**: This view stores cumulative or trend-based volume metrics, providing fast access to historical volume trends.

Since the data refreshes daily, a job or trigger is scheduled to refresh the materialized view at regular intervals.