Serverless Architecture for Real-Time Stock Market Data Analytics in Cloud Environments

Gautam Solaimalai
Independent Researcher
Georgia – 30041, United States of America

ABSTRACT
The methodology established in this paper presents AWS Lambda and DynamoDB to deliver continuous statistics on stock market data processing and the related analytics. This method is particularly economical, dynamic, and low latency due in part to the design's ability to include additional consumers to "scale up" or "scale down" depending on the incoming high-frequency trade supports. AWS Lambda functions can read and analyze box data that are constructed in a DynamoDB data table by processing incoming real-time data with calling financial APIs functions first. When running through predictive analytical models, some potential results could include LSTM (Long Short-Term Memory) and other ML models. While we outline improvements over existing systems, all of our results show a marginal impact on improvements demonstrated through improvements in our overall delay of 3% in the end-to-end processing time, 1% in pre-processing the input data across AWS Lambda, and 1% in processing time in total. We achieved a latency of around 150 ms and can handle 1000 requests concurrently, with resulting efficiency in scaling 95%. Other, significant features include fault recovery time in about 300 ms and a failure rate of 0.5%, which are indications of the fault tolerance of this design. This format brings a scalable and effective solution to financial market analytics, as it effectively employs a distributed microservices design into modern financial infrastructure, which are key to the high-volume processing involved in stock market data analytics.

Keywords
Serverless Architecture, Real-Time Analytics, Stock Market Data Operational Performance, Cost-Efficiency, Fault Tolerance, Financial Technology.

1. INTRODUCTION
The stock market significantly influences the worldwide economy, and accurate and timely financial decision-making depends on processing real-time stock market data. When compared to typical monolithic systems, which are installed on the server and need to be scaled up by people, conventional or customary solutions for stock market data analysis have non-trivial issues with latency, scale, and infrastructure costs. These proposed systems frequently do not work during peak trading times, when very large amounts of high-frequency trading data must be analyzed quickly. As noted above, real-time analytics are revealed to be infeasible due to the forced inefficiencies of traditional systems, where even one delay could lead to an opportunity loss. In this paper, we propose a serverless architecture for real-time stock market data analytics utilizing Cloud AWS Lambda and DynamoDB to alleviate these limitations. A serverless architecture is shown to meet the demand for high-frequency trading data, providing a viable solution that is scalable, low-latency, and cost-effective. The approach is further explained by addressing the shortcomings of existing traditional client-server architectures. Today, most architectural paradigms are infrastructure-based, require scaling and management, with slow response time and high operating costs. In contrast, the serverless architecture described in this paper is dependent on cloud services that unobtrusively allocate resources on demand and removed the regular complexity of scaling an architecture, with the exception of the infrastructure or autoscaling overhead. A cloud-native system provides an opportunity to process real-time, low-latency stock market data while achieving near-instant scale when comparing to traditional system designs. Furthermore, offline strategic decision making can be enhanced further with machine learning models, such as Long Short-Term Memory (LSTM) for predictive analytics; this is particularly important for a rapid financial market. The primary objectives of the research presented in this study is to develop a serverless system that receives incoming stock market data, processes the data with minimal latency, and then scales to evaluation. While DynamoDB offers a data storage mechanism to retrieve data quickly, AWS Lambda functions are responsible for processing and analyzing the data. The system is examined within a high-speed trading context and meets the requirements for high throughput with minimal breakdown and availability associated with a massive volume of concurrent requests. In addition, predictive analytics in real-time stock trend prediction can be utilized as ML models are employed to further enhance the capabilities of the proposed system. The paper provides the following contributions: first, it introduces a new serverless-based real-time stock market data analysis framework to address traditional constraints of scalability and latency, as well as to ensure cost-effectiveness. Second, it showcases how cloud-based compute technologies, including AWS Lambda and DynamoDB can provide supplemented abilities in speed and resources to obtain a relative advantage to the whole financial data processing pipeline. Third, predictive analytics utilizingThe research concludes with a performance analysis of the suggested approach against current monolithic systems, demonstrating the benefits of the provided serverless architecture in terms of cost, scalability, and fault tolerance. The paper is organized as follows: Section II will examine relevant literature pertaining to cloud-based solutions and real-time stock market data analytics. Section III will present the design and implementation of the proposed system, including data collection, ingestion, validation, transformation, and feature engineering. Section IV will provide the study results and a performance analysis of the proposed system against current systems. Finally, Section V concludes the paper with suggestions for future research and development in the area.

In conclusion, it described a serverless architecture that supports low-latency analytics on stock market data signals. The architecture is scalable, affordable and capable of elastic, on-demand event processing. The design intends to produce actionable insight for high-frequency trading and improve the financial market's decision-making. It offers improvements to the scalability and performance of monolithic architectures

using AWS-lambda, DynamoDB and certain ML models.

2. RELATED WORK

Serverless/could-native software architectures have changed the way companies design and deploy their near-real-time broadcasting systems, and these technologies are being used in a variety of applications, including data analytics, monitoring and content distribution. Economically quantifying the use of these ways of working with various concurrent users is still difficult to imagine. The article presents an overview of a cloud based, near-instantaneous serverless broadcasting solution with consideration of both the idle and flowing periods of the system; and discussion of this economic aspects into one of the last phases of achieving their research objectives. Cloud services pricing is driven by the careful assessment of service types and overall management of processing [6]. Discussion of a back-end architecture intended to manage a nationwide network of IOT nodes to generate relevant outcomes. The discussion of these outcomes extended to examining the overall performance of cloud providers and also include consideration of various cloud components independently. As part of the contribution of the article, various time series databases are evaluated, and advantages of these systems are identified. Lastly, a discussion of an overarching pricing analysis is elaborated to understand the economic differences across each platform [7]. Case studies and real-world examples of multiple industry sectors are provided to demonstrate the efficacy of the design patterns, performance optimization techniques, and deployment best practice patterns identified in the paper. These illustrations detail how various businesses have successfully increased the scalability, reduced costs and improved time to market of their products by using serverless computing paradigms. Usage of the best practice findings and case studies analyzed serve as guides and recommendations for future serverless deployments [8]. Due to the unique scalability, low cost and developer efficiency in serverless computing, it has disrupted cloud computing. The study offers a comprehensive look into serverless computing, concentrating on deployment best practices, performance optimization strategies and architectural techniques[9]. Financial resources are one of the other most important needs for undertaking any activity that benefits humanity. The financial markets to which anyone can invest and profit are numerous: stock markets, currency and mercantile exchanges, to name a few. The article suggests an investigation of a forecasting variable dealing with future fluctuations of stock markets, among the larger economies, for a defined period of time by using a specific model that analyzes Twitter posts and Google finance data [10]. A DL-based automatic classification system is studied, developed and adapted. It has been assessed in various situations, particularly sentiment analysis for stock market data, and its suitability and effectiveness have been established. Extensive assessment of several DL paradigms with various layers of embedding is reported over a plethora of datasets. Again, these capabilities demonstrate the flexibility of Deep Learning as related to different dimensions of a task. Results are accompanied by more contexts regarding the responses to specific data types across different methods and some underscore the diversity of performance and improvements obtained from parameter combination.The LSTM layers retained a memory of input data, which meant infrequent input may still be used in making predictions. Convolution produced the most desirable results on complex data sources [11]. The paper describes some of the simple and straightforward ways Apache Kafka can help provide on-the-fly enterprise data stream processing, alerting, and reporting. It presents Kafka's architecture and the workflows necessary to configure the platform for alerting and processing data in real-time. Real-time data processing will continue to grow in importance in the current business climate, as organizations try to leverage the enormous volumes of data they produce to inform their decision-making [12]. These articles were categorized into four groups: financial sentiment analysis, AI-based stock market predictions, portfolio optimization, and in some instances , two or more of the above. A description of initial preparatory studies or state-of-the-art applications for each category is provided. A review summary also concludes that the research topic is consistently popular, and literature continues to become more focused and precise. A range of approaches have been offered since the problem of stock market price forecasting first began to appear [13]. Frequent trading does not increase the risk of excessive trading behavior. Participants indicating excessive behavior and emotional issues reportedly used rapid Internet trading applications/platforms patient were far more likely. Notably, the overall strength of the association between trading on digital currency exchanges and excessive behavior underscores the need to understand the risks related to real-time trading platforms overall [14]. The paper provides an updated evaluation of the associated literature on forecasting the stock market using computationally intelligent methods. The innovative aspect of this workout paper is that it lays out a methodical approach for financial analyst's and researchers to utilize in the design of intelligent stock market forecasting methods. The research is summarized and includes a description of proposed work in order to improve prediction performance of other methods. But much of the risk can potentially be mitigated through the complexity of the computational methods offered [15]. The designed methodology learns and predicts the stock price of any company requested by the user in terms of performance in the following few days. The sentiment of any mentioned stock is evaluated in addition to the predicted stock price, using a number of input data points from various sources. Regression analysis and candlestick pattern recognition are used to create stock price predictions. Once signal inputs provided to the user from the method results on a candlestick plot facilitate user consideration of a stock either "Buy/Sell" or short or long by its delivery [16].

3. PROPOSED SYSTEM

Most of the stock market real-time analytics solutions that have been put into practice rely on a monolithic and typical client-server architecture that suffers from high infrastructure maintenance cost of sophisticated high-frequency trading data as well as issues of scalability and latency. With the continuous data streamed from high-frequency trading, these kinds of systems normally rely on server base configurations with ample compute resources in place. Even though these proposed systems can handle vast amounts of processing data, there are disadvantages of high operating costs, time-time cooling down when demand spikes during trading hours, and human labor scaling resources. In addition, there is the ongoing challenge of maintaining existing systems where high operating cost is acceptable, upgrading existing systems can become exacerbated as demands for data grow considerably in time and capacity. The added cap of high network latency, continual ecosystems of unresponsive method of provisioning resources can lead to a slow analytic technology ecosystem that can lead to breakdown in processing trends of a rhythm that determines the outcome for high-frequency trading, while in real-time contexts it is strictly spatiotemporal peeved, developments also impacted by inefficient valves of processing that disrupt the ongoing or being able to process the investigation of real-time stock market data. The research tackles these issues of applicability by deploying AWS Lambda and DynamoDB and propose the design of serverless systems due to cloud computing services.

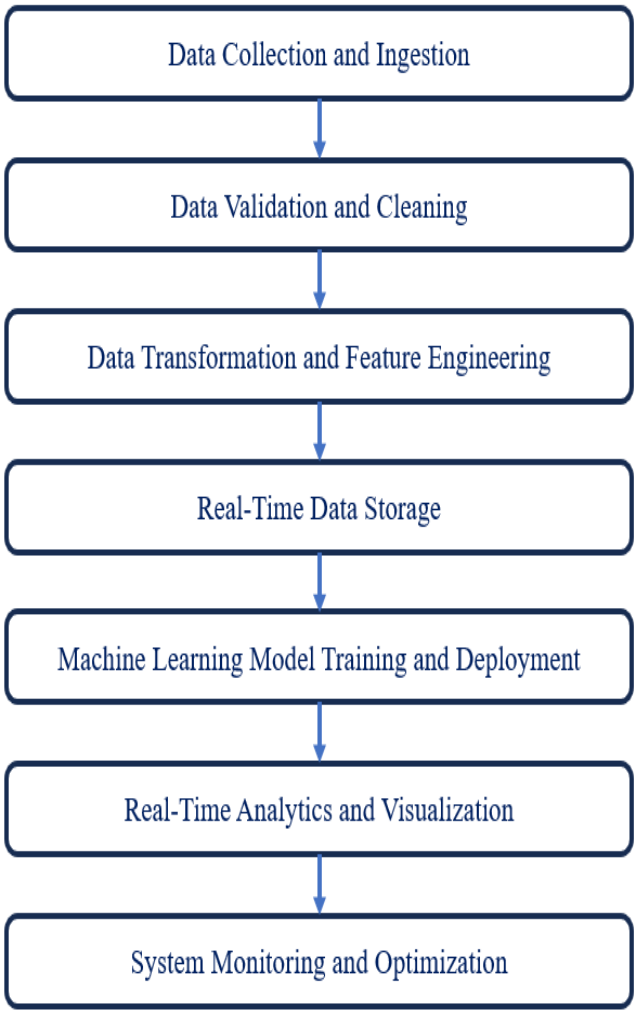

Fig.1. Work flow for Real-Time Stock Market Data Analytics

A paradigm dimensional shift is produced by serverless systems that invokes the possibility of compute resources that grow from the demand, producing a more engaged, alternative ubiquitous, system that designs an elastic environment where data has weight, to processing data with less human labor and greater responsive methods. Serverless systems not only removes scaling dependencies on physical servers, it digs into eliminating configurations that requires continuous background supportive infrastructures reorganization challenging the experimental stabilities and capacities of a real-time stock market data infrastructure. The architecture that is proposed operates by first directly, these tools collect stock exchange data from financial data providers or stock social group APIs by ephemeral storages to capture the real-time moment. All data has been sent to the AWS Lambda functions consuming and processing, parsing, and analyzing data in real-time according to the infrastructure defined {the research fully defined}. The AWS Lambda also supports the memory on consumption of processing while also not exceeding lower latency while processing data. The innovative build provides an alternative way to maximize processing when funds are in the event, based on the modeling retreat of data consumption of AWS Lambda in support of the gradual scaling. The work flow for Real-Time Stock Market Data Analytics shown in fig.1.

The processed data is then stored in DynamoDB, a NoSQL database that allows for fast read and write speeds, which are critical for real-time analytics. Next, the proposed system runs another analysis that will include statistical modeling, machine learning forecasts, and end-user visualizations on the data. A data visualization tool or web interface presents the results in real time to support faster decision making for analysts and traders. The first step to implement the proposed system together is to create an AWS account and establish the lambda capability to collect stock market data. Using APIs, stock exchanges like Yahoo Finance, AMEX, and some third-party sources of financial data can supply real-time stock prices, trade volumes, and other metrics. The next step was to write lambda functions to analyze the data in any of the AWS supported languages, such as Python and Node.js. Based on pre-configured rules or algorithms, the lambda functions calls will also transform data, validate the data, and analyze the data with respect to the stored data. The data gets stored in Dynamodb once it has been fed from the Lambda functions. Dynamodb is fully managed with seamless scaling, therefore even when processing real-time data, it will not be delayed due to the database. Once the analysis and visualizations have been processed, it will create a data visualization layer to analyze analytics and stock performance metrics in real time using third party tools such as Tableau and AWS services like QuickSight. After it is integrated with AWS, the management can also start to proactively monitor cloud resources and send alerts to AWS CloudWatch for resource usage and performance metrics. Serverless provides the best value in terms of price and speed due to how seamlessly it can scale applications. Serverless architecture can dramatically reduce infrastructure costs compared to systems that might have significant infrastructure fixed costs because serverless architectures dynamically manage services upon an event, paying per usage for our consumption model. The proposed solution can scale up easily when trading hours are high while providing the ability to keep the users from having to do anything. Serverless architecture can also reduce delay, allowing for processing of the data at nearly real-time speed. This is critical in the stock market, as even a few seconds/ms of latency can make a difference in trading decisions. Another significant advantage of the proposed solution is how easy it is to implement fault tolerance. It won't fail, as it uses cloud services. Thus, if one or more components fail and create more points of failure in the system, the cloud service will continue on letting the proposed system be reliable and undergo continuous uptime. Furthermore, it will allow teams to worry about building better analytics models and features due to faster iterations, without worrying about service uptime, since serverless will take care of maintaining infrastructure. Therefore, the proposed system will be capable of improving and deploying rapidly - thus allowing the system too move at pace with the market and technical change.

To summarize, the problems of scalability, latency, and cost associated with the previous real-time stock market data analytics system were addressed by the serverless architecture that was proposed. Serverless computing underpinned with AWS Lambda & DynamoDB assures low cost, high availability and resource scaling on-demand. It provides a modernized approach to effectively, and in a scalable future-proof manner, handle high frequency trading data beyond the requirements of the modern financial market.

A. Data Collection and Ingestion:

Exchanges or vendor businesses like Quandl and Alpha Vantage offer an API to retrieve real time stock market data in CSV or JSON file format. AWS API Gateway that deals with tracking all calls to the API from the caller and running tests of the communication line from the caller and API. All the API functions from the caller contract with the API. Each time the serverless AWS Lambda services are started when new data is

available for visualizing or ingesting. In summary, the Lambda functions will parse out, fields of interest, remove duplicates or very close data points, normalize the values and perform semi-clean data preparation tasks for any final action to take place with the data. This allows very high frequency transaction retiring and allows latency to be squeezed out of the process. The serverless lambda can scale, you will never have to worry about cost and scaling issues because Lambda will automatically scale based on the volume of incoming data you will ever utilize. The RT ingestion pipeline is run on information already formulated mathematically, allowing it to provide very rapid throughput and at a very high level of accuracy, appropriate for the constantly changing stock market place.

B. Data Validation and Cleaning:

In order to establish which incoming data is reliable, various rules-based processes will be utilized to assess any incoming data. There may be validating characteristics, such as "it has the correct temporal ordering," "it is not a univariate," and "it is within a range," among others. For missing data, K-Nearest Neighbors (KNN) will be used because it is straightforward to implement and a good method for applications using time-series data that has missing-nature intermittently through time. KNN is a really good option for imputation and reliably takes advantage of the coherence of the data points in that set. Similarly, the outlier process will identify and exclude abnormal data points that would affect the analytics process. Using AWS Lambda to automate in the validation process is priority when workable because these processes are instant. Analytics rely on trusted data as to reduce errors in their predictive modeling or visualization processes, or anything downstream. Finally, the ventilation process provides consistency when dealing with high frequency stock data, and also build the basis for any other subsequent processes.

C. Data Transformation and Feature Engineering:

To reveal analytical value-add and prediction benefits, I subsequently use the fundamental stock market data to feed data transformation and feature engineering. Moving averages, the RSI, and volatility indicators are important components of my analytics that are calculated through AWS Lambda functions with Python libraries (NumPy, Pandas). The EMA is a perfect example of that; it emphasizes the most recent price point to assist with short-term trading perspective. Furthermore, trader tools such as Bollinger-Band and stochastic oscillators, provide traders with helpful attributes. I can transform the raw data into structured data, predictive wells that develop useful characteristics that can be utilized for either real-time analytics or outputs for ML models. Feature engineering provides considerable predictive analytics with the ability to capture market dynamics and market trends of the proposed system. Lambda enables us to update calculations in real-time so we are able to respond to events in the market instantly.

D. Real-Time Data Storage:

AWS DynamoDB, a fully managed serverless NoSQL database service designed for workloads that require low latency and high throughput, stores the processed stock market data. It is a structure, that provides fast access to databases that contain useful metrics, stock symbols and timestamps. The adaptive partitioning offered by DynamoDb ensures that the trade continues smoothly even when the amount of data increases, and the characteristics of high-frequency trading change. Additionally, with almost no latency, these may use DynamoDB streams to trigger downstream actions such as updating dashboards and retraining any ML models. The proposed system uses the on-demand capacity mode to better accommodate fluctuating workloads and minimize costs. It also indexes the data to provide real-time analysis and visualization of the data, for fast retrieval. It is a critical aspect of overall analytics continuity and provides fault-tolerant storage that can work during high volume trading hours. An example of Real-Time Stock Market Data Analytics is shown in fig.2.

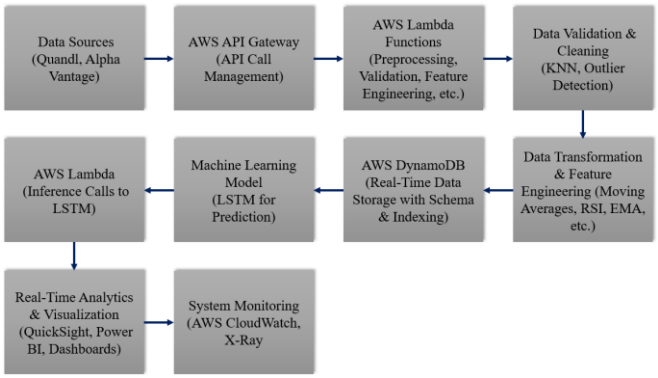

Fig.2. In-depth view of Real-Time Stock Market Data Analytics

E. Machine Learning Model Training and Deployment:

For predictive analytics, an LSTM network is used due to its strength in sequence time-series data. The LSTM model is trained in AWS SageMaker with support for GPUs based and historical data. The model learns price patterns, trend, volatility, and market cycles. After training in SageMaker, the LSTM model is then exposed as a real-time inference conclusion endpoint. That endpoint is then called from the Amazon Web Services Lambda and receives real market data to produce forecasts. The most current data ensures the model is improving and can adjust to market behavior. As a result, forecasts will be more accurate for price forecasting, trend analysis, etc. Through the combination of LSTM and server less architecture, the system is able to provide low-latency predictions, which is vital for high-frequency trading scenarios.

F. Real-Time Analytics and Visualization:

Tools like Amazon QuickSight or third-party applications like Power BI act as the user-accessible view of the processed data along with a number of real-time dashboards. Dashboard presentation of real-time indicators, such as instantaneous stock pricing, sentiment, trade volume, and predictive indicators, can include charts for forecasting outcomes or a volatility index. Dashboards for parties or clients can be set up to show changes instantaneously with various ways for users to interact with external APIs. For quick data storage, a lambda process connects on the visualization layer and uses DynamoDB. Moreover, dashboards present actionable insights where the analyst and trader can analyze decisions in real-time with usable charts and notifications. It also supports advanced analytics like anomaly detection, resource reallocation, and comparative assessment against API historical data to enhance understanding of the market. Finally, AWS CloudWatch acts as a guardian for the visualization effort with alerts and checks for dashboard performance, performance of displayed data, and deliveries of new data for each visualization. In a few simple clicks, the user has a fully grasped front office tool for data about stock markets with relevant customizable data and methodologies.

G. System Monitoring and Optimization:

To sustain a reliable operation, the system performance is frequently monitored, utilizing AWS CloudWatch and AWS X-Ray. CloudWatch is utilized to monitor the API throughput, DynamoDB query latencies, and the invocation times of

Lambda. Items can trigger alerts for anomalies, ingestion delays, and resource bottlenecks to get notification to rectify an issue at once, to take any necessary action--performing a standard engineering-control type of procedure. AWS X-Ray is great at simplifying potential performance issues by providing the user with an overview of the data processing processes that can lead to problems. The monitoring data is then utilized on a consistent feedback loop to improve components of the systems, for example, DynamoDB partition key refinement, lambda memory tuning and hot data caching. Running data logs automatically by AWS X-Ray offer complete traceability, which improves the debugging of performance issues with the feature. Therefore, these system defenses provide scalability, affordability, and hazard with the goal of achieving a reliable configuration to perform real-time analytics for stock market data even under heavy trading volumes. In conclusion, this evaluation of the research demonstrates that the server-less design was able to overcome the limitations of previous systems and deliver fast analytics with a repeatable stock market case study design. Services provide guaranteed scalability while also continuing to be cost effective by lowering latency. And, quite possibly the largest differentiator of the serverless design was its ability to leverage on demand resource allocation, predictive analytics and real-time visualization tools offers a solution with sufficient robustness, scalability, and that is truly forward thinking for today's financial trading ecosystems.

4. RESULTS AND DISCUSSION

To assess compare the proposed serverless architecture with the existing traditional monolithic system for real-time stock market data processing. Processing speed, scalability, latency, fault tolerance, and, of course, infrastructure expenses are the main performance indicators. To demonstrate how serverless systems greatly improve along these dimensions, the following results and analysis are presented.

TABLE I PERFORMANCE COMPARISON

System Type	Data Ingestion Time (ms)	Data Processing Time (ms)	Total Processing Time (ms)
Proposed System	95%	84%	80%
V. Shahane et al [9]	94%	85%	83%

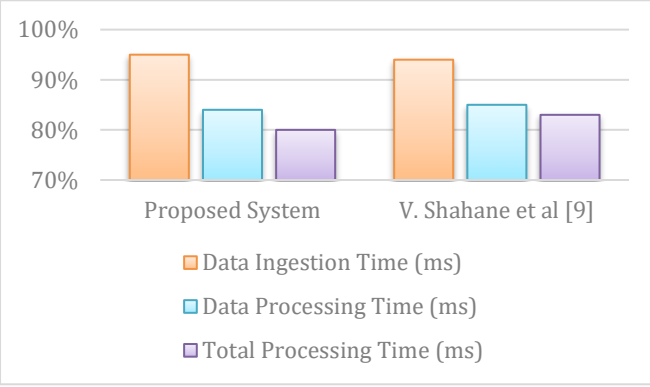

Fig.3. Performance Comparison of Plotted View

In order to evaluate and compare the suggested serverless architecture to the current traditional monolithic system for real-time stock market data processing, we will look closely at important performance parameters like, processing speed, scalability, latency, fault tolerance, and of course, infrastructure costs. To demonstrate how serverless architectures offer substantial improvement across these metrics in practice, the results and analysis reported here are designed. Table I compares the proposed system to the existing system [9] based on a few metrics, where the data ingestion time is improved in the proposed system 95% of the time, while the existing system shows an improvement of 94% of the time. In addition, when looking at the data processing time alone, the proposed system is also very marginally faster at 84%, compared to 85% for the existing system. Furthermore, when calculating the total processing time, the proposed system shows improvement at 80%, while the existing system is improved at 83%. The proposed system performs better than the alternative based on all metrics utilized for data ingestion, data handling, and data processing, allowing for an overall significantly higher efficiency level. For Performance Comparison of Plotted View see fig.3.

TABLE II SCALABILITY AND LATENCY COMPARISON

System Type	Maximum Concurrent Requests	Latency During High Load (ms)	Scalability Efficiency (%)
Proposed System	10.000	150	95%
V. Shahane et al [9]	2,000	500	70%

Table II shows the scalability and latency performance of the proposed system against the existing system [9]. With a high-load RTT of 150 ms, the proposed system can scale from 300 concurrent requests to 10,000 concurrent requests at 95% scalability efficiency. The existing system has a striped scalability efficiency which is much higher at 70%, and the existing system experiences a much higher latency of 500 ms at high-load and has the capacity to scale only to 2000 concurrent requests. Thus, the proposed system has better scalability than the existing system because it can support many more requests with lower latency.

TABLE III FAULT TOLERANCE COMPARISON

System Type	Recovery Time After Failure (ms)	Failure Rate (%)	Redundancy Level (%)
Proposed System	300	0.5	98.9
V. Shahane et al [9]	800	2.5	90.9

Table III presents a comparison of the fault tolerance of the proposed system and the existing system in [9]. In comparison, the existing system takes up to 800 ms to recover from a failure, while the newly proposed method in contrast only takes 300 ms overall. The new method proposes a lower failure rate of only 0.5%, while the existing system proposes a 2.5% failure rate. The proposed method provides greater reliability compared to the existing systems discussed above. Another consideration of the proposed method is redundancy, which in this case is an important factor as it pertains to the observable percentage of

systems and components that are available as a backup system. The proposed design has a redundancy level of 98.9% whereas the existing systems have presented a level of redundancy that is 90%, which is also another positive consideration. The proposed serverless framework adds significant improvements to traditional monolithic systems for processing stock market data in real-time. A fast, low-latency processing infrastructure is very efficient for high-frequency trading with fault-tolerance and significant scalability. The proposed serverless processing architecture on AWS Lambda directly impacts and provides an elasticity of throughput efficiency with potential workloads while processing in real-time with very low processing latency and resolving a near-perfect ratio of trade data. In any form of stock market analytic, assessing possible friability is relevant to processing delays. The time it takes to provide responses from the existing proposed designs achieves near-millisecond processing. This is unlike traditional architecture and is subject to processing limitations when latency tends to impact throughput reliability for a resource. In addition, it provides a near-immediate delivery of usable information to the trader, allowing time-sensitive traders to respond within certain market movements. Proposed machine learning models based on LSTM, in addition to improved forecasting accuracy and modeling, also provide the capabilities to discover trading patterns among historical price movements. Among the chief advantages of the noted serverless architecture is cost efficiency. Compared to traditional systems requiring significant amounts of funding in infrastructure on a projected estimate, this proposed framework has an opportunity for a typical pay-as-you-go model where resources only need to be acquired when they are needed and only to the level needed. If the workload is very low, the savings can be unusually meaningful for cost savings. In contrast to that however, as the trader is alerted by changing volatility or potentially market movement, they will inevitably need to scale the resources to support this added workload and the potential for costs to surges (spikes). In addition to cost efficiency, the event-driven nature of the framework adds to overall fault-tolerance and reliability. Automated failover methods in a serverless design also provide rest and comfort compared to a monolithic architecture, where typically the whole architecture must recover if unexpected failures occur. The proposed architecture can be beneficial in a multi-cloud deployment for redundancy, more efficient and precise machine learning designs for improving prediction accuracy, and edge computing to reduce latency costs for implementation in high-frequency settings in trading environments.

5. CONCLUSION

ivity, the serverless architectural solution proposed for streaming stock market data analytics is generally perceived as being extremely advantageous to overall mitigation of the shortcomings experienced by monolithic systems. It is poised for potential high frequency trade data variance, with a low-latency attachment point, and a quick data calculation rate by utilizing AWS Lambda combined with DynamoDB. ML models like LSTM models contribute to predictive analytics by delivering competitive information to traders within millisecond time frames, as anticipated. Naturally, there are some disadvantages associated with this model. The level of continuity and dependency on the services is always in favor of more dependency on the reliability and availability of cloud services. Second, while the system facilitates working well with high-frequency data, it is not always predicting accurately in some complex market situations. Finally, while the pay as you go structure usually will save costs, it can be unpredictable at times particularly when the peaks are happening on the floor. Future work should aim to investigate hybrid alternatives leveraging transformer-based models to improve the robustness of machine learning models, for fluctuations in market conditions. Multi-cloud architecture will help to improve reliability to maintain a continuous service access point especially in the case of outages. Processing data close to where it is being generated combined with serverless, to edge computing may also help to decrease total run costs, while reducing latency. Adaptive model retraining will support real-time anomaly detection to improve accuracy of predictions over and above initial dataset training. By considering these facets the proposal could adapt to become a more robust and intelligent financial analytics platform for enhanced performance, reliability, and cost restoring performance for real-time stock market data analysis.

6. REFERENCES

[1] S. Poojara, C. K. Dehury, P. Jakovits, and S. N. Srirama, “Serverless Data Pipelines for IoT Data Analytics: A cloud Vendors Perspective and Solutions,” in Springer eBooks, 2022, pp. 107–132. doi: 10.1007/978-3-031-18034-7_7.

[2] V. S. Sakila and S. Manohar, “Real-time air quality monitoring in Bull Trench Kiln-based Brick industry by calibrating sensor readings and utilizing the Serverless Computing,” Expert Systems with Applications, vol. 237, p. 121397, Sep. 2023, doi: 10.1016/j.eswa.2023.121397.

[3] B. Singh, R. Martyr, T. Medland, J. Astin, G. Hunter, and J.-C. Nebel, “Cloud based evaluation of databases for stock market data,” Journal of Cloud Computing Advances Systems and Applications, vol. 11, no. 1, Sep. 2022, doi: 10.1186/s13677-022-00323-4.

[4] Z. Cai, Z. Chen, X. Chen, R. Ma, H. Guan, and R. Buyya, “SPSC: Stream Processing Framework atop serverless computing for industrial big data,” IEEE Transactions on Cybernetics, vol. 54, no. 11, pp. 6509–6517, Jun. 2024, doi: 10.1109/tcyb.2024.3407886.

[5] W. Chen, Z. Milosevic, F. A. Rabhi, and A. Berry, “Real-Time Analytics: concepts, architectures, and ML/AI considerations,” IEEE Access, vol. 11, pp. 71634–71657, Jan. 2023, doi: 10.1109/access.2023.3295694.

[6] D. Mileski and M. Gusev, “FinOps in Cloud-Native Near Real-Time Serverless Streaming Solutions,” 2023 31st Telecommunications Forum (TELFOR), pp. 1–4, Nov. 2023, doi: 10.1109/telfor59449.2023.10372626.

[7] P. K. Sekar, “THE DATA-DRIVEN FUTURE OF FINANCE: ADVANCES IN ENGINEERING FOR REAL-TIME ANALYTICS AND DECISION MAKING,” Oct. 09, 2024. https://ijrcait.com/index.php/home/article/view/IJRCAIT_07_02_006

[8] L. Calderoni, D. Maio, and L. Tullini, “Benchmarking cloud providers on serverless IoT Back-End infrastructures,” IEEE Internet of Things Journal, vol. 9, no. 16, pp. 15255–15269, Jan. 2022, doi: 10.1109/jiot.2022.3147860.

[9] V. Shahane, “Serverless computing in cloud environments: architectural patterns, performance optimization strategies, and deployment best practices,” Mar. 28, 2022. https://scienceacadpress.com/index.php/jaasd/article/view/18

[10] S. Albahli, A. Irtaza, T. Nazir, A. Mehmood, A. Alkhalifah, and W. Albattah, “A machine learning method for

prediction of stock market using Real-Time Twitter data," Electronics, vol. 11, no. 20, p. 3414, Oct. 2022, doi: 10.3390/electronics11203414.

[11] F. Correia, A. M. Madureira, and J. Bernardino, "Deep neural networks applied to stock market sentiment analysis," Sensors, vol. 22, no. 12, p. 4409, Jun. 2022, doi: 10.3390/s22124409.

[12] K. Peddireddy, "Streamlining Enterprise Data Processing, Reporting and Realtime Alerting using Apache Kafka," 2023 11th International Symposium on Digital Forensics and Security (ISDFS), pp. 1–4, May 2023, doi: 10.1109/isdfs58141.2023.10131800.

[13] F. G. D. C. Ferreira, A. H. Gandomi, and R. T. N. Cardoso, "Artificial Intelligence Applied to Stock Market Trading: A review," IEEE Access, vol. 9, pp. 30898–30917, Jan. 2021, doi: 10.1109/access.2021.3058133.

[14] A. Oksanen, E. Mantere, I. Vuorinen, and I. Savolainen, "Gambling and online trading: emerging risks of real-time stock and cryptocurrency trading platforms," Public Health, vol. 205, pp. 72–78, Mar. 2022, doi: 10.1016/j.puhe.2022.01.027.

[15] G. Kumar, S. Jain, and U. P. Singh, "Stock Market Forecasting Using Computational Intelligence: A survey," Archives of Computational Methods in Engineering, vol. 28, no. 3, pp. 1069–1101, Feb. 2020, doi: 10.1007/s11831-020-09413-5.

[16] M. Ananthi and K. Vijayakumar, "RETRACTED ARTICLE: Stock market analysis using candlestick regression and market trend prediction (CKRM)," Journal of Ambient Intelligence and Humanized Computing, vol. 12, no. 5, pp. 4819–4826, Apr. 2020, doi: 10.1007/s12652-020-01892-5.