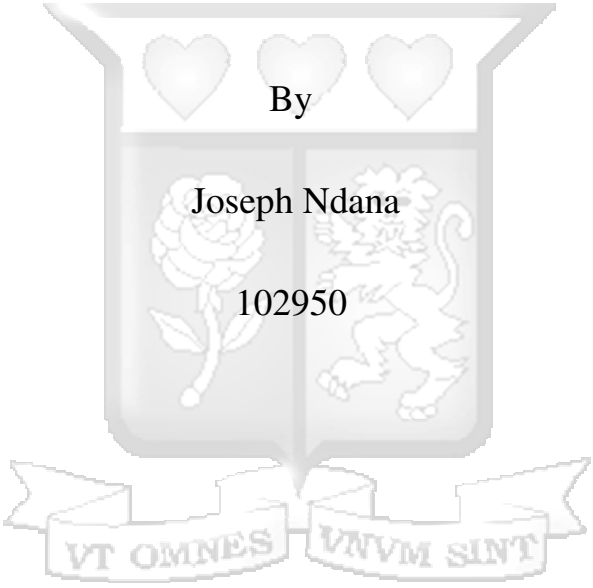
2023

Retailer stock levels optimization tool.

Ndana, Joseph
School of Computing and Engineering Sciences
Strathmore University

Retailer Stock Levels Optimization Tool



By

Joseph Ndana

102950

Master of Science in Computer Information Systems

2023

Retailer Stock Levels Optimization Tool

By

Joseph Ndana

102950

Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Information Systems at Strathmore University

School of Computing & Engineering Sciences
Strathmore University

Nairobi, Kenya

July, 2023

Declaration and Approval

Declaration

I declare that this work has not been previously submitted and approved for the award of a degree by this or any other University. To the best of my knowledge and belief, the dissertation contains no material previously published or written by another person except where due reference is made in the dissertation itself.

Student's Name: Joseph Ndana

Sign: _____ Date: 07 JUNE, 2023

Approval

The dissertation of Joseph Ndana was reviewed and approved for examination by the following:

Dr. Allan Omondi
School of Computing & Engineering Sciences,
Strathmore University

Dr. Julius Butime,
Dean, School of Computing & Engineering Sciences,
Strathmore University

Dr. Bernard Shibwabo,
Director of Graduate Studies,
Strathmore University

Abstract

The purpose of this research was to design a statistical model that allows a retailer to optimize stock levels based on stock related parameters such as demand, lack of stock, stock replenishment lead time, service level, maintenance cost, and costs of replenishing stock. The study adopted applied research to solve the business challenge on optimization of stock. The study utilized secondary historical data relating to stock obtained from a supermarket in developing the optimization model. Additionally, the study applied prototyping methodology to design, develop and test the prototype. The web application was developed using HTML, JavaScript and Java Server Pages in Netbeans IDE. The server applications were developed based on Java Programming language. Apache Kafka was used for ingestion. Spark was used for data streaming while YugabyteDB was used for storage. The model, based on parameter values of a high moving product, yielded a service level at 95.2%. This was a positive indicator that the retailer would not hit a stock-out during the subsequent replenishment cycle for this product. On a probability scale of 0.01 to 0.99, the probability of running out of stock was 0.048. The model yielded an optimum order quantity of 15 units, against an average of 17 units on supplies made. Moreover, the model computed an optimum safety stock level within the 10-20% range of 14 units, which allows the retailer to cater for varying vendor delivery periods, as well as meet the changing consumer demands. Based on these values, the model computed an optimum stock level of 21 units, which allows the retailer to only reorder when the cycle stock, computed at 7 units, nears depletion. Similarly, the retailer can further inform decision making in the reorder placements based on the computed average lead time, such that the delivery is made every 6 days. The model was further tested on another high moving product, yielding a service level of 96.5%, implying that approximately 96% of the periods the model is able to cover for the customer demand for the given product.

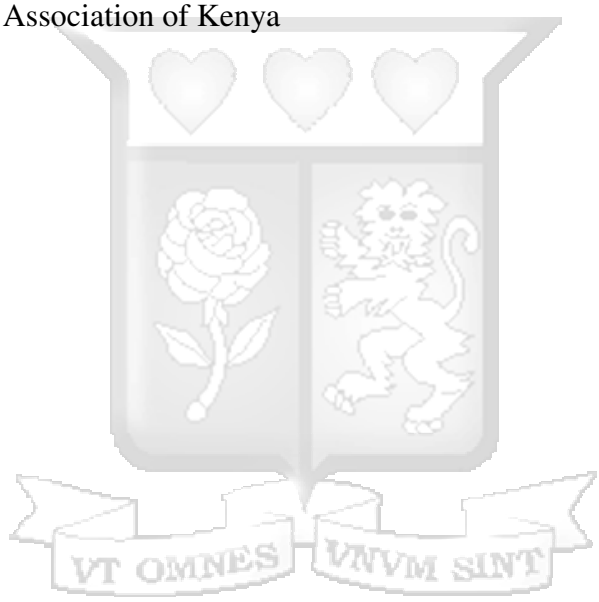
Table of Contents

List of Figures

List of Tables

List of Abbreviations

ACID	Atomicity, Consistency, Isolation, Durability
APIs	Application Programming Interfaces
BI	Business Intelligence
DAO	Database Access Object
DW	Data Warehouse
HDFS	Hadoop Distributed File System
KS	Knowledge sharing
NOSQL	Not Only SQL
POS	Point of Sale
Retrak	Retail Trade Association of Kenya



Chapter 1: Introduction

1.1 Background to the Study

Optimization of stock levels is important if firms are to minimize the incurred overall costs (Tarigan et al., 2020). They assert that information relating to demand and supply is vital for both retailers and suppliers in maintaining an optimized stock level. Further, to replenish the retail stock based on the demand, retailers need excellent support from the supplier, and that by using a fast and secure communication system, this support can be achieved. In the long run, firms increase their revenue (Olow et al., 2020). Subsequently, customers benefit from low prices accruing from low transactional costs incurred by retailers and suppliers. Categorizing information and arranging it in an appropriate way enhances sharing of information through various communication technologies (Carrim et al., 2020). Additionally, meaningful knowledge can be derived from resources, such that uncertainties in demand and supply can be addressed appropriately.

Financial resources are tied down by surplus inventory, while stock-outs cause loss of goodwill and potential revenue (Ongbali et al., 2019). They further contend that in order to strike and maintain a balance between satisfying the demand and the inventory safety stock, the optimal policies of inventory must be determined. According to Lukitosari & Subriadi, (2020), inadequate or excess stock is caused by demand that is unstable, where the surplus stock accumulates and eventually becomes obsolete. Furthermore, they contend that stacked goods are then sold at throw-way prices or stored in warehouses until there is demand. According to them, it is necessary to harmonize the uncertainty in supply and demand so as to achieve the optimal quantity to be ordered. Consequently, overstocking or understocking has adverse effects on the revenues of both retail buyers and suppliers.

Maintaining an optimum stock level therefore, ensures there is neither surplus nor inadequate stock. Further, the stock levels should meet the customer demand, while minimizing operational costs related to stock such as cost of storage and delivery costs. Additionally, surplus inventory results in improper utilization of resources that would otherwise be used for more profitable activities. Likewise, inadequate stock leads to loss of revenue as opportunities arising from high demand are not explored.

The study by Ochelle et al., (2017) and Ndwiga and Kiarie, (2017) recommended adoption of technology that incorporates stock optimization models to manage stock. Olang'o, (2018) recommended that the various healthcare management teams need to ensure the availability of data consumption and inventory control tools in order to help capture consumption data to aid in accurate demand forecasting and requisition process. The MOH HPT strategic report identifies the need to develop commodity-based dashboards that

enhance decision making with respect to the levels of stock and efficient redistribution of drugs across the facilities.

To design the systems that manage optimizing and replenishment of stock, there are two questions considered: when to place the order and the appropriate quantity to order (Krzyzaniak, 2017). Further, to solve these questions, one must derive the parameters that control replenishment of stock. In addition, these parameters are related to demand and factors that influence the stock replenishment process such as random variability of demand and lead time, the service level, costs related to replenishment, cost of stock maintenance and inadequate stock. Moreover, these parameters can be derived from historic data related to lead time and demand by computing their average and standard deviation, identifying their limitations and the imposed requirements during their processing.

This study sought to automate the optimization of stock levels in a retailer premise with an installed Point-of-Service system, based on a stock optimization model. The model computes the parameters that control stock replenishment from the historical data related to stock levels and replenishment. The historical data for this study is obtained from a mini-supermarket. The model then computes the optimum quantity to order, the optimum safety stock, the optimum cycle stock and the optimum stock level. Consequently, the retailer can determine when to place order once the cycle stock level falls below or approaches the depletion of the cycle stock. In addition, the model computes the optimum service level for a particular product, ensuring that the retailer does not hit a stock-out during the subsequent replenishment cycle. Furthermore, the retailer can determine if s/he is overstocking by evaluating the service level, such that service levels above a hundred percent would imply over-stocking.

1.2 Problem Statement

Based on the research by Ooms et al., (2020), Ochelle et al., (2017), Sporta, (2018), Ndwiga and Kiarie, (2017), Makori et al. (2016), Olang'o, (2018), Shajema, (2018) and Muhoza et al., (2021), the information relating to stock levels is not analyzed appropriately for optimization of stock levels, resulting in stock outs or overstocking. Consequently, their research, as well as the report from the Ministry of Health Strategic plan 2020-2025, indicate that overstocking increases the cost of operations, requiring more resources to transport commodities to the supplier and retailer premises. Furthermore, overstocking consumes more space and therefore increases storage costs. Additionally, overstocking would demand more human resources for efficient management thus increasing human resource costs. Correspondingly, understocking leads to increased cost as a result of under-utilization of available resources in a timely manner. Additionally, retailers have to meet the costs of under-utilized space, human and transport resources. Moreover, in the event that demand rises, retailers have to react in order to meet the demand. This reaction

requires sudden increase in resources with respect to transport, storage and human resources to sustain delivery performance. Likewise, the sales revenue for retailers decreases due to delayed delivery of commodities in high demand. As a result, retailers increase commodity prices to meet their operational costs such as storage space costs.

The risk of either overstocking or understocking can be mitigated by ensuring retailers derive meaningful insights from stock related information, that will enable them to optimize their stock levels, and thus maintain efficient delivery of goods. To maintain an optimum stock level, stock related data should be analyzed appropriately, so as to compute the stock-related parameters that control replenishment of stock. These parameters would include demand, cost of stock replenishment, stock deficit cost, purchase price, stock-carrying cost coefficient, service level, unit quantity of the assumed constraint and the lead time. Based on the Economic Order Quantity (EOQ) theory (Shajema, 2018), a high demand, cost of stock replenishment, stock deficit cost, service level and pro-longed lead time would increase the optimum order quantity. Likewise, a high stock carrying cost coefficient, purchase price and the unit quantity of the assumed constraint would reduce the optimum order quantity. Subsequently, the optimum order quantity would influence the optimum cycle stock, such that order quantity is directly proportional to cycle stock.

1.3 Research Objective

The aim of the study was to design a statistical model that allows retailers to optimize their stock levels based on stock related parameters. The study was based upon the following specific objectives:

i. To review the causes of limited stock optimization.
ii. To review the existing models of stock optimization and the existing supporting technologies in information ingestion, storage and streaming
iii. To develop a stock optimization model based on stock-related historical data
iv. To test the model for validity and reliability

1.4 Research Questions

i. What are the causes of limited stock optimization?
ii. What are the existing stock optimization models and supporting technologies that support information ingestion, streaming and storage?
iii. What is the required historical data needed to develop the stock optimization model?
iv. How will the functionality of the model be validated and tested for reliability?

1.5 Significance of the study

Retailers will mitigate the risk of high transactional cost arising from either overstocking or understocking. Consequently, Retailers will benefit from improved delivery performance, lowering transactional costs by accurately responding to the market demand, therefore avoiding wastage of resources. Consumers will enjoy lower prices of commodities accruing from reduced retailer transactional costs. The system will allow further research related to demand and supply by providing access to quality data, as well as real time sales data emanating from retail buyers.

1.6 Scope of the study

This study was limited to retailers. The research was conducted in urban area due to limited resources and the assumption of uniformity in the supply of commodities across various retailers. In addition, most retailers stock products from the same supplier, and vice versa. The research was limited to the stock-related information that support control of the stock levels and the flow of goods at a retailer premises. In addition, the research did not explore how overstocking or understocking under the supplier affects the retailer performance. The study applies only to a retailer that has installed a Point-of-Service system.

Chapter 2: Literature Review

2.1 Introduction

This chapter reviewed various theories, models, architectural design and existing systems related to information sharing and stock optimization. Additionally, the chapter reviewed the current causes of lack of stock optimization and possible solutions. In particular, the Information Processing Theory was reviewed in regard to utilization of information to gain insights within organizations. Additionally, various existing theoretical models of stock optimization were reviewed. The EOQ model, ABC technique, and Just-in-Time approach were reviewed, including their limitations towards predicting stock optimization. Moreover, various statistical models for calculating safety stock were assessed, as well as replenishment methods for stock levels. Lastly, the optimized order quantity based on Lagrange function was evaluated.

In addition, various architectural designs were evaluated to assess the suitability towards developing a prototype. The Hexagonal, monolithic, microservice, Lambda, Kappa, Solid, Bolster and Polystore architectures have been discussed, including the comparison of each towards meeting the Big Data Architectural requirements. However, it was noted that the suitable architecture is based on the requirements of the use case.

Moreover, some of the existing similar systems were discussed, as well as the observed limitations of each. Eventually, the conceptual model of the study was illustrated, showing the high level integration of the proposed system.

2.2 Information Processing Theory

The general model of IPT postulates that the performance of organization depends on the level at which it dynamically integrates information to information processing (Kmetz, 2021). Further, this integration can be described within the cybernetic context, by how an organization, through flow of information, links its internal environment to its external environment. It was advanced by Galbraith in 1973, stating that the main task of an organization is to process information, and that their performance is dependent on the amount of information they have. In addition, he submitted that the basic proposition is that the uncertainty of a task is directly proportional to the required information to be processed at the execution stage by the decision makers. This implies that the preplanning of an activity is dependent on the level of understanding before its execution. Further, a lack of understanding implies that more information processing is needed during execution to derive more knowledge, which results in disruption in the pre-planned strategies.

Additionally, organizations limit their ability to pre-plan for activities prior to their execution when impeded by uncertainty. Therefore, it is hypothesized that organizations are characterized by their

strategies towards their capability to pre-plan prior to task execution, their response capability to adjust to unplanned tasks, and their performance level in cases of variability. Further, the choice of a strategy depends on its cost implication. Therefore, the framework adopted should be capable of identifying the required strategies and associated costs. The function of the framework is to identify these strategies and their costs. Therefore, organizations that have timely access to more information have a better competitive advantage. Also, they aver that beyond a certain level, information does not have a positive impact on performance. However, organizations can still manage sharing of information to their advantage.

2.3 Economic Order Quantity Theory

EOQ theory was created by Ford Harris in 1913 and is an inventory management approach that tries to minimize total inventory holding costs and ordering costs, according to Shajema, (2018) analysis of the EOQ model. Additionally, the Economic Order Quantity model of inventory management is used to identify the ideal delivery size and select the least expensive carrier, which ensures that the overall cost of investments in inventories is kept to a minimum. He claims that the EOQ model is a method that establishes the ideal quantity of inventory to order each time the stock of that item runs out. In deciding the quantity to utilize while refilling item stocks, the EOQ model also takes into account the trade-off between ordering cost and storage cost. Also, a greater order size lowers the frequency of orders and, thus, the cost of orders, but it also necessitates keeping a bigger average inventory, which raises the cost of holding. A smaller order amount, on the other hand, results in a reduction in average inventory but necessitates more frequent ordering and greater ordering costs. Although the model is effective in minimizing inventory costs, it encourages large order stocks to reduce ordering frequency, which provides a contradiction as to whether the storage costs actually lowers the total transaction costs in the long run. It also downplays the role of buffer stock that caters for variations in lead-time and demand, thus making it unsuitable in practice.

2.4 Retailers

A retailer is a business that trades commodities or services to consumers for their utility (Levy et al., 2012). Further, they state that retailers play a key role in a supply chain by linking consumers to manufactures. Additionally, most supply chains feature some vertical integration, such as a retailer engaging in wholesale activities by operating its own chain stores. According to them, retailers can be categorized based on size, merchandize and services offered as follows. Additionally, a conventional supermarket is characterized by its large size and self-service, stocking products ranging from food to general merchandise. Similarly, a supercentre is typically a supermarket combined with a discount store. Likewise, a warehouse club is a retailer that offers an irregular and limited assortment of products at cheap prices with little service, targeting small businesses and specific consumers. Additionally, convenience stores provide quick service at a convenient location, such as stores in gasoline stations. Also, department stores are retailers that offer unique service to customers, with their stores organized

into departments that are distinct for displaying a particular merchandize, such as apparel and bedding or furniture stores. Specialty stores provide a high level of service focusing on a unique brand or product category such as drugstores.

According to them, a retail channel is the manner in which a retailer channels the merchandise and services to its clients, commonly through a store. Additionally, they assert that retailers also use other channels such as the internet, catalogues mailed to customers, direct selling through sales people or automated retailing such as automatic vending machines. Although some retailers stock their products directly from manufacturers, the small scale retailers rely on wholesalers. Levy et al., (2012) assert that wholesalers buy large quantities of goods and then resell them to other businesses or retailers in smaller quantities. Furthermore, the submit that retailers and wholesalers may have similar functions however, wholesalers meet the retailers' needs, while retailers meet the consumer needs. Additionally, they aver that some wholesalers function as both retailers and wholesalers, including supplying merchandize to other businesses. Consequently, the wholesalers act as the suppliers to the retailers.

Kerin et al., (2002) assert that retailers can be classified based on ownership, level of service or merchandise line. According to them, the most common form of ownership is the *Independent Retailer*, accounting for the small scale retailers. Additionally, *Corporate Chain,* a second form of ownership, involves numerous outlets under a single ownership, with centralized decision making structure. Further, *Contractual Systems* involve stores that are owned independently that band together form a chain-store such as franchises. *Full service, Limited service* and *Self-service* such as specialty stores form the categories under Level of Service.

Obayi et al., (2017) submit that in recently, power has shifted from manufacturers to retailers, as retailers adopt capital-intensive and advanced retail models such as e-commerce and superstores. As a consequence, they contend that this power shift has led to re-defining the role of retailers in buyer-supplier relationships, prompting a paradigm shift in the supply chain strategies. Therefore, they assert that to achieve stock optimization, retailers need to harmonize their operations to the demand and supply. Further, they contend that this balance is plays a critical role in maintaining customers, due to time-based competition. Additionally, they hold that shoppers prefer availability over brand and price based competition.

The Kenya Retail Industry Outlook Survey 2020 by retrak indicates that online sales have increased, indicating a rise in the adoption of e-Commerce. Further, 42.4% of retail businesses had no existing e-Commerce store, of which 39.4% considered a combination of in-store and online presence in the next 18 months, of which 30.3% currently implement an e-Commerce platform, among which 3% planned to implement in the next 6 months; while 6.1% planned to implement in the next 18 months. Notably, among the interview respondents, more than half (57.58%) already had an e-Commerce platform. The

survey portrays a positive trend in adoption of technology within the retail industry, which is vital in supporting information sharing with suppliers.

2.5 Supplier's Influence on Service Level

According to Craig et al., (2016), it is important that suppliers comprehend the effect of varying the service levels to orders placed. Moreover, they assert that inventory service level is directly proportional to the changes in supply chain process and technology required. In addition, they submit that although the costs arising from incrementing service level is measurable, the benefits are not directly measurable. According to them, by increasing the service level a supplier can regain lost sales as well as alter the demand received from retailers. Following their research, they found out that stockouts occurring at the supplier end have a short term positive and long term negative effect on demand. Further, they observed that the retailers can acquire more information about the service level of the supplier, by frequently placing orders, thus creating a strong correlation between the retailer demand and the historical service level. Further, they submit that the service level plays a key role in the retailer's product portfolio performance. According to them, retail buyers can reduce the overall inventory if suppliers increase the service level, thus improving the shopping experience while still maintaining a lower safety stock. Additionally, they assert that retailers can better plan in-store sales promotions if the supplier maintains a consistent availability of a product.

2.6 Retailer-Supplier Information Sharing

Recently, the retailer-supplier relations have been strengthened by deeper cooperation and sharing of information (Choudhury & Mahata, 2021). As a result, the supply chain has taken a two-echelon structure. According to Obayi et al., (2017), the close proximity of retailers results in a higher visibility to their operations in comparison to the producers. Consequently, retailers have fast-hand information on their consumer demand patterns, and are better placed to respond to changes in demand and supply chain disruptions. Therefore, they have an additional responsibility to collate, integrate and share such information with suppliers, from which knowledge can be derived. Additionally, they submit that the derived knowledge is essential for forecasting distribution and formulating strategies.

According to Shajema, (2018), information is a less expensive inventory as compared to the inventory asset. He contended that reliable, consistent, accurate, and timely information leads to less inventories, cost reduction and faster deliveries of products. However, this fails to show how the cost of sharing of such information with suppliers, influences the transaction costs. Further, he submitted that recently, firms have improved their management of inventory by improving their cooperation with other firms within their supply chains. He contended that by sharing information, they are able to reduce demand variability and thus reducing risks arising from variable demand. Additionally, he asserted that overstocking or understocking, which is caused by poor inventory management practices, impacts negatively on financial performance. However, his argument fails to incorporate the role and influence

the supplier has in stock inventory at the retail stores. In as much as such poor practices can affect stock levels, the role of a supplier contributes significantly to stock optimization, and thus a retailer cannot be treated in isolation.

Tongya et al., (2021) asserted due to the higher value of real-time data, there has been a paradigm shift towards adoption of technologies that support such data. They further state that real-time or stream data, can be described as unbounded data that is continuously produced from numerous sources of data simultaneously, in a time-ordered fashion. Furthermore, the data has to be incrementally processed in an event-driven structure. As a result of exponential growth in both heterogeneous data and rate of data generation, applications that analyzed Big Data in batches have had to shift to stream processing (Ovidiu-Cristian et al., 2018). Consequently, meaningful insights can be derived within shorter periods.

According to Obayi et al., (2017) companies manage volume, variety, and delivery lead time flexibility with cutting-edge information technology solutions. However, they point out that the variables influencing the exchange of knowledge between buyers and suppliers for the efficient application of flexibility techniques have largely gone unexplored. According to them, the term "transactive memory systems" refers to a strategy used by the companies to jointly encode, store, and retrieve meta- and important knowledge. Additionally, they aver that the degree to which organizations can coordinate their technological, technical, and social-cultural systems with their partners is often said to be a measure of organizational interoperability.

According to Hou et al., (2022), information exchange between retailers is significantly impacted by the presence of supplier encroachment. First, they argue that a shopkeeper may, in certain situations, voluntarily exchange demand information with her supplier. They contend that information sharing benefits the retailer when the quality cost coefficient is low, provided the supplier does not infringe. Additionally, when the quality cost coefficient is moderate, the retailer wants to share demand information with the supplier in cases where the supplier encroaches. Second, they contend that under supplier encroachment, the retailer has a greater motivation to disclose demand information than under non-encroachment since the supplier can alter the product's quality depending on his prior assumptions or the retailer's provided knowledge, which increases market demand. As such, so as to achieve a win-win situation, the retailer, in response to the supplier's effort, is more inclined to provide information. Thirdly, they demonstrate that when the quality cost coefficient is large, the supplier prefers to create a direct selling channel. Despite their study showing the significance of information sharing, it does not consider the limitations based on various market structures where encroachment is unattractive to a supplier.

Huang et al., (2017) submit that due to its distance from the end market, the supplier finds it difficult to monitor the information on demand. In such a situation, they assert a potential resolution in which the supplier bases decision-making on the retailer's superior demand knowledge. Furthermore, they assert

that the development of electronic data interchange technology makes it much easier for retailers to communicate with suppliers about the market. They further claim that, according to surveys conducted by BearingPoint Management and Technology Consultants, the frequency of communications between retailers and vendors is on the rise: More than 62 percent of American retailers said they spoke with suppliers on demand projections at least once a week. Similarly, they further state other reports also indicate that the majority of U.S. food merchants are sharing their daily and even weekly store sales and other data directly with their suppliers. Additionally, sharing activities have also been seen at other shops like Target and Walmart.

Notwithstanding, the shared information for this study would be limited to the model parameters. Additionally, it should be noted that the model would compute the values to the parameters based on historical data within either the retailer premises, augmented by information related to stock which is supplied by the retailer. Furthermore, the exchange of information happens only when the supplier supplies goods to the retailer, in which case the retailer is able to register this information. Also, the information exchanged is limited to the particular goods supplied by a supplier to a retailer. Therefore, information shared is restricted between a retailer and a supplier with respect to the goods supplied. Consequently, the critical information to be shared would only pertain to the cost of replenishment.

2.7 Research on Stock Optimization Challenges in Kenya and Southern Africa

The research by Ooms et al., (2020) on reproductive health commodities in Southern and East Africa found out that every month, the stock-out days on average ranges from three days in the private and private not-for-profit sector in Kenya, to twelve days in public sector in Zambia. Similarly, Ooms et al., (2021) observed that there were stock-outs in the snakebite commodities, further affected negatively by the COVID-19 pandemic. Another study by Ochelle et al., (2017) observed that the various sugar manufacturing firms in western Kenya have been grappling with the challenge of inventory control, evidenced by increased volumes of obsolete stock due to poor inventory control practices, resulting in poor procurement function performance. Likewise, the challenges identified through the study by Sporta, (2018) on Kenya Medical Supplies Agency, include obsolete products due to overstocking, under-stocking and poor inventory management.

The Ministry of Health of Kenya Health Product and Technologies(HPT) Supply Chain Strategy 2020-2025 acknowledges that there are missing information in quantification of HPT, overstocking and understocking. Consequently, this has contributed significantly to obsolete stock, additional costs, and disruptions in health services. Furthermore, there lacks sufficient quality data in national and county level on available stock, consumption, and service statistics. As a result, there is low accuracy and reliability of quantification estimates for HPT. For this reason, there is poor and uninformed decision making in procurement strategies. In addition, the Kenya Health Facility Assessment (KHFA) also revealed that for the 18 tracer medicines, there is a 44% stock-out rate in Health facilities for 7

consecutive days in a month in Financial Year (FY) 17/18. Besides, although there some HPT have Logistics Management Information System (LMIS) designs, data collection and sharing are inadequate. Moreover, data management based on the information system, related to consumption and stock levels, still poses a big challenge. In addition, the effectivity of the planning and decision making in HPT supply chain is jeopardized due to inaccurate and untimely data of low quality.

The study by Ndwiga and Kiarie, (2017) on retail chain store in Nairobi similarly observed that there have been significant cases of materials overstocking, understocking or lack of it that results in obsolete stock and subsequent losses incurred by supermarkets. In addition, they hold that supermarkets as a retail chain strive to have the lowest level of inventory possible but still be able to respond to customer demands. In the recent past, they highlight the challenges faced by Uchumi Supermarket ranging from stock-outs to overstocking, to the extent of losing 250 Kenyan Shilling in revenue in its various outlets, and incurring 1.9 billion shillings in warehouse charges due to overstocking. They observed that the major retail chain stores in Nairobi have signed mutual contracts with suppliers, where the suppliers supply directly to the shelves. However, this has introduced a number of challenges to the organization in optimization of stock. A similar study by Makori et al. (2016) on real time information within supermarkets in Nairobi observed that the supermarket retail industry is being saturated due to urbanization and subsequent growth of small scale retailers. Consequently, this has resulted in high competition and therefore higher uncertainties in demand. Uncertainties in demand lead to challenges in optimizing stock levels.

Olang'o, (2018) through his research on stock out of HIV Rapid Test Kits in Kenya concluded that, frequent stock out of HIV RTKs in healthcare facilities in Kenya leading to interruption of services to clients, is caused by the poor inventory demand forecasting practices among the healthcare workers. The study additionally indicates that an assessment conducted of Kenya health commodity supply chains within the vertical programs found out that the major causes of chronic stock outs of health commodities in healthcare facilities are high wastages, poor reporting rates, gap in inventory management practices and lack of supervision, monitoring and evaluation at all levels of health facilities. Another study by Muhoza et al., (2021) on contraceptive stock-outs observed that as a result of lack of stock, individuals are unable to request for their preferred choice of contraceptive, thus re forced to choose other methods that do not align with their needs or preference. They further indicate that there is insufficient data on stock-outs on contraceptive methods across facilities, and continuous monitoring of stock-outs only happens in a few countries

Lack of stock optimization therefore, can mainly be attributed to improper analysis of information relating to stock levels. Furthermore, the challenges in analysis of stock related information is as a result of lack of supporting synergies that would support the analysis. These include lack of efficient sharing of information that supports optimization of stock levels such as current stock levels. Additionally, poor

inventory control practices would result in insufficient data required for optimization of stock. Further, lack of proper quantification of commodities results in missing information required for stock optimization. Additionally, absence of sufficient quality data on available stock and consumption statistics leads to low accuracy and reliability of quantification estimates for commodities. In addition, insufficient data management strategies related to consumption and stock levels would result in low quality data for analysis. Moreover, the effectivity of the planning and decision making in stock optimization is jeopardized due to inaccurate and untimely data of low quality. The major causes of chronic stockouts are as a result of high wastages in utilization of commodities, poor reporting rates, poor stock management supervision, monitoring and evaluation.

To overcome these limitations, there is need for excellent support from suppliers to retailers which can be achieved through proper communication channels. Additionally, the balance between satisfying the demand and the inventory safety stock can be maintained by determining the optimal policies of inventory. Further, these limitations can be mitigated by enhancing retailer-supplier integration systems such as through Vendor Managed Inventory Systems. In addition, adoption of technology that incorporates stock optimization models can be used to manage stock levels. Also, availability of data consumption and inventory control tools in order to help capture consumption data to aid in accurate demand forecasting and requisition process can help mitigate the aforementioned limitations. Consequently, there is need to develop commodity-based dashboards that enhance decision making with respect to the levels of stock and efficient redistribution of commodities across the facilities. Also, there is need to improve efficiency in collaboration, where firms adopt advanced information systems that guarantee high performance and quality of service

2.8 Stock

The determination of the quantity of goods or resources that inventory can hold is referred to as stock (Maina & Ngugi, 2019). Further, they assert that stock refers to the products, such as raw materials or completed goods, that an organization has for sale or is preparing to sell. A system of policies and controls known as an *inventory system,* that keeps track of inventory levels and decides how much should be ordered, when they should be replenished, and what levels should be maintained. (Nzioka & Were, 2017). According to Olang'o, (2018), looking ahead is one of the parts of *inventory planning* that determines how much and when to order to maintain overall stock levels. (demand forecasting), while the inventory control part adheres to the predetermined procedures from the planning stage in order to control stock by continuously or sporadically checking stock levels, then making decisions about how to proceed based on the information acquired. Additionally, he submits that inventory management's main objective is to reconcile competing economic interests in order to avoid overstocking or understocking, to guard against increasing inventory costs such as storage, theft, obsolescence, stock outs, spoiling, and lastly ensuring that products are available in the required quality, quantity, timing, and location. Shajema (2018) supports the argument by Chambers Lacey (2011) that the inventory

policy of many companies is to hold enough finished stock to satisfy market demand while reducing holding costs and enabling them to achieve their goals.

2.9 Stock Optimization

According to Shajema, (2018), overstocking necessitates a lot of space, suitable storage, and significant financial investment, yet the movement of the stocks depends on how quickly the products are sold. Additionally, when sales are low, stocks move slowly and businesses are very concerned about the risks of losing products with expiration dates or are perishable, which become obsolete. Additionally, stock losses from expiration and perishability might result in poor financial performance. He adds that carrying costs could be as much as 30% of the value of the goods. On the other hand, he contends that understocking stocks lowers the amount tied up while simultaneously raising the danger of running out of goods. Also, he contends that due to potential loss of sales and goodwill, it is expensive to run out of stock. He claims further that the management must therefore balance the two expensively incompatible issues of overstocking and under-stocking. He concludes by stating that consequently, a solid inventory management system should be any retail chain store's top concern if it wants to perform better. Notwithstanding, his argument isolates the supplier in the inventory management, and thus prompting the interrogation of sustainability of examining a retailer stock level optimization without including the role of the supplier.

The main goal of inventory management is to provide managers with information on how much of various commodities to order, when to reorder, how often to place orders, and how much safety stock is necessary to prevent frequent stock outs (Olang'o, 2018). Furthermore, they contend that efficient inventory management can significantly boost a company's profit and return on total assets. However, they assert that the fundamental problem is figuring out the inventory level that works best with the existing organization's operating system or systems. In addition, they claim that firms' inventory managers are under increased pressure to create solutions that would allow them to reduce inventory expenses and enhance inventory movement in the supply chain. Consequently, they would be able to satisfy the market demand in good time. According to them, their top priority is to keep the appropriate amount of goods on hand to meet demand, while avoiding excess inventory that would result in increased holding costs or stock out resulting to lost sales and customers.

In the manufacture or delivery of goods, the phrase "stock out" refers to an inventory shortage brought on by unforeseen demand, insufficient inventory control, production bottlenecks, or interrupted replenishment. (Olang'o, 2018). Further, they state that its effects are felt at all levels leading to lost sales and dissatisfied customers. As such, they claim that stock control is carried out in order to keep track of how much stock is available at any given time and to monitor it. According to them, effective stock control enables the appropriate amount of stock to be in the right location at the right time because it enables the stock user to monitor stock level, place orders, and distribute stock using either

straightforward manual procedures like stock books, stock cards or electronic systems like barcodes or business models like Just in Time (JIT). Additionally, they maintain that efficient stock control of commodities will ensure effective and efficient services rendered to customers and this can only be experienced when there are proper structures and systems in any organization or sector. They conclude that frequent stock out of any commodity indicates that there is an inefficient stock control which may be due to poor planning, poor forecasting, poor coordination, poor procurement procedures, inventory pilferages, inventory obsolesce and low funding.

According to the traditional stockpile management technique, the ideal stock level is one obtained from the lowest total expenditures, implying that stock level optimization is viewed from the perspective of expenses (Korponai et al., 2017). Furthermore, they claim that while the costs associated with procurement reduce with an increase in order quantity, holding costs rise linearly as lot size increases. Similarly, there is a trade-off between holding expenses and the costs associated with a stock shortage. As a result, suppliers strive to maximize the total cost function, which meets the goal of cost minimization, and to quantify the derivable order quantity and the order time.

2.10 Stock Optimization Indicators

The *service level* in inventory management is the projected likelihood of not hitting a stock-out during the subsequent replenishment cycle or the probability of not losing sales, which is established in a firm by the level of stock (Radasanu, 2016). In addition, he argues that the safety stock level should be high enough to cover vendor delivery periods, adequate to meet changing consumer demand, but not so high as to cause the business to lose money due to excessive carrying costs. However, he asserts that holding a high amount of stock is costly and comes with a number of hazards, including those related to storage, expiration, and price declines. Additionally, he asserts that the heuristic technique is typically utilized to maximize the service level particular to each product. In addition, he argues that the goal service level can be described as a compromise between the cost of stock-outs and the cost of inventory. He claims that the cost of working capital, storage space, routine manipulations, and damaged or lost goods can all be classified as the cost of inventory. Additionally, a number of criteria, including the price of lost sales, can be used to determine the cost of stock-outs.

According to (Radasanu, 2016), the four elements that impact on service level: delivery flexibility, readiness to deliver, reliability, delivery time and quality. Further, the formula used to gauge delivery readiness based on the quantity of units sold is as follows:

$$service\ level = \frac{the\ number\ of\ quantities\ delivered\ in\ time}{the\ total\ quantity\ of\ the\ demand} \ldots\ldots\ldots\ldots (1)$$

Table 2.1: Options for calculating the service level

Criterion	***Formula for service level***
Stock-out	The number of quantities delivered / the total quantity of the demand
Frequency of stock-outs	The number of order delivered / the total quantity of customer sales
Frequency of stock-outs	The number of order item delivered / the total quantity of order items
Loss of sales	The value of quantities delivered on time / the value of the total quantity of the demand
Stock-out period	The number of days with stock-out / the total number of days

Source: (Radasanu, 2016)

Radasanu, (2016) defines *safety stock* as the inventory kept on hand to avoid scenarios like stock outs and backorders. Safety stock, in his opinion, guards against a variety of deviations, including delivery date variances (when the lead time of replenishment varies), variance in requirement (when the forecast is off), variance in quantity delivered (when the vendor does not deliver sufficient products or the quality of delivered products is subpar), and variances in inventory (when inventory recognizes a deviation between the actual inventory and plan). These variations are listed in detail, along with their connections, in Figure 2.1. Additionally, he argues that the safety stock can be determined using either the historical distribution of demand or a projected distribution of demand (forecast error). Firstly, he submits that the standard method of computing safety stock follows the formula below:

$$safety\ stock = safety\ factor * average\ replenishment\ lead\ time \ldots\ldots\ldots\ldots (2)$$

The calculation makes the assumption that demand is spread normally across the period of replenishment. He also asserts that the safety factor identifies the number of standard deviations that make up a given service level. Additionally, he claims that the safety factor from the distribution function of the standard normal distribution can be used to determine whether the stock issues are distributed normally. Secondly, the safety stock is as follows based on the ultraconservative strategy.

$$safety\ stock = maximum\ daily\ consumption\ *\ maximum\ replenishment\ lead\ time\ in\ days \ldots (3)$$

However, he indicates that this approach, which is always based on maximum, results in surplus stockpiles and should only be utilized for important products or stock keeping units (SKU) whose consumption cannot be forecast. Third, the safety stock is computed using the percentage technique as follows:

$$Safety\ stock\ =\ average\ consumption\ *\ average\ replenishment\ lead\ time\ *safety\ factor \ldots (4)$$

Furthermore, he submits that the safety factor is between 20 and 40 percent and is decided without taking any particular factors into account. However, he notes that this strategy is dependent on the stock

controller's expertise and experience, as well as the company's acceptance of this aspect. However, he claims that in practice, the following guidelines are followed to establish the safety stock:

$$Safety\ stock\ =\ average\ period\ consumption\ *\ replenishment\ lead\ time\ in\ days\ \ldots\ldots(5)$$

or

$$Safety\ stock\ =\ 10-20\%\ of\ the\ average\ stock\ level\ \ldots\ldots\ldots(6)$$

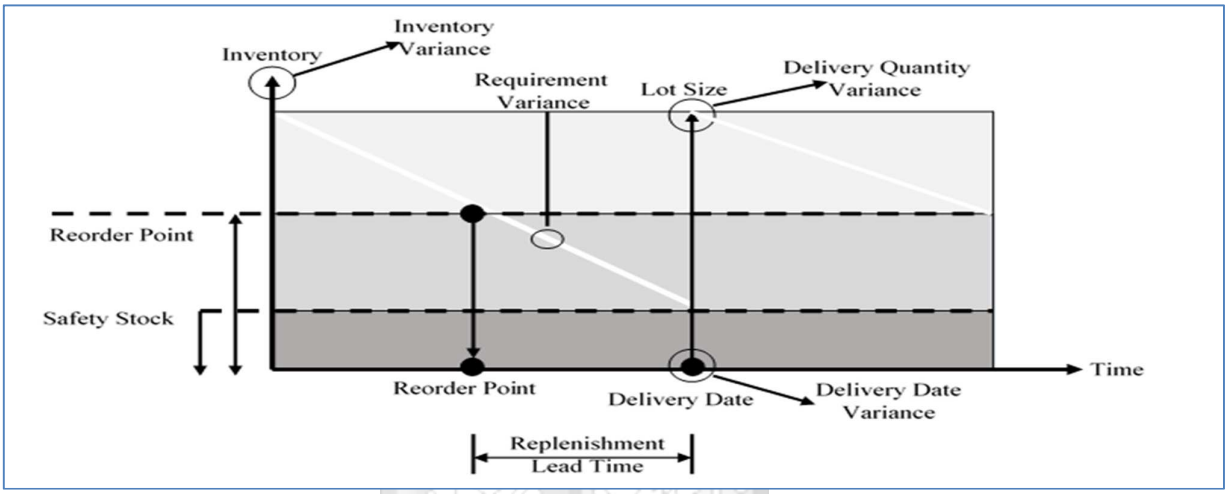


Figure 2.1: Reason for Safety Stock due to Deviations (Radasanu, 2016)

2.11 Stock Optimization Approaches

Hoswari et al., (2020) evaluated various methods of optimizing stock levels. These include Economic Order Quantity, Just-in-Time method and Activity Based Costing. Hoswari et al. (2020) asserts that the efficiency of each method is dependent on reliable data as the input.

2.11.1 Economic Order Quantity Approach

The primary concerns of stock management models are the maximum amount that can be purchased at one time at the most advantageous total expenses and the ideal timing of the purchase (Korponai et al., 2017). Additionally, they claim that the economic order quantity with planned shortage model can be used to determine the equilibrium between the stock level and costs. To achieve the equilibrium, there are initial conditions that have to be satisfied. First, the supply rate can be considered as being infinite. Secondly, the ordered quantity arrives as one item. Thirdly, the frequency of supplies is scheduled for identical periods. Further, the demand is known and pre-definable with absolute certainty. Also, both the customer and the supplier want to satisfy the demand. The demand is continuous and the utilization has a consistent intensity, thus the demand rate is constant. Accordingly, within a supply period, the stock level shows a strictly monotonous descending linear function in relation to time. Additionally, the stock shortage is accepted at a certain cost. Moreover, the ordering costs are independent from the order quantity. Also, the holding costs per unit are constant and they change linearly with the stock quantity.

In addition, the purchase price per unit does not depend on quantity, thus the purchase price does not influence the stock management policy to be chosen. Lastly, by assuming an infinite time horizon, the costs are independent of the time factor.

Consequently, they submit that the objective function defines the minimum of the function of total costs: According to them, the relationship between the change in stock level, purchase cost and holding cost as a function of order quantity is the foundation of the economic order quantity fundamental model. Accordingly, the buying costs per unit are more favourable when there are less frequent orders that are placed, whereas the holding costs rise linearly. They further assert that the total cost function can be defined as the sum of these three expenses plus the value of the acquired items. They contend that the objective function establishes the minimum of the total costs function as follows:

$$C(q;d) = Q * v + C_0 + C_h + C_s = \frac{Q}{q} * c_o + \frac{d^2}{2 * q} * T * c_h + \frac{(q-d)^2}{2 * q} * T * c_s \rightarrow min \dots\dots\dots (7)$$

where:

C -total cost of inventory management for the examined period
Q -total purchase demand for the examined period
v -purchasing price per unit
Co -total purchase cost for the examined period
C_h -total holding cost for the examined period
C_s -the shortage cost during the whole analyzed period
q -purchase demand for a single period, economic order quantity
c_o - cost of a single purchase order
d -the portion of the demand covered by stock within one single period
T -the length of the complete period
c_h -holding cost per time unit
c_s -shortage cost per time unit

Additionally, they submit that by solving the system of preceding equations (7) and setting the form of the partial derivatives corresponding to q and d of the function of total costs to zero, the optimal order quantity can be determined as follows:

$$q = d * \frac{c_h + c_s}{c_s} = \sqrt{\frac{2 * Q}{T} * \frac{c_o}{c_h}} * \sqrt{\frac{c_h + c_s}{c_s}} \dots\dots\dots\dots\dots (8)$$

Further, the lowest overall cost incurred for the entire period, including the cost of any purchased stocks is computed as follows:

$$C = \sqrt{2 * Q * T * c_o * c_h * \frac{c_s}{c_h + c_s} + Q * v} \ldots\ldots\ldots\ldots (9)$$

2.11.2 Just-in-Time Approach

John Krafcik introduced lean theory as an expansion of just-in-time principles in 1988. (Shajema, 2018). He adds that the theory reduces waste in the production process and does away with buffer stocks. Additionally, the theory goes into detail on how businesses can make orders with more freedom, hold less inventory on-site, and pay no carrying costs for inventory. Additionally, academic studies show that businesses can successfully optimize inventory through the use of lean supply chains strategies to achieve high levels of asset utilization and customer satisfaction, improving growth, profitability, and market share. Although the model points to maintaining minimum stock levels to reduce storage costs, the aspect of changes in demand is not included in the model. The model further alludes to frequent ordering strategy, which can imply increase in costs arising from the high frequency. Besides, eliminating buffer stock negatively affects the safety stock that guarantees sustainable supply in response to fluctuating demand in the market. It can be observed that the effectiveness of the concept relies on close, ongoing communication and information sharing between a retailer and a supplier.

2.11.3 Activity Based Costing

According to Arasa and Achuora, (2020), an organization's activities are identified using the activity-based costing (ABC) technique, which then allocates the cost of each activity to all products and services based on how much each activity consumes. Furthermore, they claim that when compared to traditional costing, this methodology assigns more indirect expenses (overhead) into direct costs. Additionally, it determines the cost groups of activity centres in organizations and assigns costs to goods and services based on a number of transactions or events that are crucial to the process of producing the good or providing the service. They also claim that the Consortium of Advanced Manufacturing-International used the idea extensively in its early stages, around 1987, but that retailing companies later applied the approach to stock management. According to them, retail businesses can use ABC to find operations that don't offer value and should be discontinued. They claim that by doing this, advancements can be made that could result in the retail industry achieving higher performance results. However, although the ABC method can lead to reduces costs based on the effect of the cost of each activity and contribute to suitable pricing of commodities, it fails to incorporate the influence of demand variability as a significant factor that determines the optimum stock level.

ABC analysis is predicated on the notion that a product's purported "importance" increases with its sales, for both the shopkeeper and his customers (Radasanu, 2016). Additionally, he claims that this presumption provides a practical manner to group products according to their respective sales volumes. Additionally, he asserts that each group is subsequently given its own service level in the manner shown below:

- items A, top 20% products, classified as “critical few”: high service level, e.g. 96-98%;

- items B, next 20-30% products classified as “interclass”: medium service level, e.g. 91- 95%;

- items C, last 50-60% products classified as “trivial many”: lower service level, e.g. 85- 90%;

Therefore, he avers that the ABC analysis is used to calculate the appropriate service level for groupings of products, but it is theoretically feasible to identify the best service level for each specific product.

2.11.4 Vendor Management Inventory System

Vendor Managed Inventory (VMI) is a streamlined method of managing inventory and ensuring customer satisfaction in which the supplier is entirely in charge of replenishing stock in light of useful Point of Service (POS) data provided from the retailers (Shajema, 2018). Furthermore, he argues that the strength of the relationship and trust between buyers and sellers, the structure of the information-communications technology, and the force of data exchange all have a favourable impact on the execution of VMI. Additionally, he asserts that the upstream information shared with suppliers, such as the current stock level, is the most important component for the efficient use of VMI. He also makes the point that, as part of the VMI contract, the supplier maintains the retailer's inventory and determines when and how much to restock. He adds that the VMI cooperation enables the suppliers to make crucial choices about the replenishment of retailers' inventory. Consequently, he claims that the vendor periodically decides on order quantities, shipment, and scheduling after physically or electronically tracking the buyer's inventory levels. He also asserts that the vendor's acceptance of the purchase order may be the first sign that a transaction is actually happening. He goes on to say that as a result of this arrangement, customers cede control over important resupply decisions and, in some situations, shift financial responsibility for the inventory to the supplier.

According to Shajema, (2018) research, the majority of Kenya's retail chain outlets have switched over to VMI systems. Through his research, he came to the conclusion that the technologies had improved relationships and integration between merchants and suppliers by enabling the use of order confirmation systems and information-sharing systems. However, according to Olow et al. (2020), he contends that before applying VMI, it is crucial to assess the degree of customer demand uncertainty since a high level of demand uncertainty has a detrimental effect on the performance gained by VMI. Furthermore, they contend that the most crucial element in the effective deployment of VMI is upstream data supplied to suppliers regarding current inventory levels and precise sales forecasts. They further claim that because suppliers stock more to reduce stock-out risks, retailers profit as well by reducing shortage and holding costs.

However, a VMI system in itself is not sufficient to optimize stock. In particular, it does not cater for the variation of demand. Therefore, there is need to incorporate a stock optimization model that allows

retailers or suppliers to maintain optimum stock levels by determining optimum safety stock that guards against demand variations. This model therefore, seeks to augment the VMI systems with a stock optimization model.

2.11.5 Significance of the Inventory Control to Retail Chain Stores Performance

The goal of the study by Ndwiga and Kiarie, (2017) was to forecast the performance of Nairobi's retail establishments in terms of ABC analysis, EOQ, Vendor Management Inventory and Just in Time. Their research showed that an increase in EOQ would result in a 0.769 increase in retail chain store performance, an increase in ABC Analysis would result in a 0.709 increase, an increase in Just in Time would result in a 0.678 increase, and an increase in Vendor Management Inventory would result in a 0.598 increase. Similarly, the study by Nzioka and Were, (2017) demonstrated that, when all other independent variables are held constant, a rise of unit in economic order quantity will result in an increase of 0.711 in educational performance. Second, when all other independent variables are held constant, a rise in just-in-time replenishment would result in a 0.338 improvement in educational performance. Thirdly, if all other independent factors were held constant, an increase in materials and requirement planning would result in a 0.293 increase in educational performance. Last but not least, if all other independent variables are maintained constant, an improvement in inventory control would result in an increase in educational performance of 0.616 units.

The study by Arasa and Achuora, (2020) showed that the most significant impact on performance is provided by the e-inventory management system, with a p-value less than 0.05 (p=0.018) and an explanatory power of 27.9%. They contend that implementing an electronic inventory management system into business operations improves performance by enhancing replenishment, accuracy in recording and issuance, stock traceability and monitoring. Additionally, according to their research, ABC is second in line with a less than 0.05 p-value (p=0.021) and a 26.0 percent explanatory power. They also claim that there is a link between ABC and correctly pricing products for customers, which fosters customer loyalty and increases performance through repeat business. Additionally, according to their findings, vendor-managed inventory had an explanatory power of 23.5% and had the third-highest influence on performance with a p-value of less than 0.05 (p=0.000). They assert that the removal of inventory holding costs via VMI has a positive impact on business performance. Finally, they claim that the lean inventory management system had no appreciable impact on performance, with a greater than 0.05 p-value (p=0.502) and a 3.6 percent explanatory power. They further contend that the results are inconsistent with those of previous research, which found that lean inventory management systems considerably increase worker productivity, decrease wasteful spending of time and resources, and increase cost efficiency. They relate this phenomenon to Kenya's subpar logistical infrastructure and insufficient supply chain systems when compared to the United States, where comparable studies were carried out. According to them, a functional logistics infrastructure and an effective supply chain system are necessary for the success of the lean inventory management system.

From the studies above, there is no uniformity in the findings, and therefore no conclusive decision can be arrived at towards the most efficient method in stock optimization. Furthermore, the studies conducted research based on the opinion of respondents on the effectivity of each method. A further research on the performance of the retail chains with respect to profitability, stock-outs and overstocking would have yielded a more conclusive comparison. However, it should be noted that the variability of the findings could be attributed to the different parameters included in each research, and the differences in the approach of conducting their research.

2.12 Stock Optimization Models

2.12.1 Statistical Model for Calculating Safety Stock

A statistical approach to determining and achieving targeted service levels, while maintaining low inventory levels to keep operational expenses down, is the most precise and efficient way to calculate safety stock (Radasanu, 2016). He also claims that each SKU needs to have its own safety stock calculation and re-order point calculation. He contends that in order to have safety stock adjusted when demand changes, the initial value of safety stock must be checked and calculated using an information system within predetermined time frames. Furthermore, he claims that statistical functions can be used since the safety stock is used to provide a certain level of protection brought on by the divergence in demand. Therefore, he argues that judgments for obtaining a particular service level can be generated using statistical theory. He also claims that standard deviation (equation 10) can be used to more effectively regulate service level and safety stock.

$$standard\ deviation\ (\sigma) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} \ldots\ldots\ldots\ldots (10)$$

$$where\ x_i = demand, \bar{x}\ is\ the\ average\ demand$$

Additionally, he claims that with a normal distribution, 68.26% of the data falls inside the range ($\bar{x} \pm \sigma$), 95.45% of the data falls within the range ($\bar{x} \pm 2\sigma$), and 99.73% of the data within the range ($\bar{x} \pm 3\sigma$). Additionally, he claims that the service factor is multiplied by the standard deviation in order to determine the precise amount needed to satisfy the required service level. He contends that the relationship between service factor and service level is non-linear, with higher service levels requiring larger safety stock levels as well as higher service factors. He asserts that the company can set varying service factors for groups of items based on profit margin, strategic relevance, or sales contribution, as opposed to adopting a fixed service factor for all products, so that the products with greater value for the business will have more safety stock. Additionally, the safety stock calculation can be performed using the equation below by assuming a stochastic or random consumption and a deterministic replenishment period:

$$safety\ stock = Z * \sqrt{\frac{LT}{T}} * \sigma_D \ldots\ldots\ldots\ldots (11)$$

$$where\ Z = sevice\ factor\ (Z - score);$$

$\sigma_D = standard\ deviation\ of\ demand$

$LT = total\ lead\ time$

$T = period\ within\ which\ the\ standard\ deviation\ is\ computed$

He further argues that the aforementioned equation operates flawlessly under the conditions of forecast periods, order cycle times, identical lead times, forecasts that are equal to the mean of the actual demand for the aforementioned periods, and forecasts that are the same for each period. However, he asserts that this scenario is extremely rare to occur, hence more variables must be included in the calculation to account for these differences. According to him, the safety stock equation changes when the key concern is lead time variability a follows:

$$safety\ stock = z * \sigma_D * D_{avg} \ldots\ldots\ldots\ldots (12)$$

$where\ D_{avg}\ is\ average\ demand$

Additionally, statistical calculations can be coupled to produce a lower total safety stock than the summation of the two individual calculations, when lead time variability and demand variability are both present as follows:

$$safety\ stock = z * \sqrt{\left(\frac{LT}{T} * \sigma_D^2\right) + \left(\sigma_{LT} * D_{avg}\right)^2} \ldots\ldots\ldots\ldots (13)$$

The safety stock is service factor times the square root of the sum of the individual variability's squares. The safety stock is calculated by multiplying the service factor by the sum of the squares of each individual variability (Equation (11) +(12)). However, he avers that Equation (13) cannot be applied when demand and lead time fluctuations are not independent of one another. He claims that in these circumstances, the safety stock is equal to the sum of the two separate computations. (sum of Equation 11 and 12):

$$safety\ stock = \left(Z * \sqrt{\frac{LT}{T}} * \sigma_D\right) + \left(z * \sigma_D * D_{avg}\right) \ldots\ldots\ldots\ldots (14)$$

2.12.2 Reorder Point Replenishment System

Reorder Point replenishment is one of the traditional replenishment methods (Krzyzaniak, 2017). Further, under this method, orders of a defined amount are placed whenever the effective stock hits a specific reorder level. Furthermore, he argues that despite the development of alternative goods flow management concepts, inventory management remains a crucial issue in terms of costs associated with maintenance and replenishment, as well as the level of service gauged by inventory availability levels, due to the unpredictable nature of demand. According to him, the ordering level B is specified as:

$$B = D * LT + SS \dots\dots\dots\dots (15)$$

$$where\ SS = safety\ stock\ (Z * \sigma_{LT})$$

Additionally, he submits that the stock replenishment procedure is given by:

i. Current status of effective stock Se is specified

$$S_e = S_w + S_o + S_{er} - S_b \dots\dots\dots\dots (16)$$

where:

- S_w -stock physically available in the warehouse
- S_o -orders placed, but not yet implemented
- S_{er} -stock en-route,
- S_b -stock already booked

ii. Comparison of calculated effective stock level Se with reorder level B

iii. Placement of an order with an adopted fixed quantity Q, if Se ≤ B

2.12.3 Safety Stock Model for Demand Driven Inventory Replenishment

The demand-driven materials requirement planning (DDMRP) inventory replenishment approach has a lower average inventory and, concurrently, a lower stock-out rate than existing systems (Lee & Rim, 2019). However, they contend that unlike any other safety stock method, the DDMRP replenishment uses a subjective criterion for safety stock. Additionally, they assert that the guideline enables the user to arbitrarily choose the parameter values within a specific range, endangering the consistency of the performance of the inventory. They therefore present a different safety stock formula for DDMRP replenishment that is mathematically defined to be consistent and performs better than the DDMRP recommendations and other safety stock formulas in terms of stock-out rate and average inventory.

According to them, a maximum amount of stock (M) in the DDMRP replenishment model is expressed as the sum of three components: the red zone (RZ), yellow zone (YZ) and green zone (GZ). YZ is equivalent to the demand during lead time (DDLT) in the current approaches, because it is defined as average daily demand times average lead time as in (17). In the presently used replenishment models,

RZ stands in for the safety stock. Additionally, when the inventory position in DDMRP replenishment drops below the top of yellow, which is the sum of DDLT (i.e., YZ) and RZ, stated as the sum of red zone base (RB) and red zone safety (RS), an order is placed. GZ, the order quantity, is calculated by multiplying YZ by the lead time factor in Table 2.2, as in (18). The maximum stock level (M) in the current replenishment models is represented by the top of green (TOG), which is the total of YZ, RZ, and GZ as in (22). Top of yellow (TOY), which denotes the reorder point in the current models, is the sum of YZ and RZ, as in (23). Additionally, they submit that in DDMRP, the replenishment order quantity (RQ) is determined by subtracting the available stock (AS) from the top of green (TOG) and adding the order spike quantity (k), as shown in (24), where the order spike is a qualifying quantity of known cumulative demand that occurs within a qualifying time window and jeopardizes the buffer's integrity.

Table 2.2: Guideline for Lead Time Factor (Lee & Rim, 2019)

Lead Time	FL(%)	Purchased Part (Days)
Long	20-40	26+
Medium	41-60	11-25
Short	61-100	1-10

Further, they submit that applying a safety stock theory that relies on an inaccurate demand distribution shape might have disastrous results; as a result, the right distribution form must be identified in order to calculate the safety factor. First, the formula still adheres to the DDMRP recommendations while avoiding inconsistency brought on by the arbitrary selection of values within a range. Second, the historical data must be used to determine the parameter values. Third, both lead time variability and demand variability are taken into account. Fourthly, there is no usage of the safety factor based on service level. Fifth, the demand-driven theory holds that shortages should be kept to a minimum and average inventories should not rise. They further show that:

$$YZ = d * L \dots\dots\dots\dots (17)$$

$$GZ = YZ * F_L \dots\dots\dots\dots (18)$$

$$RB = d(1.02\sqrt{L} + 1.15) \dots\dots\dots\dots (19)$$

Where:

d -average daily demand

L -average replenishment lead time

Assuming RS=σ_L, and the coefficient of variation, CV=$\frac{\sigma}{\mu}$,

$$RS = d\left(1.02\sqrt{L} + 1.15\right)\sqrt{CV_d^2 + CV_L^2 * L} \ldots\ldots\ldots\ldots (20)$$

$$RZ = RB + RS = \left[d\left(1.02\sqrt{L} + 1.15\right)\right]\left[1 + \sqrt{CV_d^2 + CV_L^2 * L}\right] \ldots\ldots\ldots\ldots (21)$$

$$TOG = GZ + YZ + RZ \ldots\ldots\ldots\ldots (22)$$

$$TOY = YZ + RZ \ldots\ldots\ldots\ldots (23)$$

$$RQ = TOG - AS + k \ldots\ldots\ldots\ldots (24)$$

$$where\ AS\ follows\ equation\ (16)$$

2.12.4 Optimization Model Based on Lagrange Multiplier for a Single Stock Item

The Harris-Wilson formula for determining the economic order quantity is a good illustration of how optimization is frequently carried out in the field of stock management in respect to cycle stock (Krzyzaniak, 2022). He claims that there are no restrictions placed on the optimal quantities for these models. However, he asserts that there are limitations, in practice, that should be considered (Figure 2.2). The goal of the study was to develop a model to identify the ideal stock structure for the various circumstances in which stock replenishment is carried out. The solution's constraint conditions were contained in the model, which was a part of a Lagrange function. The study's conclusion was a system of equations that can be used to calculate the optimal Lagrange multiplier value, which is used to determine the components of the inventory structure and other quantities (such as service level indicators and costs like stock replenishment, stock deficit costs and stock maintenance) that are related.

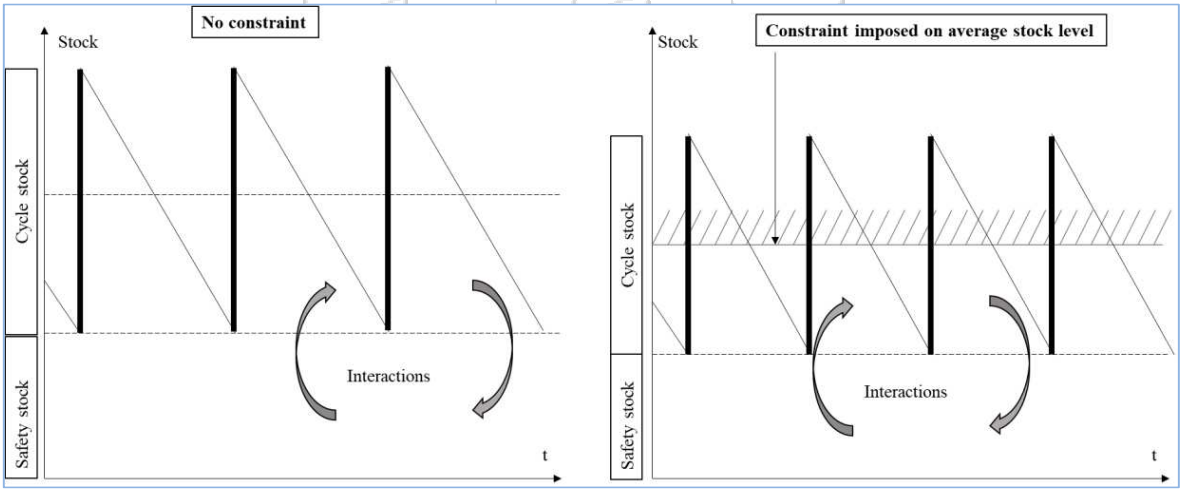


Figure 2.2: Optimization of cycle and safety stock in the absence and presence of constraints (Krzyzaniak, 2022)

According to Krzyzaniak (2022), there are interactions between the best cyclic stock size (best delivery quantity q) and the best safety stock level (best safety coefficient ω). He goes on to illustrate by stating

that reducing the cycle stock (by reducing the delivery amount) necessitates increasing the number of orders, which raises the anticipated cost of stock deficiency and necessitates increasing the safety stock. However, reducing the safety stock raises the possibility of a shortfall during the replenishment cycle, which necessitates a rise in the number of deliveries and, as a result, a rise in the cycle stock. The Reorder Point mechanism of stock replenishment with a set order (delivery) quantity was considered in the Total Stock Cost Model with Constraints. As a starting point, it uses a cost model that accounts for stock replenishment, stock carrying, and stock shortfall. The following labels are used by the model in relation to demand, costs, and service level:

D	-demand in a time unit (e.g., daily/weekly demand)
σ_D	-standard deviation of demand in an adopted time unit
σ_{DLT}	-standard deviation of demand in a stock replenishment cycle of mean Lead Time (LT)
D_a	-annual total demand,
cc_a	-annual stock-carrying cost coefficient
ω	-safety coefficient - directly influencing service level:
αSL	-service level (probability of a nonoccurrence of stock deficit in its replenishment period, probability to serve demand in a cycle), corresponding to safety coefficient ω treated as an independent variable
FR	-(fill rate) - as a percentage realization of demand in a quantitative approach
q	order/delivery - quantity independent variable
cr	-unit cost of stock replenishment (cost of order, organization, and execution of a single delivery)
cd_1	-cost related to stock deficit occurrence during the stock replenishment cycle
cd_2	-cost related to stock deficit occurrence in relation to one missing piece of the stock item
p_u	-purchase price (variable production cost) of a unit of the discussed stock item
nd_a	-number of orders (delivery) per year

The model is a Lagrange function indicated by L, obtained after addition of the constraint component and the Lagrange multiplier as follows:

$$L = \frac{D_a}{q} * C_r + \frac{1}{2} * q * p_u * cc_a + \omega * \sigma_{DLT} * p_u * cc_a + cd_1 * [1 - F(\omega)].\frac{D_a}{q} + Cd_2 + * I(\omega) * \sigma_{DLT}$$
$$* \frac{D_a}{q} + \lambda * (\frac{1}{2} * q * c + \omega * \sigma_{DLT} * c - C \ldots\ldots\ldots\ldots\ldots \quad (25)$$

Where

C	-constraint: maximum admissible average stock level
c	-the unit quantity of the assumed constraint:
c	= pu (unit price if the stock holding cost is a constraint: C = SHC)

c = v (volume of the stock unit, if the stock volume V is a constraint: C = V)

c = m (mass of the stock unit, if the stock mass M is a constraint: C = M)

c = 1 (when the constraint is the stock quantity in natural units)

The safety coefficient ω, which appears in the section of formula (15), deals with carrying safety stock, and depends on both the kind of demand distribution and the adopted service level, which is regarded as the likelihood of meeting the total demand in a replenishment cycle αSL. The standard deviation in the demand lead time follows equation below:

$$\sigma_{DLT} = \sqrt{(\sigma_D^2 * LT + \sigma_D^2 * D^2} \dots\dots\dots\dots\dots (26)$$

F(ω) – distribution function corresponding to the service level that has a relationship to the demand distribution seen throughout a stock replenishment cycle αSL; thus [1- F(ω)] is a likelihood that an item will go out of stock during a lead time for restocking. The following calculation is used to compute the volume of shortfalls expected during a cycle:

$$I(\omega) \cdot \sigma_{D,LT} \dots\dots\dots (27)$$

I(ω) – standardized number of deficits is calculated as follows:

$$I(\omega) = F(\omega) - \omega * [1 - F(\omega)] \dots\dots\dots\dots\dots (28)$$

where f(ω) is the density distribution function. Further, he submits that The first derivatives of the L-function with respect to both of the independent variables, q and ω, must be zeroed in order for the L-function to have a minimum. It can be demonstrated that (Krzyzaniak (2022): from the condition $\frac{\delta l}{\delta q} = 0$:

$$q = \sqrt{\frac{2 * D_a * \{c_r + cd_1 * [1 - F(\omega)] + cd_2 * I(\omega) * \sigma_{DTL}\}}{p_u * cc_a + \lambda * c}} \dots\dots\dots\dots (29)$$

Assuming that $\frac{\delta l}{\delta \omega} = 0$ and considering that $\frac{\delta F(\omega_i)}{\delta \omega_i} = f(\omega_i)$:

$$q = \frac{D_a * [cd_1 * f(\omega) + cd_2 * \sigma_{DTL}(1 - F(\omega))]}{\sigma_{DTL}(p_u * cc_a + \lambda * c)} \dots\dots\dots\dots (30)$$

Comparing the delivery quantity q from equations (29) and (30) and replacing $I(\omega)$with equation (28), we obtain the following equation:

$$\sqrt{\frac{2 * D_a * \{c_r + cd_1 * [1 - f(\omega)] + cd_2 * [F(\omega) - \omega * [1 - F(\omega)]] * \sigma_{DTL}\}}{p_u * cc_a + \lambda * c}}$$

$$= \frac{D_a[cd_1 * f(\omega) + cd_2\sigma_{DTL}(1 - F(\omega))]}{\sigma_{DTL} * (p_u * cc_a + \lambda * c)} \dots\dots\dots\dots\dots (31)$$

Additionally, he presents a condensed approach to get the Lagrange multiplier's ideal value λ_{opt} that will enable calculation of the ideal pair of independent variables $\{q; \omega\}_{opt}$ (Figure 2.3). Additionally, he claims that when there are restrictions placed on the quantities that occur in optimization models, using the Lagrange multiplier is a well-known method for figuring out the best values for independent variables. Additionally, he asserts that the use of this method in the field of inventory management typically involves groups of stock items, where, for instance, financial limits (total stock outlays) or spatial constraints (total stock volume) are included.

Additionally, he showed how the Lagrange multiplier could be used for a single stock item while assuming that the target function was the sum of the costs for replenishment, maintenance, and deficit. Additionally, he claims that the two types of deficit costs that were considered were those attributable to both the actual existence and quantitative shortages. Additionally, the order quantity and the so-called safety coefficient, which corresponds to service level, were the independently optimized variables (measured by two indicators). Furthermore, he contends that both of these values establish the cycle stock and the safety stock, the two primary elements of the stock structure. Additionally, he claims that when restrictions are placed on the stock (related to quantity, value, or space), meaning that the average total stock cannot exceed a certain level of restriction, we are able to determine the best stock structure possible for the circumstances at hand, i.e. the proportion as a percentage of each stock component in the total stock.

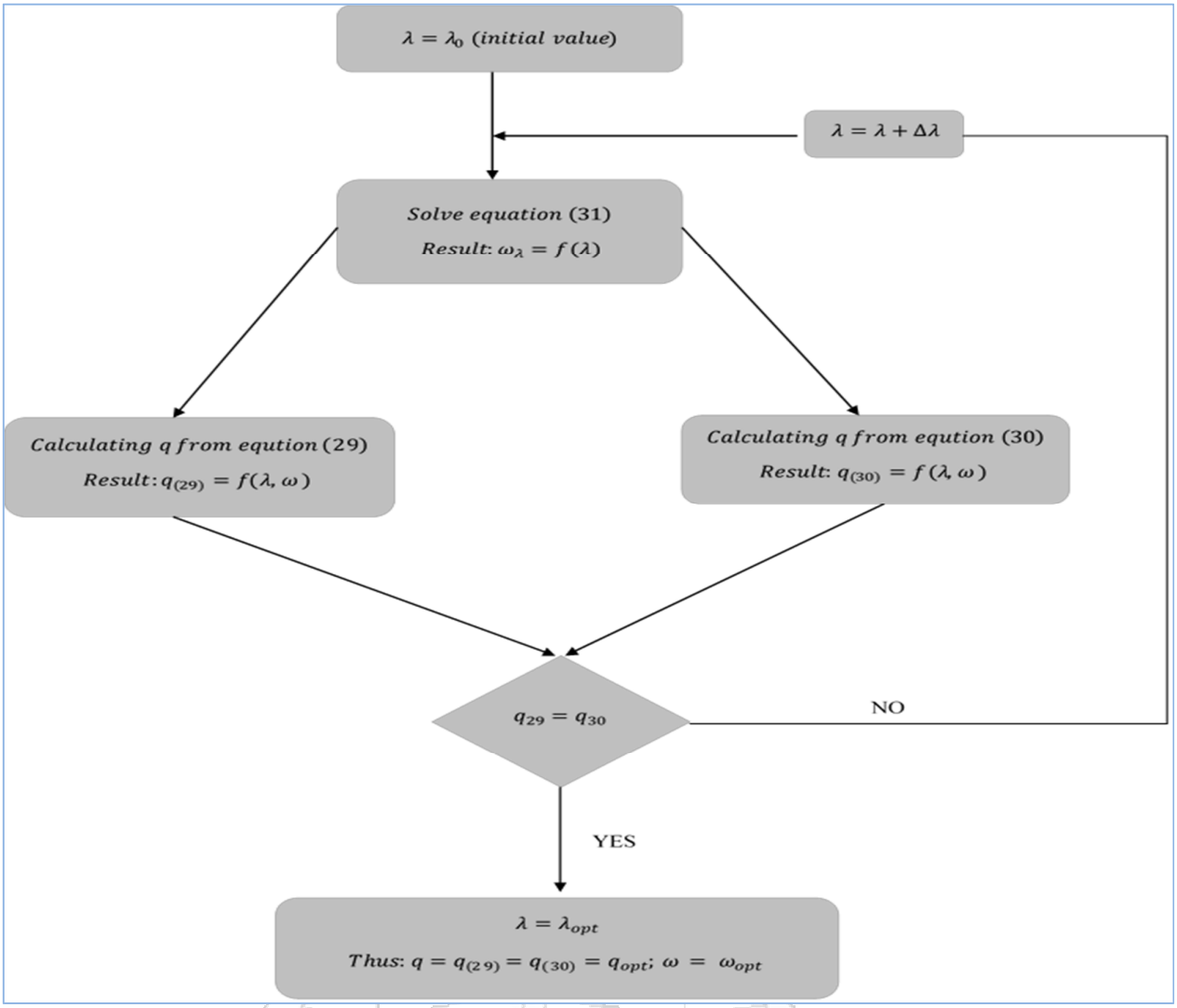


Figure 2.3: The algorithm for determining the optimum delivery quantity (qopt) and safety coefficient (ωopt) (Krzyzaniak, 2022)

2.12.5 Summary of the Optimisation Models

Name of Model	Key Author	Merits	Limitations
Economic Order Quantity Theory	Korponai et al., (2017) Shajema, (2018)	➢ Minimizing total inventory holding costs and ordering costs ➢ Computing the optimum size of delivery	➢ Encourages large order stocks to reduce ordering frequency ➢ Downplays the role of buffer stock
Just-in-Time Lean Theory	Shajema, (2018)	➢ Minimizes wastage of resources ➢ Flexibility in ordering decisions ➢ Low storage costs	➢ Undermines influence of demand ➢ Promotes frequent ordering strategy ➢ Eliminates buffer stock
Activity-Based Costing (ABC)	Radasanu, (2016) Arasa and Achuora, (2020)	➢ Promotes identification of profitable commodities ➢ and elimination of low revenue commodities ➢ Contribute to suitable pricing of commodities	➢ Undermines the influence of demand variability ➢ Limits determination of optimum stock level for a single product
Safety Stock Model for Demand Driven Inventory Replenishment	Lee and Rim, (2019)	➢ Minimizes shortages while ensuring average inventory does not increase ➢ Does not rely on service levels	➢ Limited only to safety stock computation ➢ Modelled in Excel
Optimized Order Quantity Based on Lagrange Multiplier	Krzyzaniak, (2022)	➢ Imposed constraints on the optimized order quantity ➢ Computes optimum order quantity for a single stock item	➢ Limited only to order quantity ➢ Modelled in Excel
Envisaged Solution: To Mitigate the limitations of EOQ, ABC and Just-in-Time models and approaches by incorporation the two models of computing optimum safety stock and optimum order quantity for a single stock item. Translate the excel-based models to a computer program			

Table 2.3:Optimization Model Summary

2.13 System Architecture

The software architecture of computing system is the collection of structures required to reason about the system, which consists of software, pieces, relations between them, and qualities of both (Richardson, 2019). Further, he establishes four distinct perspectives on software architecture. (Figure 2.4). The Logical view: The relationships between the program components that developers construct, such as classes and packages. The Implementation View: Modules are produced as the build system's output. Process view: The elements at runtime, each of which is a process, and their communication between processes. Deployment: How the processes are translated into machine code, including relationships (networking), virtual machines or physical, and the processes themselves. In addition to these models, there are the scenarios: - the +1, in the 4+1 model which animate the views by describing how the different architectural elements inside a given view work together to handle a request. Additionally, he claims that architecture satisfies the second group of application requirements: its quality of service, sometimes referred to as quality attributes, which describe the runtime features such as maintainability, reliability, scalability, deployability and testability.

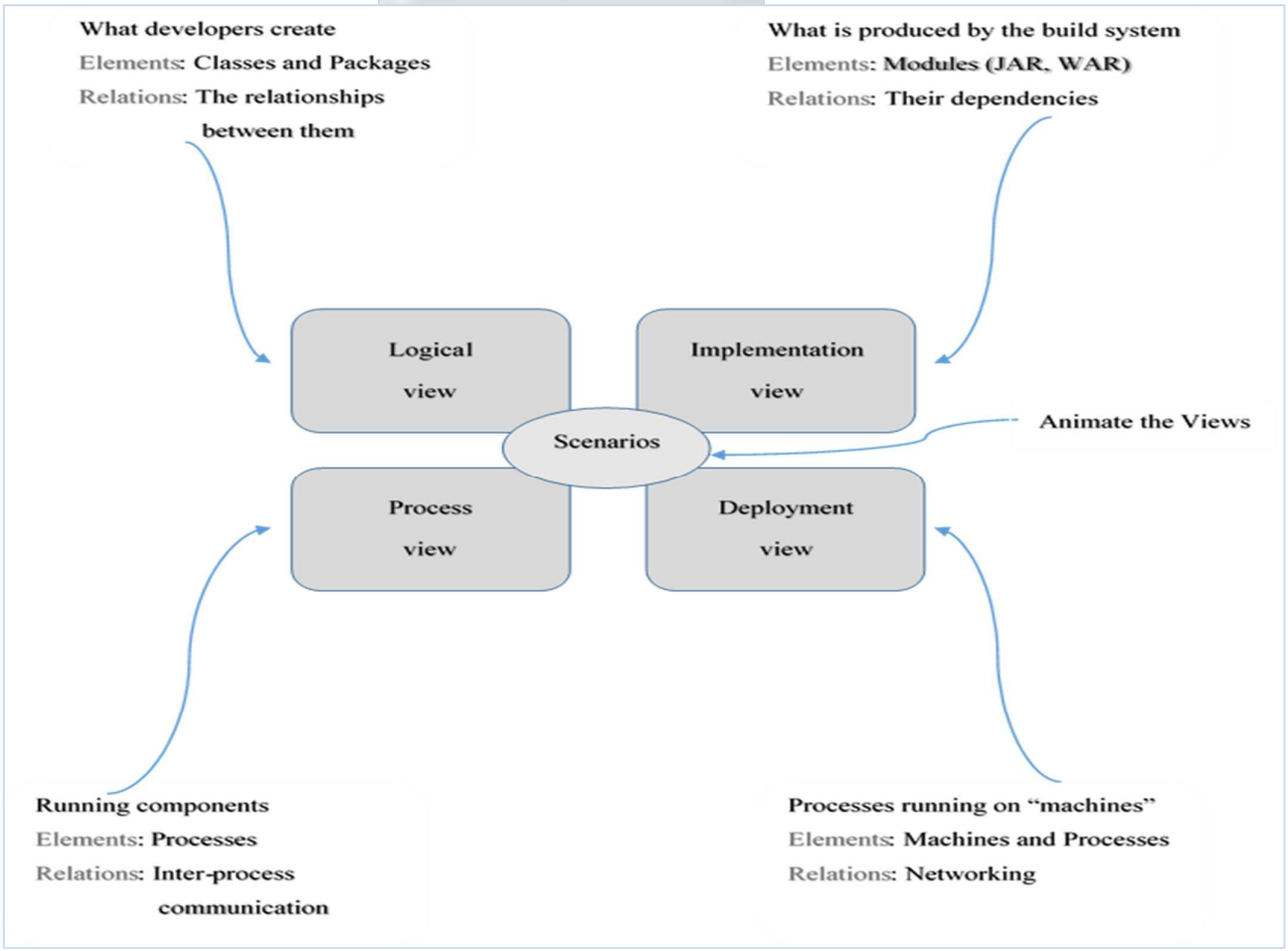


Figure 2.4 : The 4+1 view model (Richardson, 2019)

System architectures adhere to a specific architectural design (Richardson, 2019). According to him, he adopts the definition of architectural style as a family of such systems in terms of a pattern of structural organizes, which establishes the lexicon of parts and connectors that can be utilized in instances of that style, as well as a set of limitations on how they can be combined. The *layered architecture* is a well-known example. It divides software components into layers, each with a clear set of responsibilities, and it places restrictions on how much each layer can depend on the layers above it or the layers below it (strict layering). Further, he submits that the presentation layer, which implements the user interface or external APIs, the business logic layer, and the persistence layer, which implements the logic of interfacing with the database, make up the popular three-tier architecture. However, he claims that the layered architecture has significant flaws despite being an effective architectural design: *Single presentation layer*- It doesn't reflect the likelihood that an application will be used by more than one system; *Single persistence layer*- It does not reflect the likelihood that an application will interact with more than one database; *defines the business logic layer as dependent on the persistence layer.* This dependence, in theory, makes it impossible to test the business logic in a database-free environment.

Consequently, he asserts that the interdependencies in a well-designed application are misrepresented by a layered architecture. He claims that the business logic establishes an interface (or a collection of interfaces) that establishes the means of accessing data. The DAO classes that implement the repository interfaces are defined in the persistence tier. In other words, he argues that dependencies are the opposite of what a layered design shows. He offers a different architecture, the hexagonal architecture, to get around these problems.

Hexagonal architecture places the business logic at the heart of the logical view (Richardson, 2019). Additionally, the application features one or more inbound adapters that handle requests from the outside by invoking the business logic instead of the display layer. Similar to this, the application contains one or more outbound adapters that are called by the business logic and launch external apps in place of a data persistence tier. Further, he claims that the inbound and outbound adapters surround one or more ports (sets of activities) in the business logic (Figure 2.5). He further submits that the decoupling of the business logic from the presentation and data access logic in the adapters provided by the hexagonal architectural design is a significant advantage. This makes testing the business logic separately simpler. Additionally, it more closely mimics the architecture of a contemporary application. Each service's architecture in a microservice architecture is described by the hexagonal architecture.

Monolithic architecture is a design approach that organizes the implementation perspective as a single component, such as a single executable or Web Application Archive (WAR) file (Richardson, 2019). The *microservice architecture*, on the other hand, is defined by him as an architectural approach that divides the implementation perspective into a number of separate components, such as executables or

WAR files. Furthermore, each service has its own logical view architecture, which is often a hexagonal architecture, and the connectors are the communication protocols that allow those services to cooperate. He claims that the microservice design upholds the application's modularity as a result. He claims that the loose coupling of the services, which limits how the services collaborate, is a major restriction imposed by the microservice architecture.

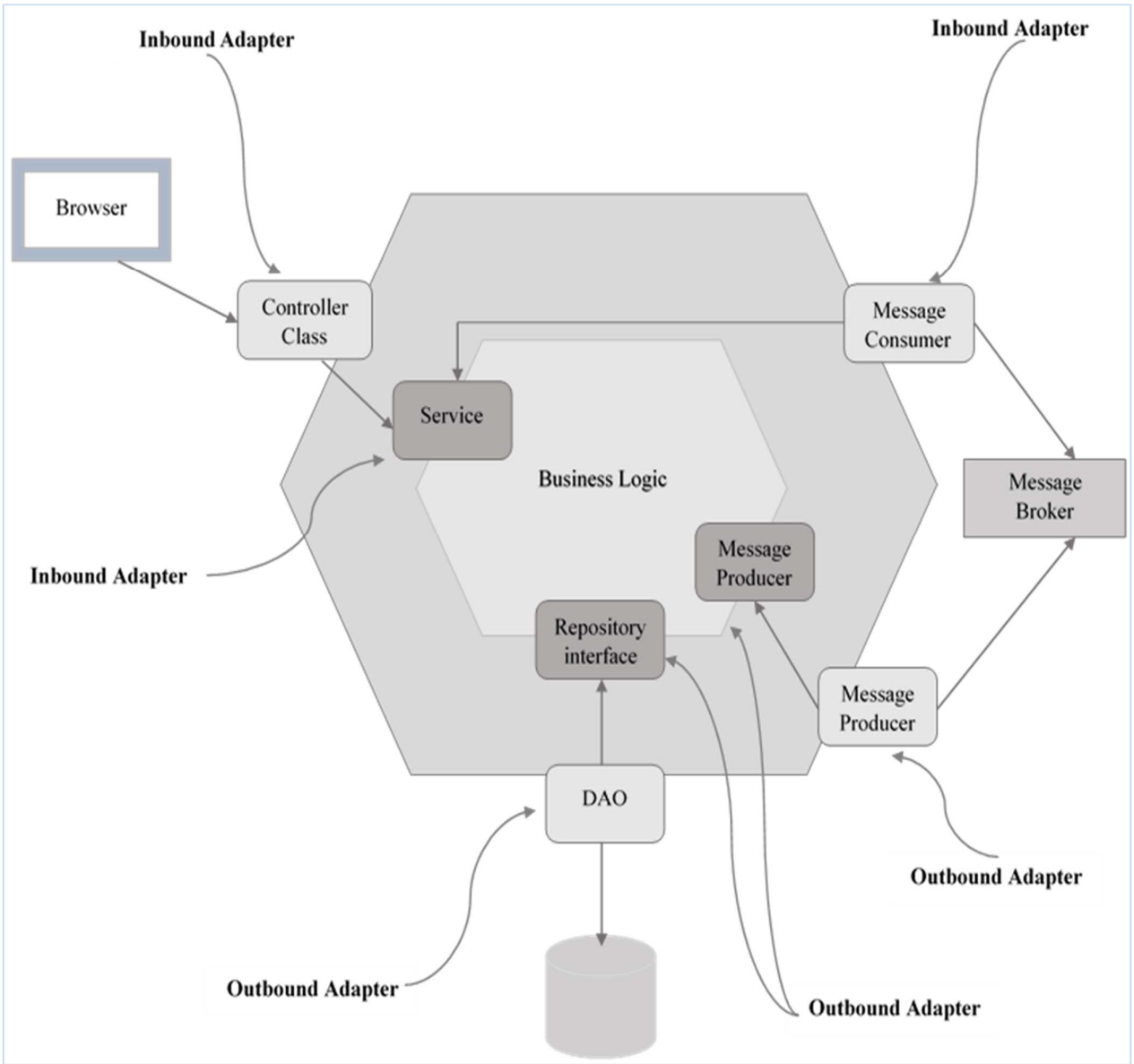


Figure 2.5: Hexagonal Architecture (Richardson, 2019)

Data Architecture. Big Data systems (BDS) are a type of computer programs that receive, store, process, and deliver enormous amounts of diverse data from numerous sources (Sergi et al., 2017). They further claim that traditional BI systems, which rely on the Data Warehouse (DW) as their de facto architectural standard, cannot be applied to Big Data environments. Furthermore, they claim that the three "V's"

(Volume, Velocity, and Variety) are often used to describe Big Data and refer to how difficult it is for relational database-based DW structures to handle and adapt to such massive, swiftly arriving, and varied amounts of data. They assert that Big Data architectures rely on NoSQL, co-relational database systems, whose fundamental data structure is not the relation, as its fundamental building blocks, to get beyond these constraints. These methods offer new approaches to the three V's by: Data and processing distribution in a cluster (typically of commodity machines); Introducing new data modelling techniques.

Further, they submit that the majority of NoSQL systems spread data (i.e., fragment and replicate it) in order to parallelize processing while taking use of the data locality principle, producing, in theory, a nearly linear scale-up and speedup. Additionally, in order to handle large-scale distributed processing, distributed NOSQL systems must loosen the well-known ACID set of attributes and the conventional concept of transaction, as stated by the CAP theorem. In addition, Sergi et al., (2017) claim that the five "V's" of volume, velocity, variety, variability, and veracity encompass the true essence of big data.

According to Ovidiu-Cristian et al., (2018), the typical streaming architecture is composed of three layers: the ingestion layer, the processing layer, and the storage layer, which is used to archive streams or store aggregated data. At each tier, a variety of systems are accessible. Several ingestion solutions use Kafka and Rest APIs. Apache Samza, Apache Spark, Apache Storm, and Apache Samza are a few of the streaming systems. Storage systems like Cassandra, HDFS, Druid, reddis, and Apache HBASE are some of the available options.

Deployment Architecture. Architectural patterns like publish/subscribe systems have become more popular from the perspective of information management, enabling businesses to create so-called data backbones that gather data in a single, distributed messaging system like Apache Kafka (Philipp Z. & Dominik R., 2017). They further claim that contemporary distributed streaming engines, such Apache Flink, can process both current and historical data simultaneously in a unified streaming architecture. Additionally, these architectures, also referred to as the Kappa architecture, lessen the effort required to deploy and maintain two different code bases for batch processing of historical data and stream processing for quickly computing real-time views required by other Big Data architectures like the Lambda architecture (Philipp Z. & Dominik R., 2017). According to them, therefore, whereas lambda architecture is organized for handling large data processing in an ordered manner based on various codebases, kappa architecture is structured for streaming event-driven big data processing based on a single codebase. BDSs are naturally distributed systems, therefore their designers must handle issues like inconsistent communication latencies, partial failures, replication, and resource efficiency (e.g., via elasticity and data compression) (Davaoudian & Liu, 2020).

According to Davaoudian & Liu, (2020), the aforementioned architectural requirements (the five Vs) listed above demonstrate how standard Business Intelligence (BI) architectures, which rely on relational databases, cannot be utilized in the context of BDSs. Furthermore, they claim that BDSs are currently

being constructed utilizing complex and ad hoc architectural solutions. They contend that, in order to choose and coordinate particular software components from among many that are available and overlap, BDS architects must possess a very high level of competence. Additionally, to make the software architect aware of the sort of components and their related linkages in advance, they claim that they are motivated to utilize some Software Reference Structures (SRAs) that assist the building of actual architectures. Hence, they submit that s/he is largely in charge of choosing the appropriate technology for those components in accordance with the needs and objectives of the firm. They claim that there are six pertinent SRAs in the context of BDS, which are examined here along with their advantages and disadvantages.

2.13.1 Data Ingestion

Data ingestion is the process of extracting raw data from a variety of data sources, such as web servers and database management systems (DBMSs), whether it is static (which do not move and need regular synchronization) or streaming (Davaoudian & Liu, 2020). Furthermore, they claim that the batch extraction functionality obtains static data (such as relational tables or Hadoop files) through some connectors, including drivers for particular protocols (such as Apache Sqoop), APIs provided by the source application, and web crawlers. Examples of these connectors include massive data transfer protocols (such as Hadoop's copyFromLocal or File Transfer Protocol (FTP)). Additionally, they contend that the batch temp data store may be used to temporarily store the static data that has been gathered. Furthermore, they mention that the stream extraction feature often obtains streaming data (such as log data) by subscribing to a streaming API (e.g., tweets in JSON format that are collected from Twitter Streaming API). Additionally, they submit that the stream data captured may be saved momentarily in a stream temp data storage like Apache Flume, Apache Kafka, or Amazon Kinesis.

The raw data streams that are pushed from the data sources are queued up in the stream ingestion component (Sergei, 2017). They also claim that since several sources can all simultaneously push data streams (such as sensor or social network data), therefore, such a component must be able to handle high throughput rates and grow with the number of sources. Additionally, they claim that one of the main duties is to facilitate the assimilation of all incoming data (i.e., adopt a No Event Loss policy). To this end, they indicate that Streams are temporarily held in a distributed memory or disk-based storage buffer called an event queue. Furthermore, they assert that while this component does not need to be aware of the data or structure of incoming data streams, it does need to be aware of the source and kind of each event. They claim that methods like write-ahead logging or the two-phase commit protocol are utilized in such a distributed environment to provide fault-tolerance and durability of findings, nevertheless, that has a demonstrable influence on the availability of data to subsequent components.

2.13.2 Data Streaming

There has been an increase in the demand for real-time stream processing, and Apache Kafka has taken over as the default streaming data platform in many businesses (Jafarpour et al., 2019). Furthermore, they claim that Apache Kafka is a massively distributed publish/subscribe messaging system where data is generated to and consumed from topics. They also claim that it offers a scalable and trustworthy platform to gather and store all the created data from various systems and to effectively distribute the acquired data to all the systems that want to use it. Additionally, they assert that Kafka's messages contain a key and a value, as shown in (Figure 2.6).

Moreover, they assert that each topic has a number of partitions to which messages are assigned according to their key. In addition, they contend that each division consists of an ongoing addition of records to a structured commit log in an ordered, immutable sequence. They claim that in order to provide fault tolerance, partitions are replicated across a configurable number of servers, referred to as brokers, in a Kafka cluster, with one broker serving as the leader for each partition and zero to more brokers serving as followers. Additionally, they contend that producers should allocate messages to the appropriate partition in the topic depending on the message key in order to publish data to the topics they choose. Additionally, they submit that Consumers are organized into consumer groups in order to consume the data that has been published on a topic, with each published message being sent to a single instance inside the consumer group. Figure 2.7 depicts two consumer groups that receive messages from a topic with four partition in a Kafka cluster with two brokers.

Furthermore, they assert that a consumer group can grow by adding new members, but it can also contract by losing members voluntarily or inadvertently. Additionally, a typical Kafka streams application will also read data from one or more Kafka topics, process the information, and then write the results into one or more Kafka topics. Additionally, they argue that a Kafka streams app's processing logic is defined as a processing topology with source, stream processor, and sink nodes, where the processing model is one record at a time. As a result, they claim, an input record from the source is processed by passing through the entire topology before the next record is processed. Further, they submit that stateful stream processing is provided by Kafka streams through so-called state stores, which are used to distribute the state of operations like join and aggregate across all instances of the streaming application.

Microservice. Microservices leverage stream processing libraries so that a single, lightweight microservice may handle each processing situation (Davoudian & Liu, 2020). Furthermore, they claim that rather of using a bulky full-fledged architecture of stream processing, a flexible lightweight microservice-based approach can be implemented. As a result, the speed layer in the Big Data architecture implementation uses the micro-services. Then, after processing the records it had received

in order, each service would produce fresh records to send asynchronously to other services or applications.

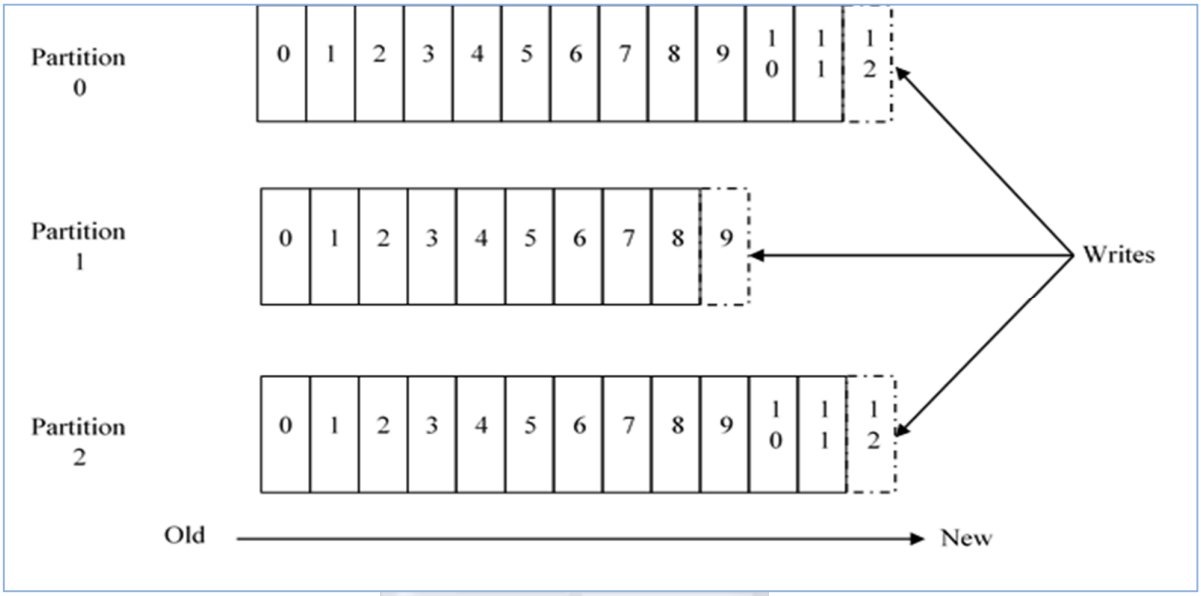


Figure 2.6:Anatomy of a Kafka Topic (Jafarpour et al., 2019)

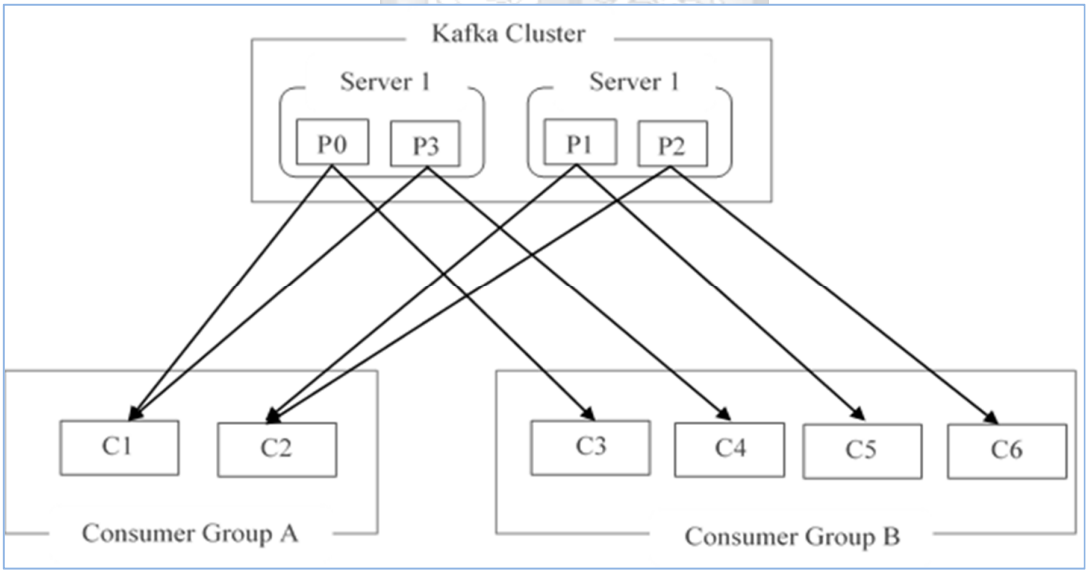


Figure 2.7: Two consumer groups Reading from a Topic with four Partitions

2.13.3 A summary of the Evaluation of the Architectures

The analysis is conducted based on whether the architectural requirements are satisfied (the five V's in Table 2.4). Table 2.4 highlights the evaluation's findings, separating SRAs and conventional architectures (Figure 2.8). In addition to the summarized analysis, other factors are highlighted as follows.

Requirements on Volume. Davaoudian & Liu, (2020) assert that the data interchange between a message broker like Apache Kafka and the streaming system may also be a bottleneck for the Kappa and Liquid designs. Further, due to triple-stores' processing capabilities, Solid does not effectuate the volume requirement (although there are efforts being made to improve these capabilities; the W3C standards do not include any mature, scalable solutions). Additionally, R1.3 is only partially satisfied by Kappa, Liquid, and Polystore because they only stream-process a portion of the total data volume.

Requirements on Variety. Davaoudian & Liu, (2020) submit that Solid does not meet this requirement as it only ingests RDF data. According to them, however, conflicts over semantic data interoperability are not resolved by any of the architectures (R3.3). Additionally, they assert that a BDS should completely utilize metadata in all data management activities, including information extraction, data cleaning, and data integration, in order to meet the Big Data integration standards (R3.2 and R3.3) and deliver an effective data analysis. However, the architectures have not fully implemented metadata management.

Requirements on Variability. Davaoudian & Liu, (2020) submit that the only architecture that satisfies the requirements for Variability is Bolster. Accordingly, it maintains in MDM the information on the input data sources (R4.3), descriptive statistics to access data evolution (R4.2), and the schema information of ingested items (R4.1).

As Table 2.4 shows, comparatively, Volume, Velocity, and somewhat Variety are more fully realized (Davaoudian & Liu, 2020). Furthermore, this is explained by the fact that these are the core capabilities of Big Data technologies, such as batch and stream processing engines and NoSQL storage. However, utilizing data semantics in data management operations is necessary to meet the other objectives. They contend, however, that built-in metadata management and repositories are features of conventional relational databases and data warehouses. For the current environment of Big Data technologies, a metadata standard has not yet been created. In conclusion, BDS architectures currently need a thorough, reliable method of managing metadata.

Retailers can stream their data to an online portal from where they can derive meaningful insights from the historical data. Using one of the architectures that meet the Volume, Velocity and Variety requirements for big data, the data can be ingested into a message broker, from where it is processed and streamed to a serving layer with storage capabilities to allow users to analyse historical data related to stock.

Table 2.4: Main Requirements for a BDS Architecture (Davaoudian & Liu, 2020)

Requirments	
R1	**Volume**
R1.1	-A scalable storage and processing of massive datasets is provided
R1.2	- Descriptive analysis is provided
R1.3	- Predictive/prescriptive analysis is provided
R2	**Velocity**
R2.1	- Streaming data are extracted
R2.2	- Streaming data are processed in a (near) real-time manner
R3	**Variety**
R3.1	- Heterogeneous data are ingested
R3.2	- A machine-readable schema of the entire data is provided
R3.3	- Semantic data interoperability conflicts are resolved
R4	**Variability**
R4.1	- Adaptation mechanisms for schema evolution are provided
R4.2	- Adaptation mechanisms for data evolution are provided
R4.3	- Adaptation mechanisms for the automatic inclusion of new data sources are provided
R5	**Veracity**
R5.1	- Mechanisms for data provenance are provided
R5.2	- Mechanisms for the assessment of data quality are provided
R5.3	- Mechanisms for tracing data liveliness are provided
R5.4	- Mechanisms for data cleaning are provided

Table 2.5: Fulifillment of the Requirements in BDS Architectures (Davaoudian & Liu, 2020)

		1.Volume			2.Velocity		3. Variety			4. Variability			5. Veracity			
SRAs	**DL Based**	R1.1	R1.2	R1.3	R2.1	R2.2	R3.1	R3.2	R3.3	R4.1	R4.2	R4.3	R5.1	R5.2	R5.3	R5.4
Lambda	Yes	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗
Kappa	No	✗	✓	✤	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗
Liquid	No	✗	✓	✤	✓	✓	✓	✗	✗	✗	✗	✗	✓	✗	✗	✗
Bolster	Yes	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓	✗	✗	✗	✗
Solid	No	✗	✓	✗	✓	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗
Polystore	No	✓	✓	✤	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗

Full support (✓), Limited support (✤), No support (✗), Data Lake (DL)

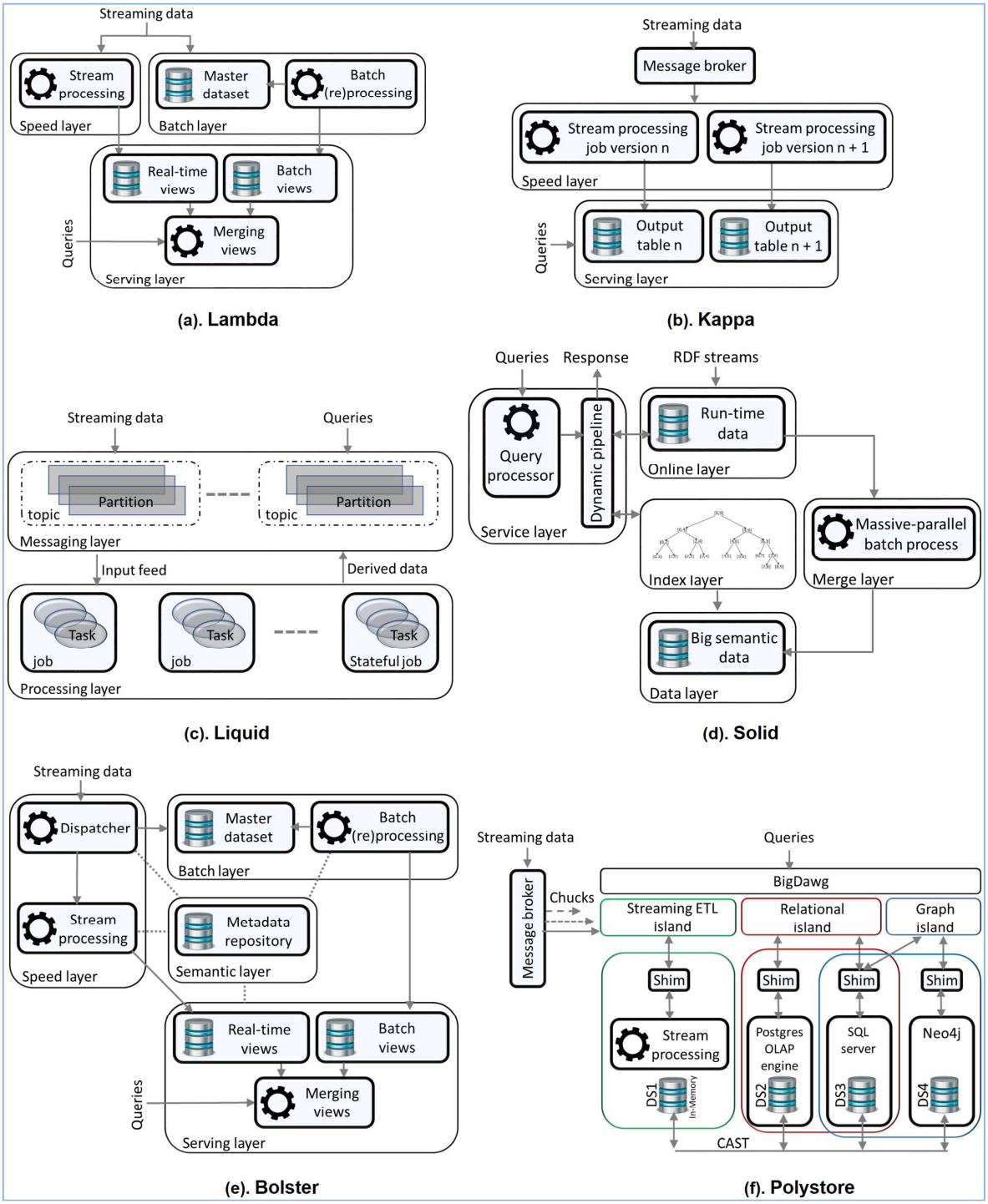


Figure 2.8: A Comparison of Various BDS Architectures (Davaoudian & Liu, 2020)

2.14 Related Work and Current Systems

Based on the discussion in section 2.16.5, ABC, VMI, EOQ and Just-in-Time are the common approach in managing stock inventory in Kenya. Additionally, the systems are mostly localized either in the retail store or synchronized within the retail chain stores. However, based on the discussion in section 2.8,

significant number of retailers or institutions have not adopted any efficient inventory management system. Retail stores such as Walmart and Procter and Gamble (Weibhuhn & Hoberg, 2021) use VMI systems to manage stores across their chain stores. However, as earlier mentioned, this approach transfers the inventory management to the supplier. Additionally, the retailer has no control over the stock levels at his/her premises, and relies on the supplier to manage his/her stock. According to them, VMI is focused on streamlining inventory management throughout the supply chain. They further claim that continuous review (r, q) type rules are common inventory policies to accomplish this. In a variety of contexts, optimal reorder point policies have been discovered under largely game theoretic model formulations. They also look at a scenario where inventory control is shared, and for various power constellations, they determine the manufacturer's and retailer's optimal choices for q and r, respectively. The parameters of a model negotiation over r and q between the parties, however, have inventory policy and contract terms that function well and are specific to inventory ownership at the retail site. In contrast to the studies mentioned above, where each side was involved in determining r and q, the vendor chooses the reorder points for his retail customers on his own, with the assumption that q is predetermined. Therefore, VMI results in the best supply chain performance, provided that businesses agree to share any cost savings via set transfer payments among themselves.

The reorder point policies can be changed to accommodate things like purchasers receiving periodic deliveries and dynamic reorder points (Weibhuhn & Hoberg, 2021). They further claim that the vendor follows a stochastic demand process continually but only occasionally completes his unfulfilled client orders (this is referred to as a "time-based, shipment-consolidation policy"). Additionally, there is a basic trade-off between minimizing consumer delivery wait times and achieving transportation economies of scale. They assert that in accordance with the relevant quantity-based policy, the vendor delays shipping replenishments to merchants until total cumulative demands equal a full truckload. Furthermore, they contend that a reorder point that dynamically alters depending on the manufacturer's order situation and the vendor-managed raw material inventory level at his subcontractor is a crucial component of the ideal (r, q) policy. Notably, this variation does not guarantee optimized stock levels at the retailer premises. The aforementioned trade-off increases uncertainty for the retailer, whose main concern is satisfying the needs of the consumer in a timely fashion.

2.14.1 Some of the Existing Enterprise Resource Planning Systems

MRPEasy (https://www.mrpeasy.com/inventory-optimization-software/): is a system that supports managing the manufacturing and distribution processes for small manufacturers, ideal for companies with 10 - 200 employees. It allows visualizing goods available in stock, how many are already booked, and if replenishments are expected to arrive. Further, it helps avoid stock-outs and shortages by setting reorder points for stock items, sends notifications whenever inventory is low and maintains inventory at a desired level while releasing capital tied up in excess stock. The observed limitations are: limitation

to number of users; only modelled for manufacturers; and manages stock-outs on the basis of pre-set reorder points for stock items.

SYSPRO (Syspro WhitePaper): is a set of fully integrated modules that acts as an Inventory Optimization (IO) Suite and gives users the means to optimize their inventory. The IO Suite can also track and manage the factors that influence inventory and its causes. It can assess the effects of changing these factors on stockholding, service levels, manufacturing, delivery performance or procurement performance. SYSPRO optimizes stock through four steps: Considers the relevance and behaviour of stock codes before classifying them into similar groups according to the ABC method; Creates a forecast for each code of stock using the most accurate demand estimate feasible; Determines the ideal balance between customer service and inventory investment to satisfy the anticipated demand by modelling a set of stock policies; Replenishes stock in a timely fashion accordance with prediction and stock policy.

Accordingly, SYSPRO Inventory Optimization produces maximum and minimum levels dynamically that can be used in Material Requirements Planning to come up with more pragmatic stock holding recommendations. Although the system incorporates demand and the ABC approach to optimize stock, the limitations of the ABC method still influence the efficiency of the system model. ABC analysis is used to determine a sufficient service level for groupings of items, but in theory, an optimal service level for each individual product is attainable.

Logility (Logility Handbook, 2022): Multi-echelon Inventory Optimization (MEIO) optimizes stock location and amount across all sites and nodes in a supply chain network. The correct MEIO approach automates the stocking and restocking process while also enabling sophisticated scenario analysis to automatically examine cost-service level trade-offs. It also employs machine learning to identify stocking patterns for seasonal or new product launches. It improves usability, user adoption, and user efficiency by utilizing powerful visualizations, MEIO dashboards, and notifications driven by events. It incorporates machine learning and optimization algorithms to model flows in inventory in a precise way through the various locations and the inter-related stages of a supply chain. It evaluates past behaviour under all conditions, resulting in a configuration that is optimal for inventory requirements and locations. Therefore, it is capable of handling any magnitude of demand and supply variability, as well as seasonality at the lowest possible cost. MEIO suggests particular ways for deferring inventory at earlier phases of the manufacturing and distribution processes, in addition to identifying the sources and types of excess stock maintained at various sites. The observed limitation is the suitability of the system for supply chains only, that have distributed supply centres.

NetStock (NetStock Brochure): is a low-cost cloud app that assists businesses in reducing surplus inventory, freeing up venture capital, reducing stock-outs, and improving revenue. It is intended to interact with ERP systems in order to provide visibility into inventory levels and investments, as well as to generate quality predictions and optimal replenishment recommendations. Visual dashboards show

overall inventory performance and provide early notice of issues that need to be addressed. It uses the ABC method to classify stock items, compute safety stock, and employs forecasting algorithms to avoid stockouts. The observed limits are similar to the previously described ABC limitations.

Other systems include EazyStock, SlimStock and Microsoft Dynamics365.

This study seeks to establish an optimum stock level for the retailer and the supplier individually for a single stock item subject to stock quantity limits. Furthermore, the optimum stock level for a particular item at the supplier premises is determined by the convergence of information from various retailers. On the other hand, the optimum stock level at the retailer premises for a single stock item is based on the demand pertaining to that item at his premises. Moreover, the level of demand, as a factor of cumulative statistics from various retail buyers at the supplier premises, is not necessarily similar to the demand at the retail premises. In addition, different retailers experience difference stochastic demand for a particular product based on various market factors such as location. The demand for a particular product in a rural setting might be different with its demand in urban centres. Additionally, based on competition from other similar products, a retailer may choose to lower his stock levels to avoid overstocking, yet the supplier demand predictors would indicate the need to increase stock to meet the demand, leading to overstocking. Therefore, it's prudent for a retailer to have access to a system that informs him/her of the optimum stock levels based on the performance at his/her premises. Likewise, a supplier should manage his/her stock levels based on the performance at his premises. The study also aims to extend stock optimization platform to small scale retailers.

2.15 Conceptual Model

The system framework (Figure 2.9) depicts the work flow and integration of the envisaged system components and user interactions. Apache Kafka will be adopted to for the speed layer, where the source data from various retailers and suppliers will be ingested and distributed within the system. In addition, the Cassandra will be adopted to serve as the database within the serving layer which will stores processed data and give response to ad hoc queries on the data. Retailers and suppliers will be able to query the system to obtain the optimized stock levels in their premises. Historical data will be used to optimize the model and improve its efficiency in computing the optimum stock levels. The model combines the safety stock model and the optimum order quantity model, and evaluates if it can satisfy the current demand, and at the same time lower the costs related to stock. The process will be iterated to achieve the optimized stock levels.

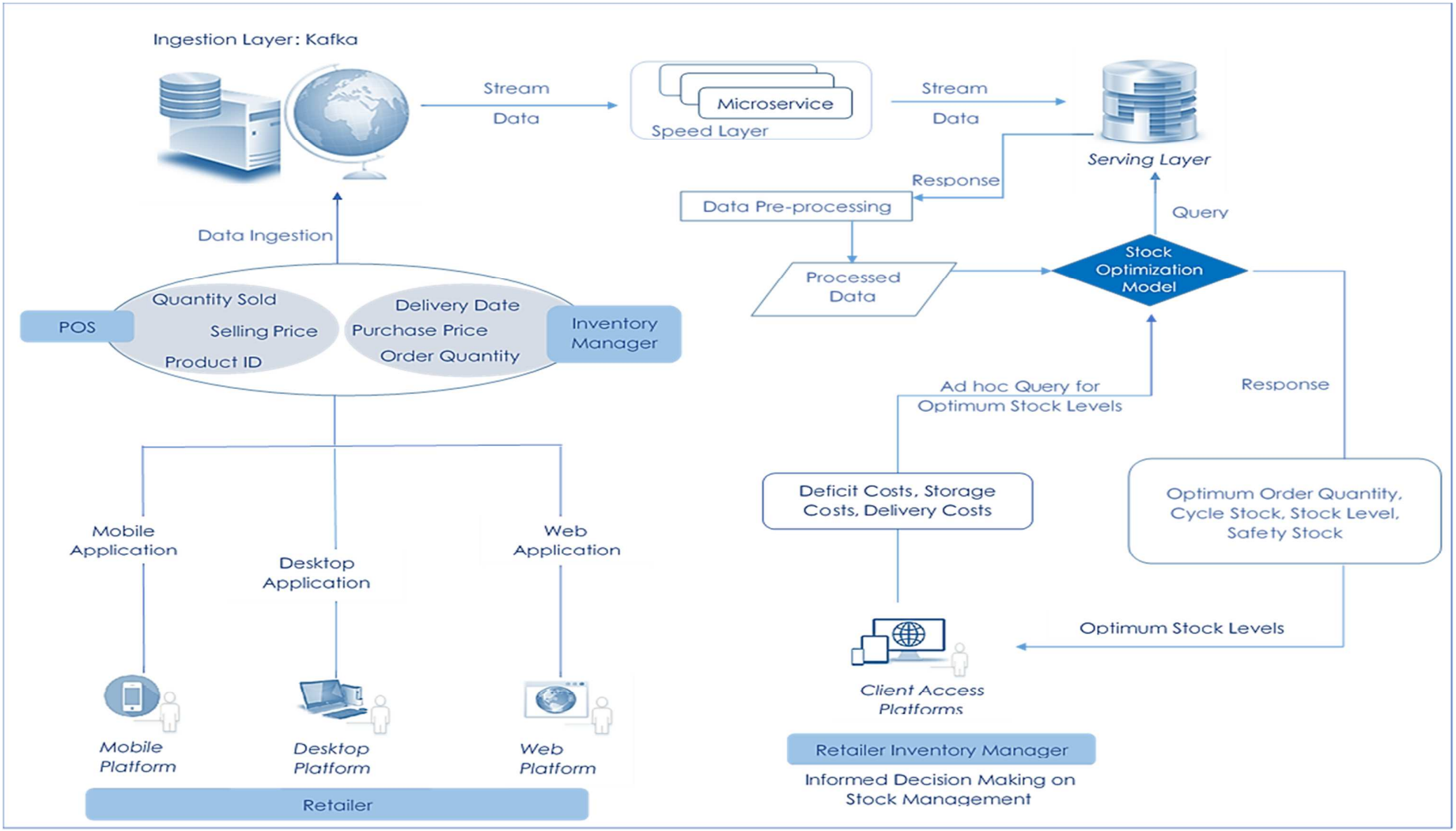


Figure 2.9: Conceptual Framework

Chapter 3: Methodology

3.1 Introduction

The aim of the study was to develop a system that enables the retailers and suppliers to optimize their stock levels, based historical data related to stock. The data formed the independent variables to the optimized stock level, which is the dependent variable. In addition, the proposed system would allow retailers and suppliers to share data in near real time. Thereafter, based on the information shared, retailers and suppliers can query the system to derive insights on the optimized stock levels. The specific objectives for the study were: To analyse the causes of limited stock optimization among retailers and suppliers; To review the existing models of stock optimization and the existing supporting technologies in information ingestion, storage and streaming; To develop a stock optimization model based on stock-related historical data; To test the model for validity and reliability.

The research plan, organization, and investigation approach were described in order to gain answers to the research questions (Kumar, 2005). Furthermore, the research design outlined the procedure that was to be used to answer the questions in a legitimate, objective, accurate, and cost-effective manner. Therefore, the goal of this research design was to develop an operational strategy for carrying out the numerous processes and tasks required to finish this study, as well as to guarantee that the procedures were sufficient to acquire valid, objective, and precise answers to the research questions.

3.2 Research Design

Research methodology can be perceived as the study of how research is carried out in a scientific way so as to systematically solve the research problem (Kothari, 2004). This study adopted *applied research* to solve the business challenge in optimization of stock. Additionally, this study adopted the evolutionary research process paradigm, which acknowledges that research (methods) change and evolve over time due to the prototyping approach of this study.

This study adopted *proportionate sampling* in selection of specific units to form a sample that represented the universe (Kothari, 2004). This technique was suitable for this study since the population constituted a number of subgroups that were immensely dissimilar in number. Therefore, achieving accuracy based on a simple random sampling was not attainable. The aim was to collect historical data for a supermarket given that, a supermarket represents the common commodity operations and variety of stock maintained by most retailers. Additionally, the secondary data obtained as historical data was used to select a model to optimize stock levels. The research was divided into two components: developing a suitable model to optimize stock levels based on historical data and the development of the system prototype (Figure 3.10) to

incorporate the stock optimization model. Based on the review of causes of limited stock optimization among retailers and suppliers in literature review section, system requirements were formulated.

The *System Prototyping of the Rapid Application Development* methodology was adopted for this research due to its flexibility that allowed alteration at any point in the development life cycle (Roth et al., 2013). It also enhanced rapid but incremental development of simplified versions of the proposed system. Furthermore, it allowed performing the analysis, design and implementation phases concurrently. Additionally, its emphasis on iteration enabled effective and timely rectification of identified system deficiencies. Therefore, the requirements, which were not initially clear, were taken care of iteratively. Moreover, with each iteration cycle, a functional version of the final system was delivered.

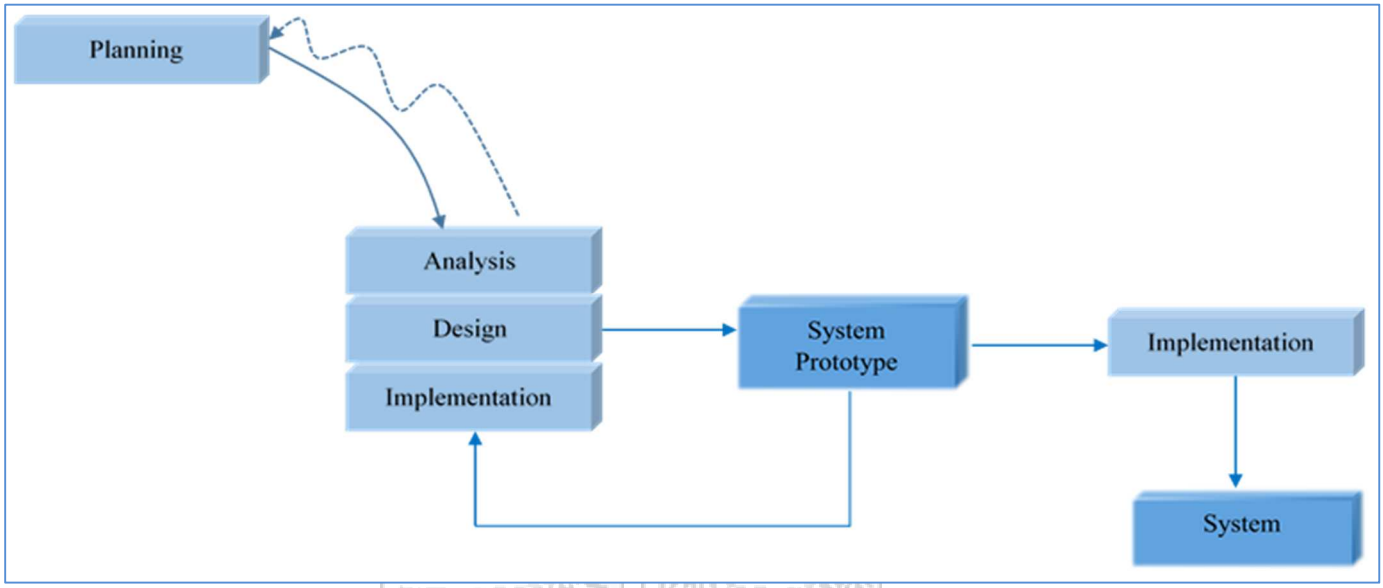


Figure 3.10: System Prototyping (Roth et al., 2013)

3.2.1 System Planning

The system information needs were identified, analyzed, prioritized and arranged in order to identify the tasks required for the system development. A Gantt chart was used to graphically represent the identified tasks and subsequent sequence of activities and allocated time to each task (Appendix G). The project scope and constraints were limited to the identified functional and non-functional requirements identified for the system.

3.2.2 System analysis and System Requirements

Problem analysis, Root-Cause Analysis and *Technology Analysis* strategies were employed for the requirements analysis. The Problem Analysis sought to identify the causes of lack of stock optimization to inform solutions to the challenges identified. Additionally, Root-Cause Analysis focused on generating root

causes for the problems to inform the type of appropriate mitigation measures as solutions to the problems. Lastly, Technology Analysis informed the existing technologies in use, and how more efficient technology was to be used to solve the identified challenges. Eventually, validation and verification of requirements were evaluated to ascertain if the correct requirements had been stated and if the requirements were stated correctly.

To convert a high-level declaration of business requirements into a working solution that fits the needs of the business process, process models (Data Flow Diagrams) were built and data models (Entity Relationship Diagrams) developed (Roth et al., 2013). The requirement analysis was scoped under two domains: system operations-*functional requirements* and system characteristics-*non-functional requirements*. Further, the functional requirements were categorized under process-oriented and information-oriented requirements. Also, the functional requirements were used as a guide in defining the data needed for the user tasks. Additionally, the process model was used to illustrate their inter-relations, how data was entered, processed, stored, created and produced by the system as well as the link between processes and system users. Moreover, the process models were utilized to aid in the identification of the software components required to achieve the functional requirements. (Figure 3.11).

The non-functional requirements were further evaluated under four key domains: *Usability* - the user interactive interface; *Performance* - capacity, speed; system *Reliability* – how well the system does what it was intended to; *Availability* – how accessible is the system when needed; *Portability* – how interoperable the system is; *Security* – authorization and access; *Cultural and Political* – business political and cultural environmental factors. To ensure the system development success, the evolution of the requirements was kept in check by keeping the requirement list rigid. Thereafter, the functional and non-functional requirements were outlined in the *requirements definition statement,* which defined the scope of the system. This assisted prioritize the requirements that influenced the version control in system prototyping delivery.

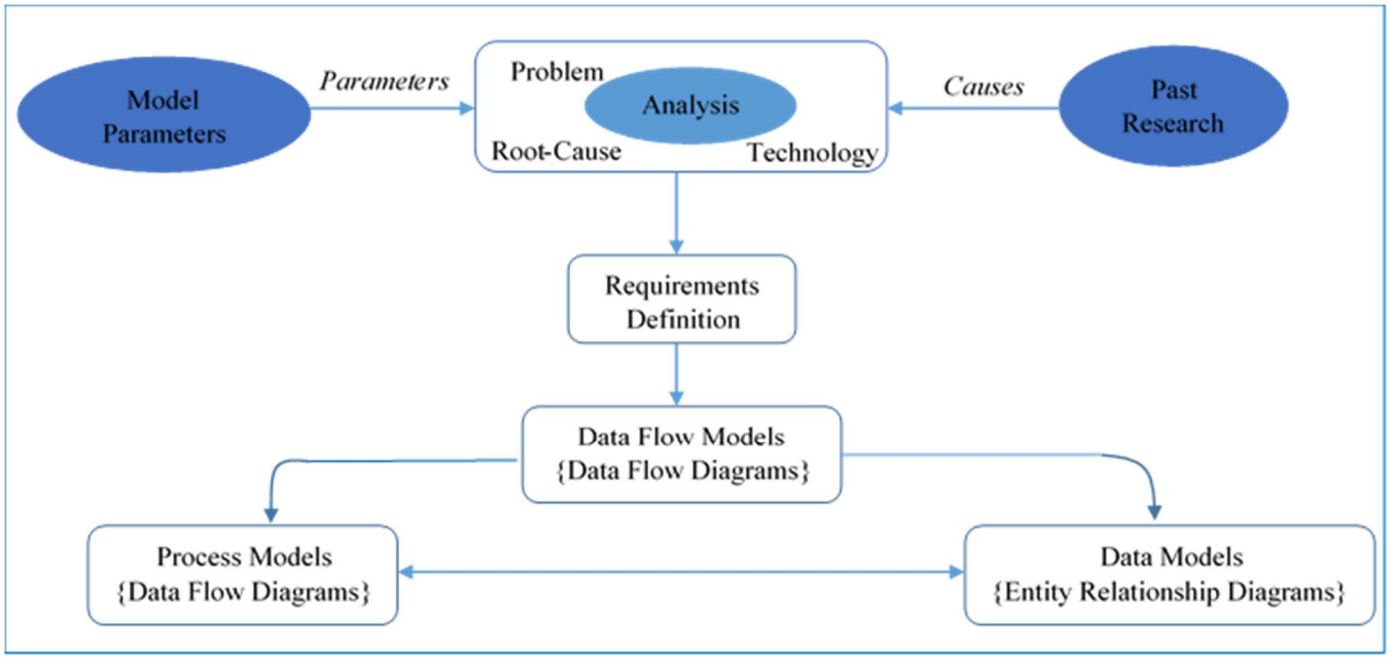


Figure 3.11: System Analysis and Requirements

3.2.3 System Design

The analysis of the system requirements formed the basis for the system design in generating the specification of the system which included the physical process models, the physical data models, the architectural design, software and hardware specifications, data storage design, interface design and program design (Figure 3.12). Based on the requirements definition statement, the system architecture was formulated showing the users, software, hardware, physical components and inter-relations among them to meet the requirements of the system. Further, the requirements were translated to system requirements that informed the technical details of the system. The literature review in Chapter 2 was used as the reference in selecting an appropriate architecture with respect to the identified system needs. Collectively, the physical process models and the design documents formed the blueprint of the system.

The Structured System Design was employed for this study. Data Flow Diagrams analysis was applied to formulate the data flow processes representation of the system. In addition, to ensure modularity, the DFDs diagrams informed the various modules required for the system. Entity Relationship Diagram were used to depict the relationships among entities. Entity Relationship Diagrams (ERDs) with Data Dictionaries were utilized in designing the data models as a blueprint to the physical data structures.

The User Interface was designed based on the understanding of the business process relating to stock optimization, graphical effectiveness and user-centric principles.

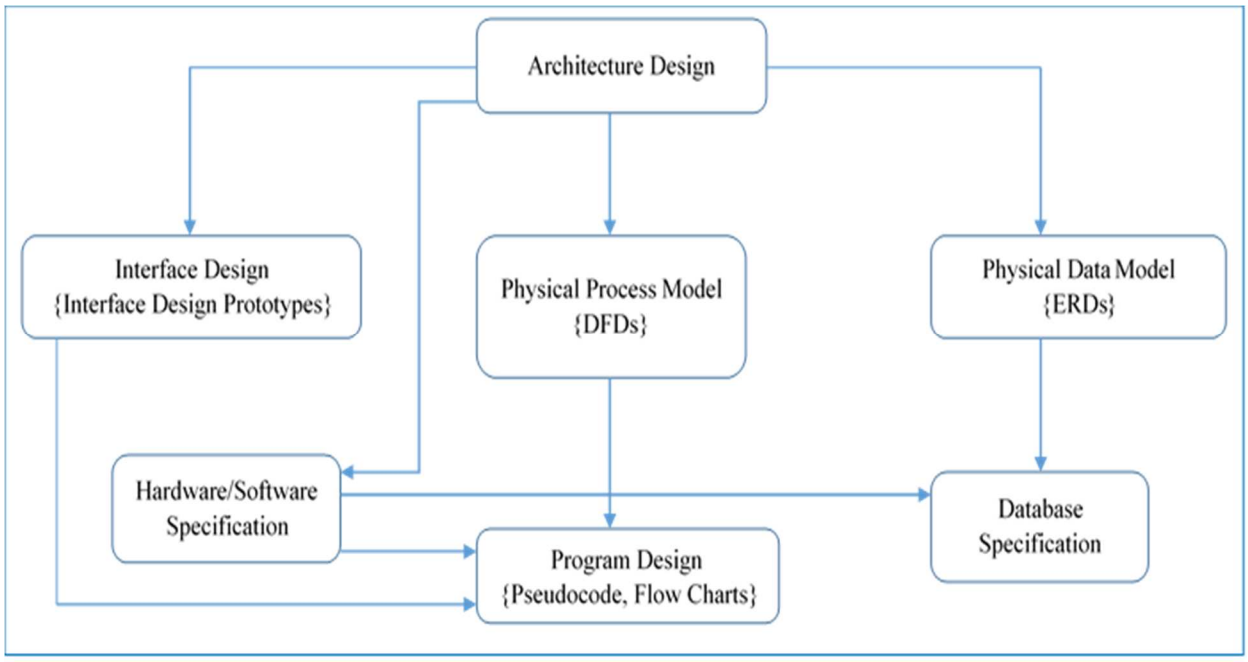


Figure 3.12: System Design (Adopted from Roth et al., 2013)

3.2.4 Stock Optimization Model Design

Based on the Literature review in Chapter 2, a model that optimizes stock levels was formulated. Historical data pertaining to stock was collected from selected suppliers and retailers, and used in the model to test for efficiency. The outcome of the model was evaluated and compared to the historical data. The model was validated empirically and tested for reliability. Thereafter, the model was translated to a programmable code that was incorporated into the system.

3.2.5 System Implementation

The proposed system was realized through coding, testing the prototypes and installation of the system (Figure 3.13). The system development was broken down into *programmable tasks* based on the modules outlined in the design phase. Java programming language was used for the server side code modules and functionality. HTML5 and JavaScript were used for Front-End Web Development. YugabyteDB Database was used for the storage of data. Apache Netbeans was used for coding.

A Test Plan was developed to define the tests to be undertaken based on the three Software-Under-Test dimensions: *Test Objective* (Functional, non-functional); *Granularity Level* (algorithm, unit, integration, system testing); and *Execution Level* (Dynamic Level: developmental testing). Developmental testing was used to test for program logic using debugging methods. *Black-Box Unit Testing* was used to test the ingestion, streaming and storage layers. *Integration Testing* was conducted to assess if individual modules

work together effectively. The envisaged testing of integration approaches included: use scenario testing, user interface testing, system interface testing and data flow testing to test exchange of data. *System Tests* were conducted to guarantee that all modules and programs operated flawlessly in terms of performance, usability and security.

System documentation and high level *User Documentation* was maintained throughout the development phase for future reference.

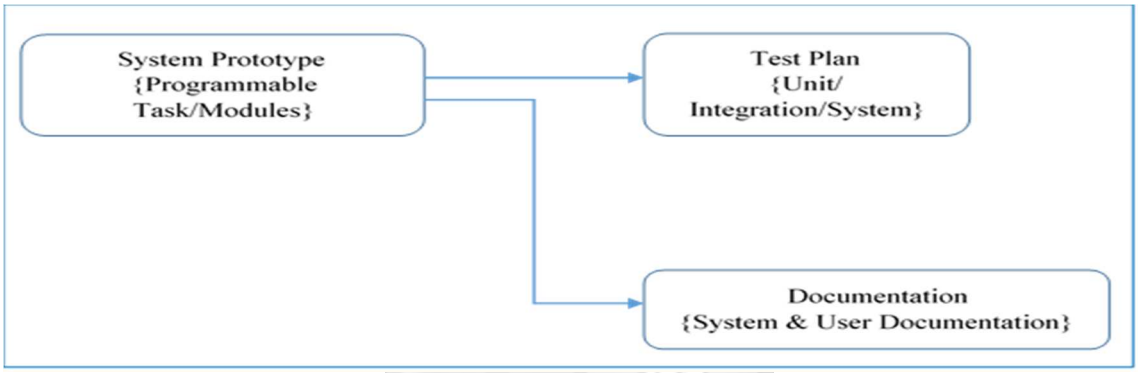


Figure 3.13: System Implementation

3.2.6 System Prototyping

Based on the identified functional and non-functional requirements, a system prototype was developed iteratively. The prototype built at each iteration was tested and then reworked to achieve the desired system. With each iteration, additional functionalities were developed, tested and added to the prototype. At the final iteration, the prototype was evaluated to contain all the required functional and non-functional requirements.

3.3 Target Population and Sample Frame

The commodity products stocked in a supermarket formed the finite universe for this study. The study focused on processed and packaged products retailed in a supermarket. The range of products included groceries such as packaged eggs, cakes, bottled water and beverages, canned beverages such as beer, books, electronics, confectionery, dairy products, press material such as newspapers, toiletries and snacks. The study was limited to a retailer i.e. mini-supermarket. The study was limited to Nairobi County based on the assumption that Nairobi being a city has most of the retailers. The sampling unit was the retail stock commodity products. This study adopted the proportionate sampling technique such that the number of units selected from each subgroup were determined by their sum total with respect to the whole population. The city has numerous retailers, therefore, owing to the limited resources, a representative sample was used to conduct the study. Additionally, having the assumption of uniformity in the retail of commodities across

various retailers, and that most retailers stock products from the same supplier, and vice versa, a representative sample sufficed for this study. The supply data for each product was obtained from the retailer, indicating the product name and identification code, the quantity supplied, the purchase price and the date supplied.

3.4 Data Collection

Historical commodity stock data was obtained from selected retailer records. In particular, the secondary data of the selected commodity products in the sample was obtained from a mini-supermarket. The attributes of the data were informed by the parameters to the models discussed in Chapter 2. For service level, data on the total quantity of demand and the number of quantities delivered on time were collected. To derive the demand over a period, the quantity sold for a stock item was collected from selected retailer. With respect to safety stock, replenishment lead times were gathered. Further, the purchasing price of items and number of orders within a time frame were collected.

3.5 Data Analysis

Data collected was validated, edited, cleaned-up, coded, classified according to attributes, tabulated and processed for further analysis. The uni-dimensional analysis entailed computation of percentages, frequencies, arithmetic mean and standard deviation. The reports were documented through tables, bar graphs and charts using Microsoft Excel. Data analysis on the historical data was carried out using Microsoft Excel. For each product selected for analysis, the total demand, average demand, the standard deviation of demand, the coefficient of variation for demand, unit cost of replenishment, stock deficit cost, stock-carrying cost coefficient and average replenishment lead time were computed. From the supply data, the purchase price and the quantity supplied for each of the product was extracted. The output of the quantitative data analysis was used to develop the stock optimization model.

3.6 Research Quality Aspects

The research quality was evaluated for relevance, credibility, reliability and validity. The research was assessed for relevance based on data that was needed and its relation to the research. To assess credibility, the right rationale was evaluated based on appropriateness of the research methods and how believable the data was. Validity to ascertain that the system did what it was intended to do was assessed empirically by evaluation of the model outcome in comparison to actual values. Content validity (logical evidence) was assessed by determining how well the research questions captured what the system was designed to do through external audits. Reliability was maintained by ensuring consistency and uniformity in the data collection and recording formats. The study ensured rigorous data profiling through examining aspects of

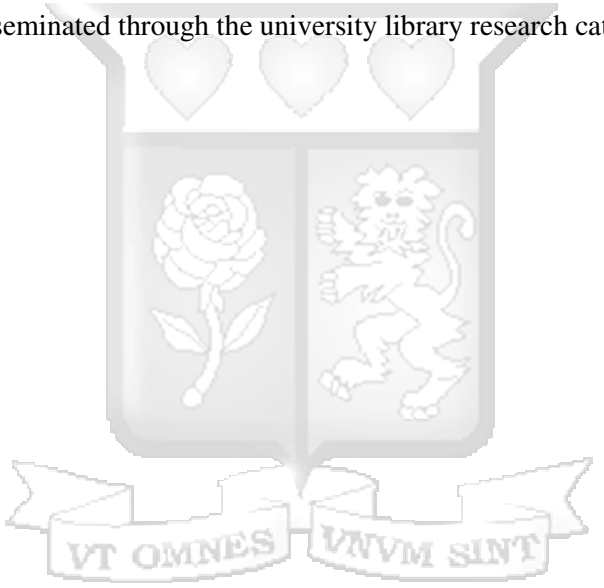
data such as data format, patterns, consistency on each record, anomalies and completeness. Duplicate data was examined to reduce biasness.

3.7 Ethical Considerations

Institutional ethical approval was sought from Strathmore University. While beginning to acquire the data, the purpose of study was disclosed to the participants in advance. The data request was accompanied by a cover letter to introduce the institution and the purpose of the study, the study objective, contact number for queries and appreciation remark. The source data owners were not exploited during collection of data. Data collected was stored securely, and was not shared with other parties, but was constrained to the study only. Privacy of participants was respected by assigning aliases during data analysis.

3.8 Research Dissemination Plan

The research report was disseminated through the university library research catalogue repository.



Chapter 4: System Analysis, Design and Architecture

4.1 Introduction

The literature review and conceptual framework in subsection 2.15 formed the basis for defining the system requirements. Additionally, a description of how the data was gathered and analyzed has been outlined. In addition, the methods and procedures of how data was analyzed has been elaborated. Furthermore, the output of the data analysis has been presented using various appropriate methods of data presentation. The requirements have been categorized into functional and non-functional requirements and evaluated in detail. Following the identified system requirements, the system design has been formulated in line with the defined needs. In particular, the system stakeholders, modules, data models and processes have been stated. To illustrate the system design, structured approach through data flow diagrams have been used for the detailed comprehension of the system workflow and functionality.

4.2 Data Collection and Analysis

Secondary data from publications were reviewed for the causes of lack of stock optimization among retailers (subsection 2.7). Additionally, the existing stock optimization approaches and models as well as their limitations were reviewed (subsection 2.12.5). Moreover, the supporting technologies and their limitations were also reviewed (subsection 2.13.3). Secondary data from records was obtained from a mini-supermarket related to commodity products. The data set consisted of numerical discrete data pertaining to sales and supply for various commodities stocked in the supermarket. Additionally, the data request was structured (formatted) to contain attributes aligning with the required parameters to the model (Appendix A). The request to obtain the data was placed to the sales administrative supervisor of the mini-supermarket. An introductory letter from the School of Computing and Engineering Sciences in conjunction with the Ethical Approval Letter were shared for acknowledgement. The data was shared via email. The data was offered on condition that it was for academic use only and the data source would be profiled accordingly. Additionally, the products would be given aliases for research purposes. The data shared was for a period of three months (Appendix B). The three-month period was as a consequence of the sales administrator taking caution over the sensitivity of the data.

The data was constrained to selected products for analysis and developing the optimization tool. The data was cleaned up and edited, so as to format it for further processing with respect to the factors that influence stock levels which are demand, cost of stock replenishment, purchase price, service level and the lead time. Data was organized into different groups and assigned a code, the data was then aggregated under each group for the received data, and thereafter enumerated for quantification analysis. Based on the aggregated data, frequencies, percentages and arithmetic mean were computed. To further elaborate the results, reports were generated and presented through tables and charts. The

data was then formatted to allow analysis for computation of the standard deviation of demand, the coefficient of variation for demand, average daily demand and average replenishment lead time.

4.1.1 Data Clean-up and Editing

The raw data was separated into two datasets, i.e. tabulated as either supply or sales datasets. The datasets were then ordered to align with the tabulation in Appendix C. Columns in the raw data that were not related to the model parameters were discarded. Thereafter, products were assigned alphanumeric unique aliases for ease of identification, profiling and the subsequent analysis.

4.1.2 Product Proportionate Sampling

The single product to be used in the model was selected through proportionate sampling of the data (equation 32). The rationale was to obtain and rank the products from high-to-low moving products based on sales, then select the high moving product. The sample size n=671, was computed from the number of unique products in the data. The proportionate value of each product was then computed (equation 33). The product with the highest proportionate value was selected for further analysis (Appendix D). Product alias A20, a canned beverage, with 228 sold pieces within the 3 months was selected (Figure 4.14).

$$\%\ of\ Product\ A = \frac{Sales_A}{Total\ Sales} * 100\% \dots\dots\dots\dots\dots\dots (32)$$

$$\%A * n = k \dots\dots\dots\dots\dots (33)$$

$$where\ k\ is\ the\ proportionate\ value\ of\ product\ A$$

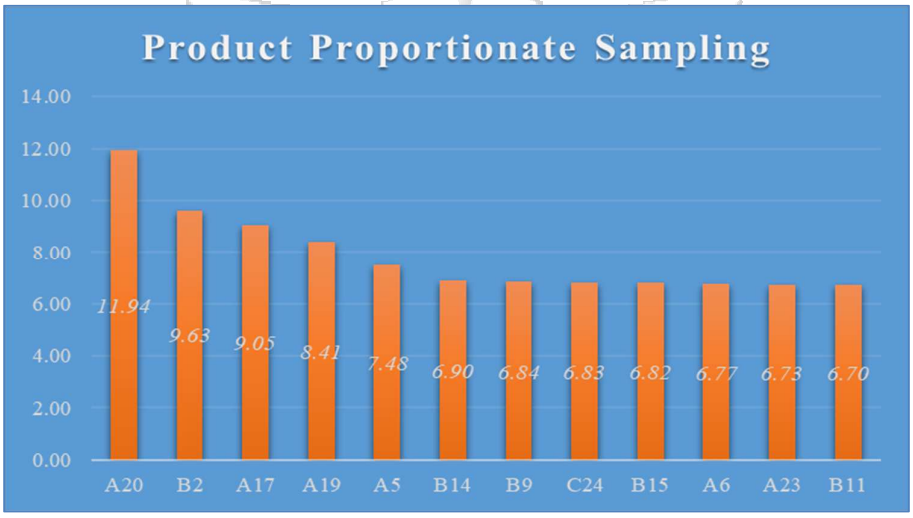


Figure 4.14: Proportionate Sampling

4.1.3 Products Statistics Analysis

The supply data for the alias product A20 was extracted (Table 4.10). The sales data for the product was summarized in (Table 4.7). A comparison of quantity supplied and quantity sold for each month has been tabulated in (Table 4.8).

Table 4.6: Alias Product A20 Supply Data

Product Alias	Date	Delivery Time Interval (Days)	Quantity (Units)	Purchase Cost
A20	12/05/2022		24	Ksh 4,768.10
A20	12/12/2022	7	24	Ksh 4,974.00
A20	12/19/2022	7	24	Ksh 4,974.00
A20	12/23/2022	4	24	Ksh 4,974.00
A20	12/26/2022	3	12	Ksh 2,487.00
A20	01/09/2023	14	24	Ksh 4,974.00
A20	01/11/2023	2	12	Ksh 2,487.00
A20	01/20/2023	9	12	Ksh 2,487.00
A20	01/27/2023	7	24	Ksh 4,974.00
A20	02/08/2023	12	12	Ksh 2,487.00
A20	02/13/2023	5	12	Ksh 2,487.00
A20	02/15/2023	2	12	Ksh 2,487.00
A20	02/22/2023	7	12	Ksh 2,487.00

Table 4.7: Alias Product A20 Sales Summary

Product Alias A20	December	January	February
Quantity Sold (Units)	107	72	47
Total Sales (Ksh)	29960	20719	13159

Table 4.8: Alias Product A20 Supply-Sales Comparison

Product Alias A20	December	January	February
Quantity Supplied (Units)	108	72	48
Quantity Sold (Units)	107	74	47

Based on the data above, it can be observed that the approach applied for replenishment of stock is Just-in-Time (Figure 4.15).

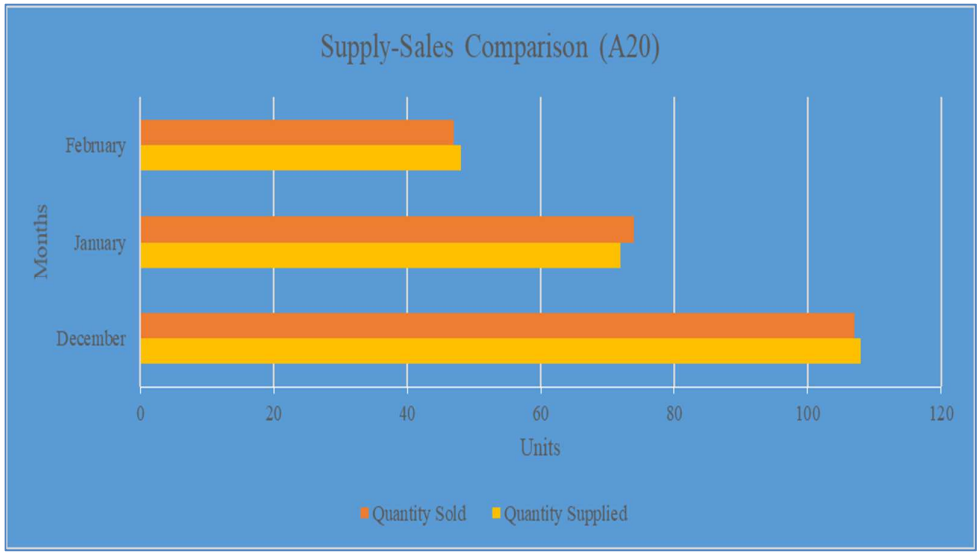


Figure 4.15: Alias Product A20 Supply-Sales Chart Comparison

Additionally, the supply data for three more high moving alias products (A5, A17 and A19) were analysed and tabulated in Table 4.9, Table 4.10 and Table 4.10 respectively. Although alias Product B2 was a high moving product, it was lacking supply data. The supply versus sales comparisons were tabulated in Table 4.12, Table 4.13, Table 4.14 and represented through charts in Figure 4.16, Figure 4.17 and Figure 4.18.

Table 4.9: Alias Product A5 Supply Data

Product Alias	Date	Delivery Time Interval (Days)	Quantity (Units)	Purchase Cost
A5	12/01/2022		120	Ksh 7,080.00
A5	12/10/2022	9	23	Ksh 1,357.00
A5	12/12/2022	2	24	Ksh 1,416.00
A5	12/19/2022	7	60	Ksh 3,540.00
A5	01/03/2023	15	60	Ksh 3,540.00
A5	01/13/2023	10	24	Ksh 1,416.00
A5	01/20/2023	7	60	Ksh 3,540.00
A5	02/02/2023	13	60	Ksh 3,540.00
A5	02/06/2023	4	24	Ksh 1,416.00
A5	02/11/2023	5	24	Ksh 1,416.00
A5	02/16/2023	5	18	Ksh 1,062.00
A5	02/19/2023	3	24	Ksh 1,416.00
A5	02/21/2023	2	24	Ksh 1,416.00
A5	02/27/2023	5	60	Ksh 3,540.00

Table 4.10 : Alias Product A17 Supply Data

Product Alias	Date	Delivery Time Interval (Days)	Quantity (Units)	Purchase Cost
A17	12/03/2022		20	Ksh 2,160.00
A17	12/05/2022	2	20	Ksh 2,160.00
A17	12/10/2022	5	15	Ksh 1,620.00
A17	12/13/2022	3	20	Ksh 2,160.00
A17	12/19/2022	6	10	Ksh 1,080.00
A17	12/23/2022	4	10	Ksh 1,080.00
A17	12/28/2022	5	20	Ksh 2,160.00
A17	12/31/2022	3	5	Ksh 540.00
A17	01/06/2023	6	14	Ksh 1,512.00
A17	01/07/2023	1	6	Ksh 648.00
A17	01/10/2023	3	20	Ksh 2,160.00
A17	01/14/2023	4	10	Ksh 1,080.00
A17	01/20/2023	6	23	Ksh 2,484.00
A17	01/26/2023	6	18	Ksh 1,944.00
A17	01/30/2023	4	5	Ksh 540.00
A17	02/01/2023	2	30	Ksh 3,240.00
A17	02/06/2023	5	6	Ksh 648.00
A17	02/08/2023	2	10	Ksh 1,080.00
A17	02/11/2023	3	18	Ksh 1,944.00
A17	02/17/2023	6	10	Ksh 1,080.00
A17	02/21/2023	4	15	Ksh 1,620.00
A17	02/25/2023	4	17	Ksh 1,836.00
A17	02/28/2023	3	14	Ksh 1,512.00

Table 4.11: Alias Product A19 Supply Data

Product Alias	Date	Delivery Time Interval (Days)	Quantity (Units)	Purchase Cost
A19	12/05/2022		18	Ksh 2,250.00
A19	12/12/2022	7	31	Ksh 3,875.00
A19	12/19/2022	7	9	Ksh 1,125.00
A19	12/26/2022	7	23	Ksh 2,875.00
A19	01/03/2023	8	19	Ksh 2,375.00
A19	01/11/2023	9	10	Ksh 1,250.00
A19	01/13/2023	2	16	Ksh 2,000.00
A19	01/16/2023	3	26	Ksh 3,250.00
A19	01/20/2023	4	8	Ksh 1,000.00
A19	01/23/2023	3	10	Ksh 1,250.00
A19	01/30/2023	7	26	Ksh 3,250.00
A19	02/06/2023	7	26	Ksh 3,250.00
A19	02/11/2023	5	15	Ksh 1,875.00
A19	02/16/2023	5	24	Ksh 3,000.00
A19	02/19/2023	3	21	Ksh 2,625.00
A19	02/21/2023	2	8	Ksh 1,000.00
A19	02/27/2023	6	18	Ksh 2,250.00

Table 4.12: Alias Product A5 Supply-Sales Comparison

Product Alias A5	December	January	February
Quantity Supplied (Units)	227	144	234
Quantity Sold (Units)	213	163	186

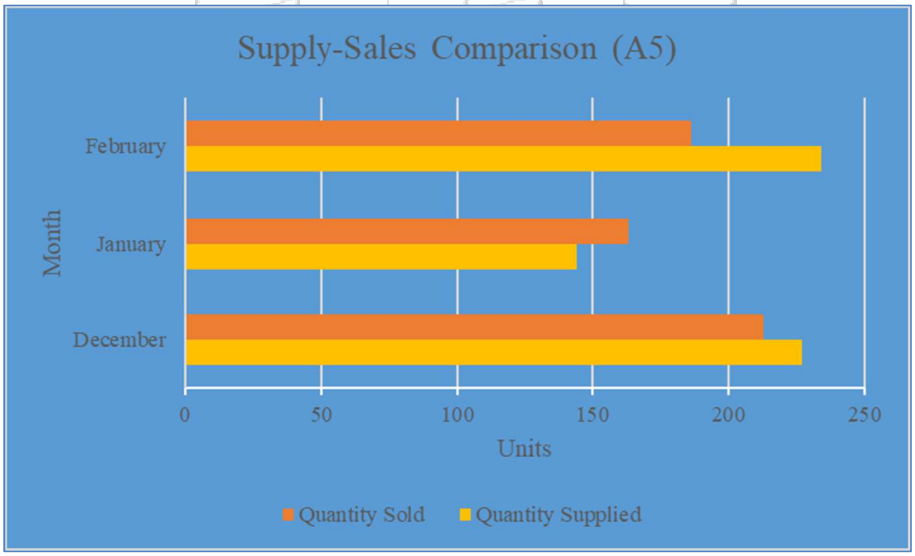


Figure 4.16: Alias Product A5 Supply-Sales Chart Comparison

Table 4.13: Alias Product A17 Supply-Sales Comparison

Product Alias A17	December	January	February
Quantity Supplied (Units)	120	96	120
Quantity Sold (Units)	108	113	102

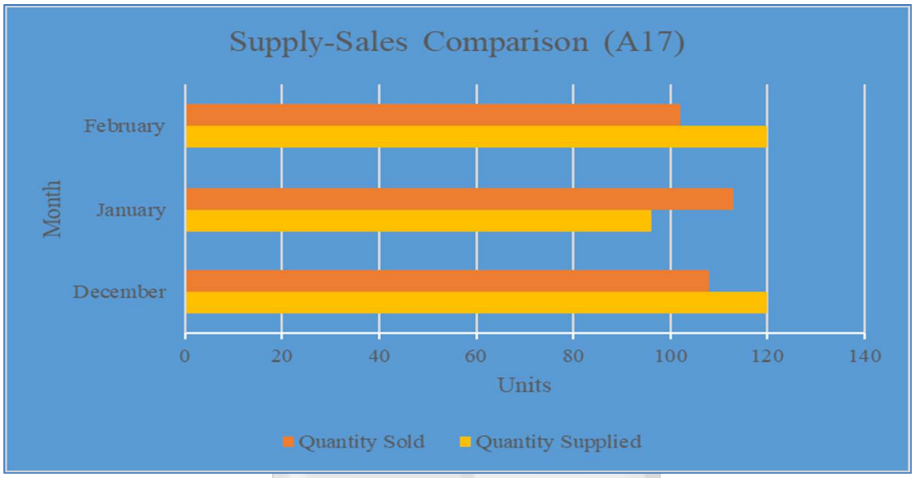


Figure 4.17: Alias Product A17 Supply-Sales Chart Comparison

The sales for alias product A5 and A17 in the month of January appears higher than the amount supplied. This can be attributed to the cross-over stock inventory from the previous month of December.

Table 4.14: Alias Product A19 Supply-Sales Comparison

Product Alias A19	December	January	February
Quantity Supplied (Units)	81	115	112
Quantity Sold (Units)	99	93	108

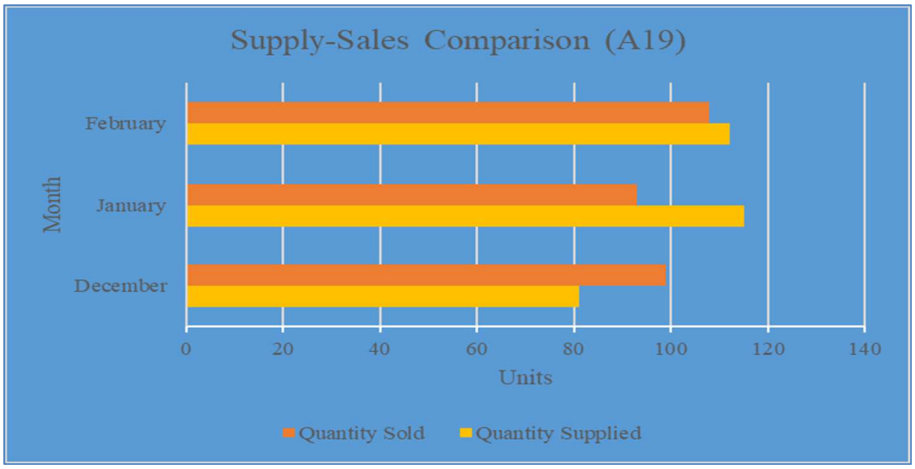


Figure 4.18: Alias Product A19 Supply-Sales Comparison

The sales for alias product A19 in the month of December appears higher than the amount supplied, implying there was a cross-over stock inventory from the previous month of November. The trend in supply frequency observed in the product supply data tabulation indicate a Just-in-Time approach of replenishing stock inventory (Table 4.15). This approach was corroborated by the retailer supervisor of the source data.

Table 4.15: Product Supply Frequencies

Product Alias	December	January	February
A20 (Frequency)	5	4	4
A5 (Frequency)	4	3	7
A17 (Frequency)	8	7	8
A19 (Frequency)	4	7	6

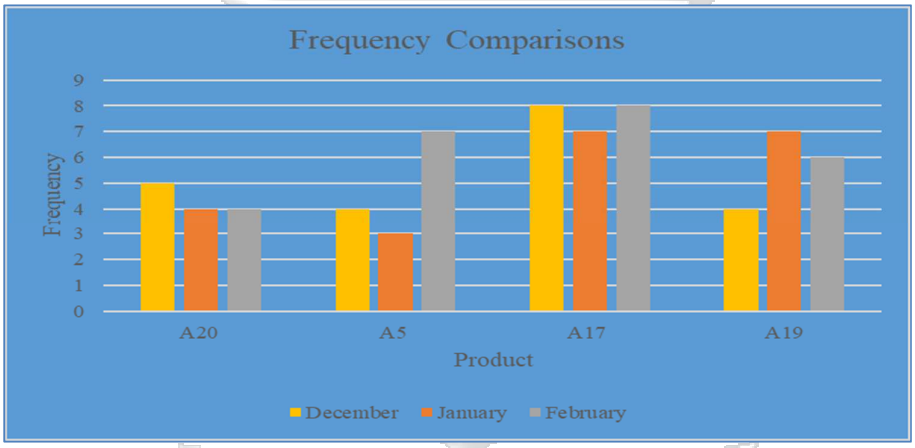


Figure 4.19: Supply Frequency Comparisons

The differences (margins) in the supplied quantities and sales for the selected products, as well as the average of the margins were computed, tabulated (Table 4.16) and represented graphically (Figure 4.20). It was observed that although there is no indication of understocking in the inventory approach adopted by the supplier and retailer, on average, the inventory level is higher than the demand. Apart from alias product A20 and A17, there is significant variation between the minimum and maximum margins, indicating lack of optimization of stock inventory, which is evident in the computed standard deviation of the margins (Table 4.16).

Table 4.16: Alias Products Supply-Sales Margin Statistics

Product Alias	Min	Max	Average	Standard Deviation
A20	1	2	0.5	0.5
A5	14	48	27	18
A17	12	18	16	3
A19	4	22	15	9

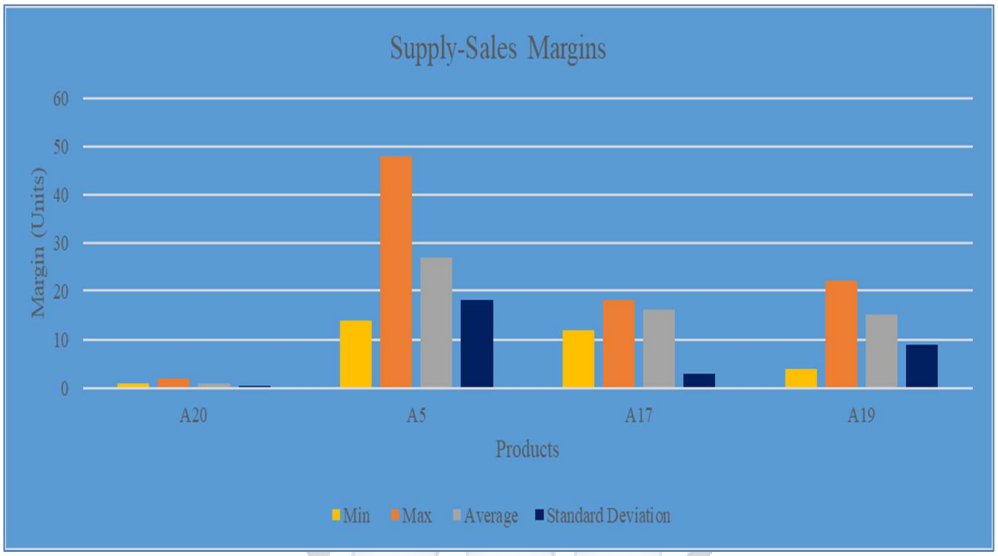


Figure 4.20: Supply-Sales Margin Chart Comparisons

4.1.4 Model Parameter Statistics

Using the data for the selected alias product A20, the coefficient of variation (CV) of demand (D) and lead time (LT) were computed Table 4.17. The lead times were computed from the average of delivery time interval. Additionally, the same calculations were repeated assuming a constant lead time of one day. Moreover, the calculations were done for each month. The covariance of demand was assumed to be constant (by observing that the prices were held at a constant during the three months).

Table 4.17: Coefficient of Variation and Lead Times

Period	Average LT (Days)	Sq. Root LT	Average Demand	CV_LT	CV_D
Cumulatively 3-Month	7	2.56	2.50	0.17	0.35
Cumulatively 3-Month	1	1	2.50	0	0.35
December	5	2.29	3.45	0.39	0.35
January	8	2.82	2.38	0.62	0.35
February	6	2.54	1.67	0.64	0.35

4.1.5 Model Validity

The model was tested for validity by comparing the results to the historical data and evaluating the differences between the values. The optimum order quantity and service level from the model output was compared to the historical supply and sales data and tabulated in Table 4.18. The computed optimum service level is less than the service level computed from the historical data. This is attributed to the Just-in-Time replenishment approach, where stock was replenished after complete depletion of the available stock. The optimum order quantity is less than the average supply by a margin of 2 units,

translating to a ratio of 0.11 (approximately 11%) difference. Consequently, it is observed that on average, the supplier was either over supplying or undersupplying. This is evident by comparing the value with the maximum and minimum supply within the replenishment period of three months. The computed optimum order quantity falls below the maximum supplied quantity and above the minimum supplied quantity, implying that the model performs better than the existing Just-in-Time approach. The model results are represented through a chart in Figure 4.21.

The optimum safety stock was compared to computed safety stock using classical approaches (Table 4.19). Two classical approaches were employed for this test. The first classical approach assumes that safety stock should be between 10% to 20% of average stock. The second approach computes safety stock as a product of the standard deviation within the replenishment lead time. The average stock was computed from quantity supplied within the three months. It is observed that the safety stock computed from the model is within the classical percentage range. Additionally, it yields a lower value than the safety stock computed from the standard deviation of demand within the replenishment lead time. The results are compared through a chart in Figure 4.22.

Table 4.18: Model Validation

	Service Level	Average Supply (Units)	Maximum Supply (Units)	Minimum Supply (Units)
Historical	100%	17	24	12
Computed	95.2%	15	15	15
Difference	4.8%	2		

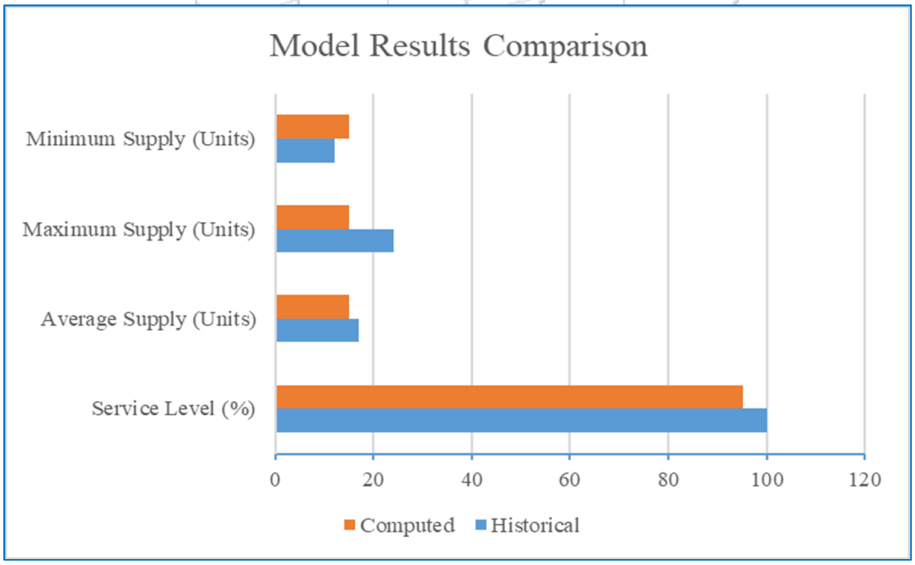


Figure 4.21: Model Results Comparison

Table 4.19: Safety Stock Comparison

Approach	10-20%	Service Factor * σ_{DLT}	Model
Safety Stock	18%	16 units	14 units

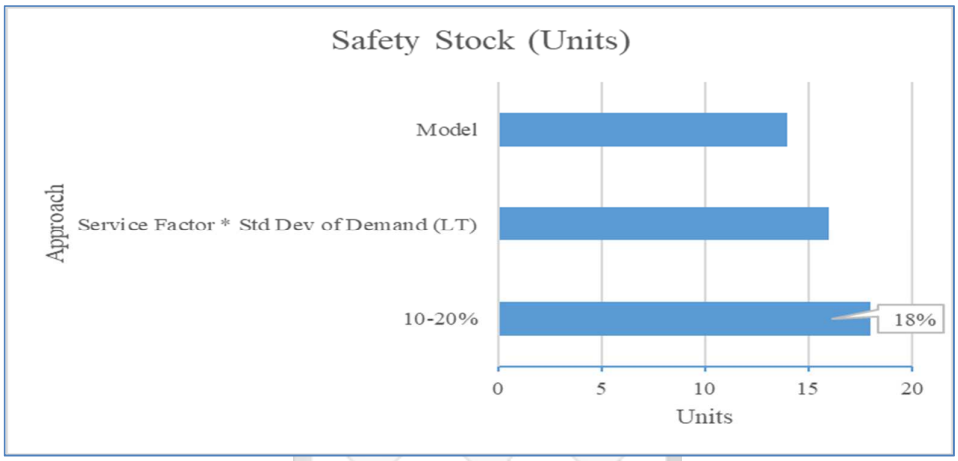


Figure 4.22: Safety Stock Chart Comparisons

4.1.6 Model Reliability

The model was tested for reliability by using a different set of data and evaluating the consistency in the results computed. The model was tested for consistency using another high moving product A17 (Table 4.20). The results are tabulated below (Table 4.20). The model results were validated against the historical data (Table 4.21). The model yields a service level of 96.5%, implying that approximately 96% of the periods the model is able to cover for the customer demand. Further, the model yielded a safety stock level within the 10-20% range and lower than the classical approach based on standard deviation of demand during lead time (Table 4.22). Thus, the model was able to compute optimized stock level for another product consistently.

Table 4.20: Product A17 Model Results

Optimum Order Quantity	Optimum Safety Stock	Cycle Stock	Optimum Stock	Optimum Service Level	Average Lead Time
17 units	12 units	9 units	21 units	96.5%	3 days

Table 4.21: Product A17 Model-Historical Data Comparison

	Service Level (%)	Average Supply (Units)	Maximum Supply (Units)	Minimum Supply (Units)
Historical	104	15	30	5
Computed	96.5	17	17	17
Difference	7.5	2		

Table 4.22: Product A17 Safety Stock Validation

Approach	10-20%	Service Factor * σ_{DLT}	Model
Safety Stock	11%	15 units	12 units

4.1.7 System Requirement Analysis

The Problem analysis, Root-Cause Analysis and Technology Analysis strategies were employed to inform the system requirements analysis. In addition, the stock optimization model functionality further contributed to the formulation of the requirements.

System Requirements

The system requirements refer to a statement that defines what the system must perform and the characteristics it needs to have for it to satisfy the business and user need (Roth et al., 2013). Therefore, the system requirements have been categorized into functional (i.e. what the software should do for the users) and non-functional (i.e. characteristics the system is to have with respect to quality attributes, implementation constraints, and external interfaces) requirements. The derived information from the analysis scoped under each objective under literature review formed the basis for requirements analysis through the aforementioned analysis strategies in subsection 3.2.1. Identified gaps further informed the requirements. Specifically, the business and user needs were derived from the review of causes of lack of stock optimization, to determine the means by which the developed system can support the stated needs, leading to the functional and non-functional requirement statement.

4.1.8 Functional Requirements

The functional requirements with respect to the system capabilities that the system must perform for the users, relating to the processes that support a user task, as well as the required information required to accomplish the task (Roth et al., 2013) were outlined as follows:

Table 4.23: Functional Requirements

Reference Number	Requirement
Process-Oriented	
FRQ1	The system should be able to capture the stock optimization parameters such as quantity sold, lead time, purchase price, constraint such as mass or cost of storage into the system for a single product
FRQ2	The system should be able to process the data to compute the optimized stock level based on the optimization model a. Compute demand b. Compute standard deviation of demand over the DDLT c. Compute standard deviation of lead time over the DDLT d. Incorporate the parameters into the model e. Compute the optimum stock based on the model
FRQ3	The supplier and retailers should be able to obtain optimized stock quantity information for a single product
Information-Oriented	
FRQ4	The system should be able to retain the (historical) data
FRQ5	The system must include real-time inventory levels at the retail or supply premises

4.1.9 Non-functional Requirements

The non-functional requirements defining the quality attributes, design and implementation constraints, the external interfaces the system should have with respect to the usability, reliability, performance and supportability has been outlined (Roth et al., 2013). This entailed:

Table 4.24: Non-Functional Requirements

Reference Number	Requirement
Usability	
NFRQ1	The system should provide a user interface for the user to conveniently access the system (through web and mobile platforms)
NFRQ2	The system should provide the user with an interface to easily specify the required parameters for a single product without technical support
NFRQ3	The system should support integration and interoperability with existing retailer or supplier inventory systems through APIs
Reliability	
NFRQ4	The system must perform without failure during working hours
NFRQ5	The incoming data from retailer or suppliers should be correctly captured by the system continuously
Availability	
NFRQ6	The system should be available for use (uptime) during working hours through mobile and web platforms
Portability & Compatibility	
NFRQ7	The API should support cross platform capabilities
NFRQ8	The web application should have cross-platform, cross-browsing, and mobile-responsive capabilities
Performance	
NFRQ9	The system should be able to update new status parameters within 1 hour of a change
NFRQ10	The system should support simultaneous users at all times
Security	
NFRQ11	The system should only permit authorized retailers or suppliers to access the system
NFRQ12	Data integrity should be maintained to ensure accuracy in computations
Cultural and Political	
NFRQ14	Data privacy should be maintained to ensure data pertaining to each entity is properly partitioned

System Architecture

The software architecture of computing system is the collection of structures required to reason about the system, which consists of software, pieces, relations between them, and qualities of both (Richardson, 2019). It refers to the architectural design for the system describing the hardware, software

and network infrastructure to be used. The goal of the architectural design is to define how the information system's software components will be assigned to the system's hardware elements (Roth et al., 2013).

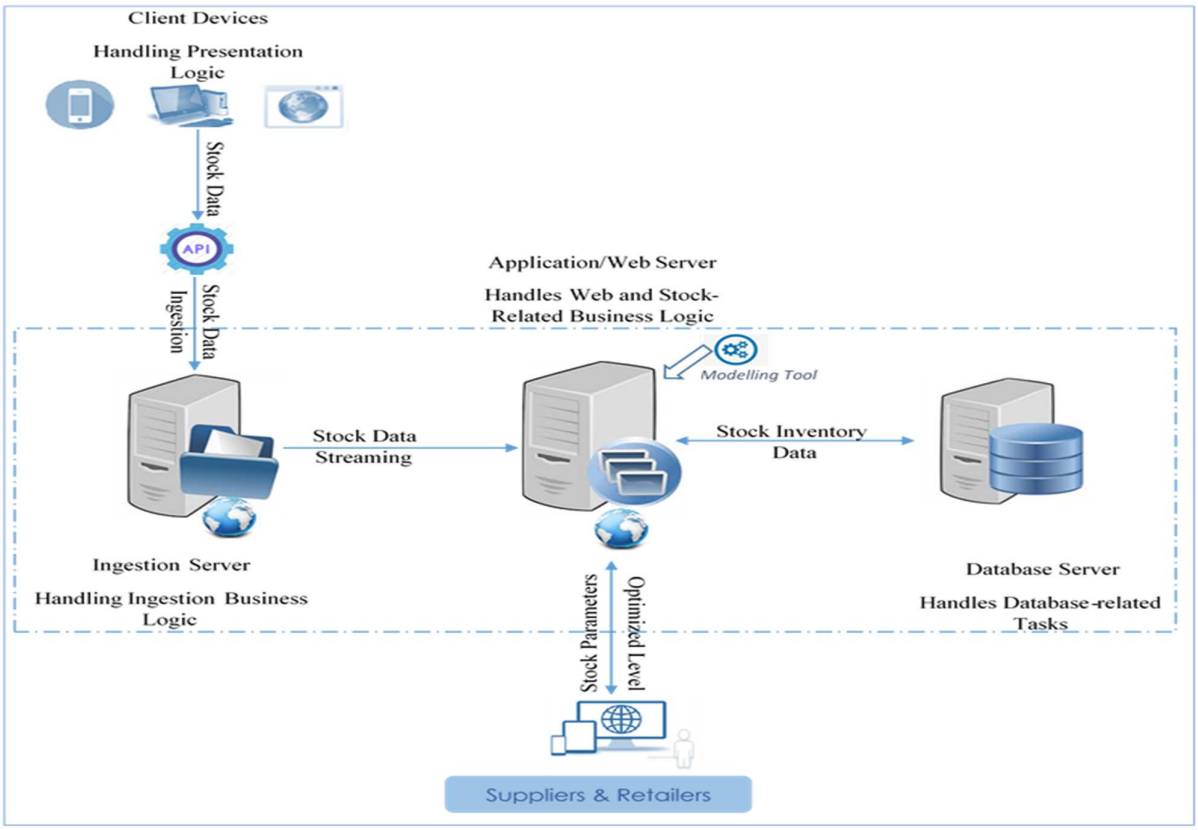


Figure 4.23: 4-Tier Client-Server Architecture

Stock inventory data from the retailer or supplier applications is transmitted to the ingestion server via an API. Data from the ingestion server is streamed to the database cluster. Suppliers and retailers can then query the system for optimized stock levels by supplying the relevant parameters. The modelling tool resides in the application server. It computes the optimized stock for a given client based on the client-specific data augmented by the data parameters supplied by the client.

System Design

System design is the process of creating the architecture, components, and interfaces for a system such that it satisfies the needs of the end user. It is the identification of the overall system architecture - consisting of a collection of physical processing components, hardware, software, people, and their communication - that will satisfy the system's essential needs (Roth et al., 2013).

4.1.10 Context Diagram

It shows the entire system in context with its environment, showing the overall business process as just one process (the system itself) and the data flows to and from external entities.

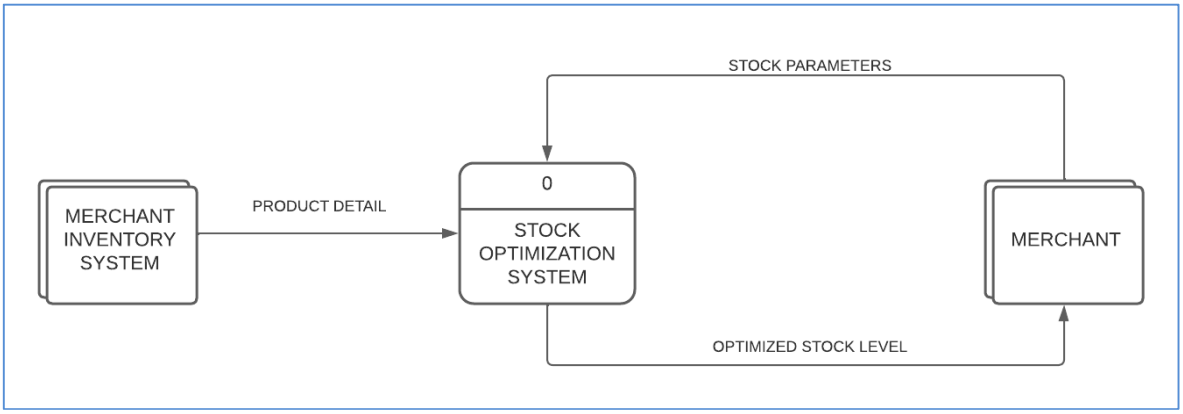


Figure 4.24: System Context Diagram

The Stock Optimization System has two external entities i.e. the Merchant Inventory System, and the Merchant who inputs the required parameter values seeking to compute the optimized stock. Merchant here refers to either a retailer or supplier.

4.1.11 Data Flow Diagram

The DFD represents the processes or activities that are performed by the system.

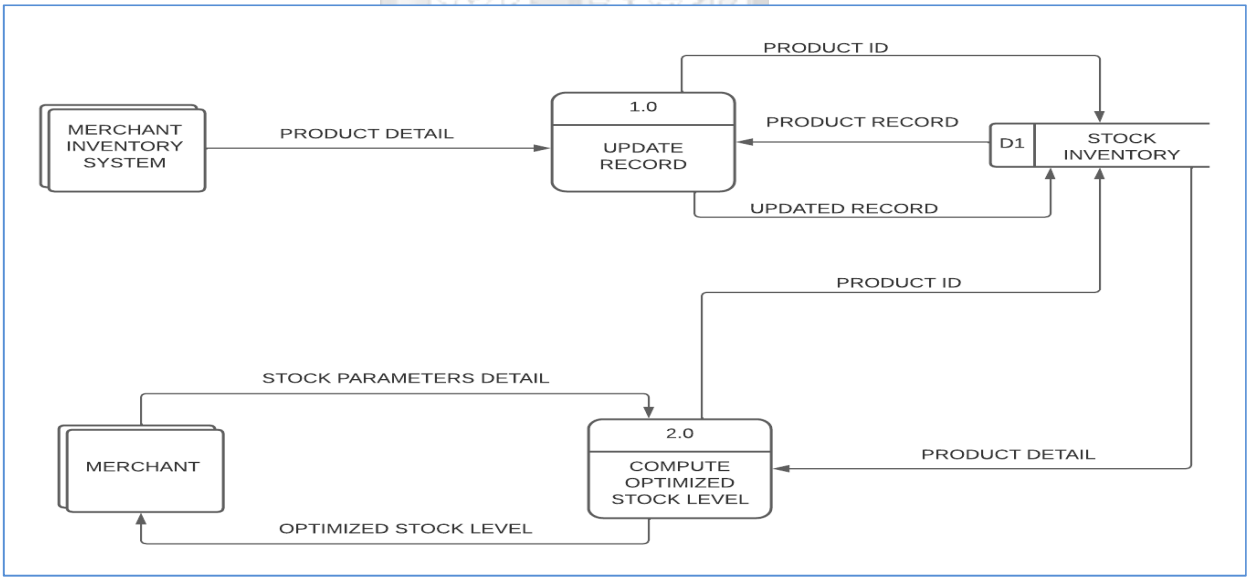


Figure 4.25: Level 0 DFD

The merchant system supplies inventory data to the system which is updated in the system data repository. Based on the data in the repository, the merchant supplies required parameters, and the system responds with the optimized stock level.

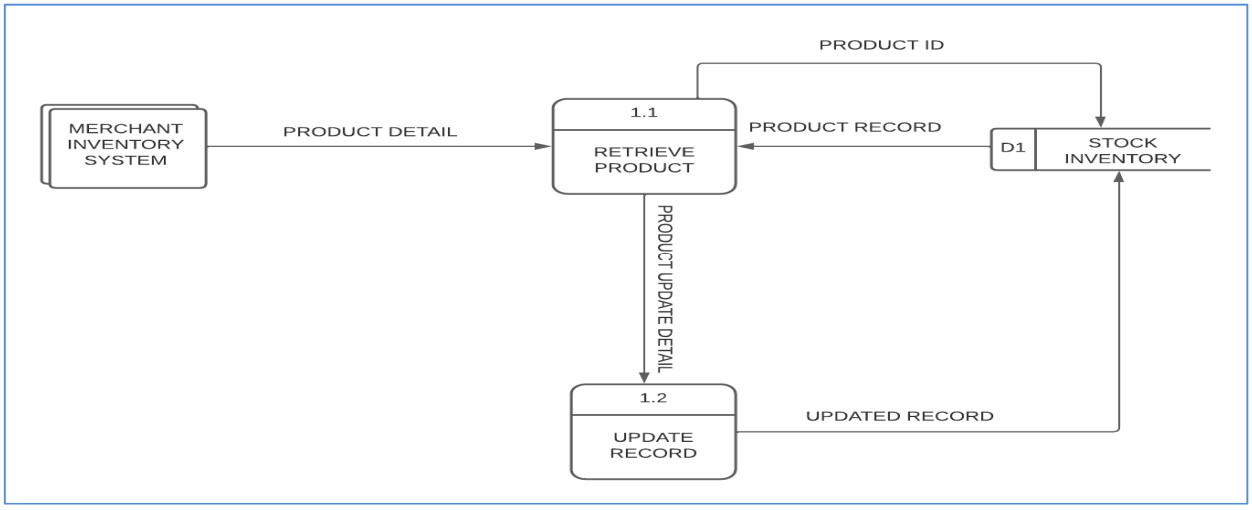


Figure 4.26: Level 1 DFD for Process 1.0

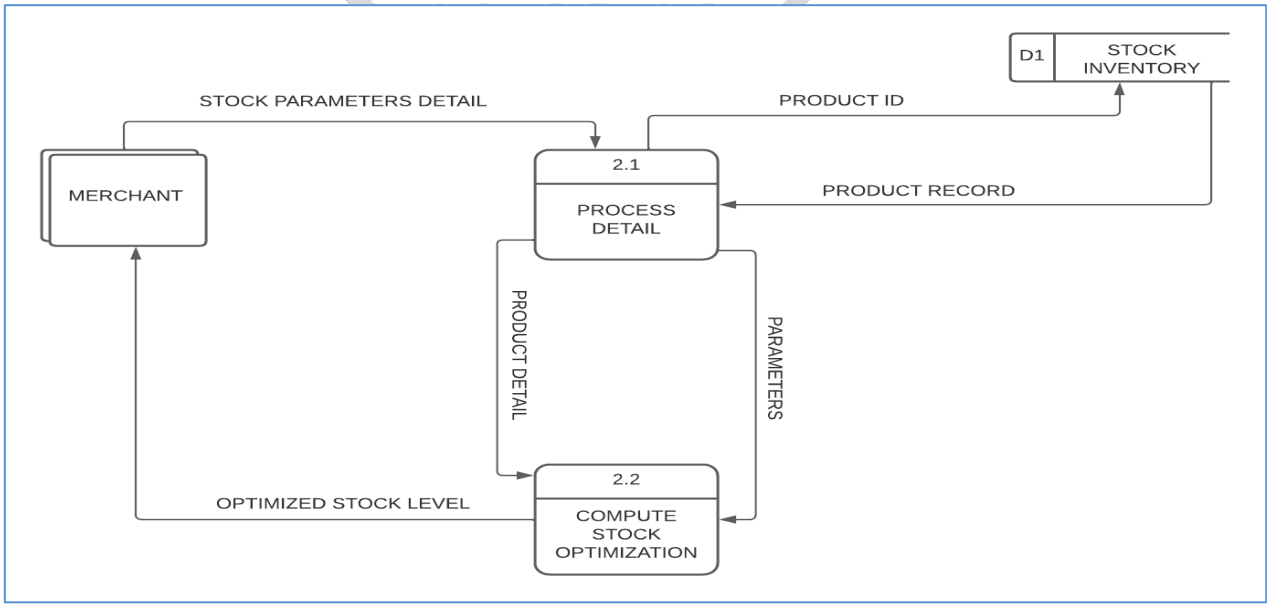


Figure 4.27: Level 1 DFD for Process 2.0

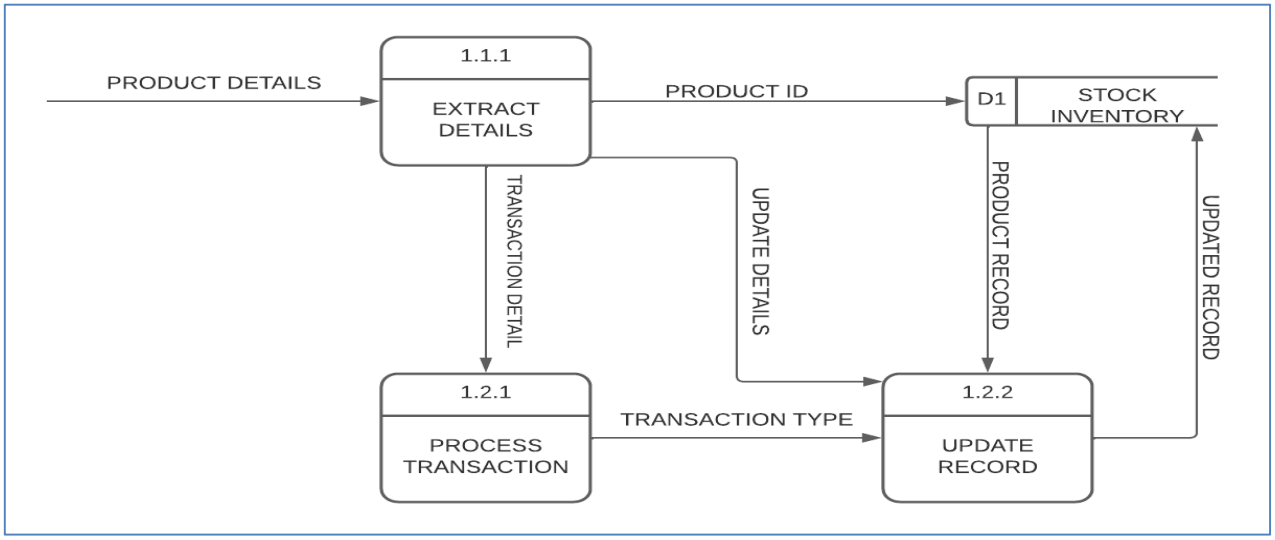


Figure 4.28: Level 2 DFD for Process 1.1

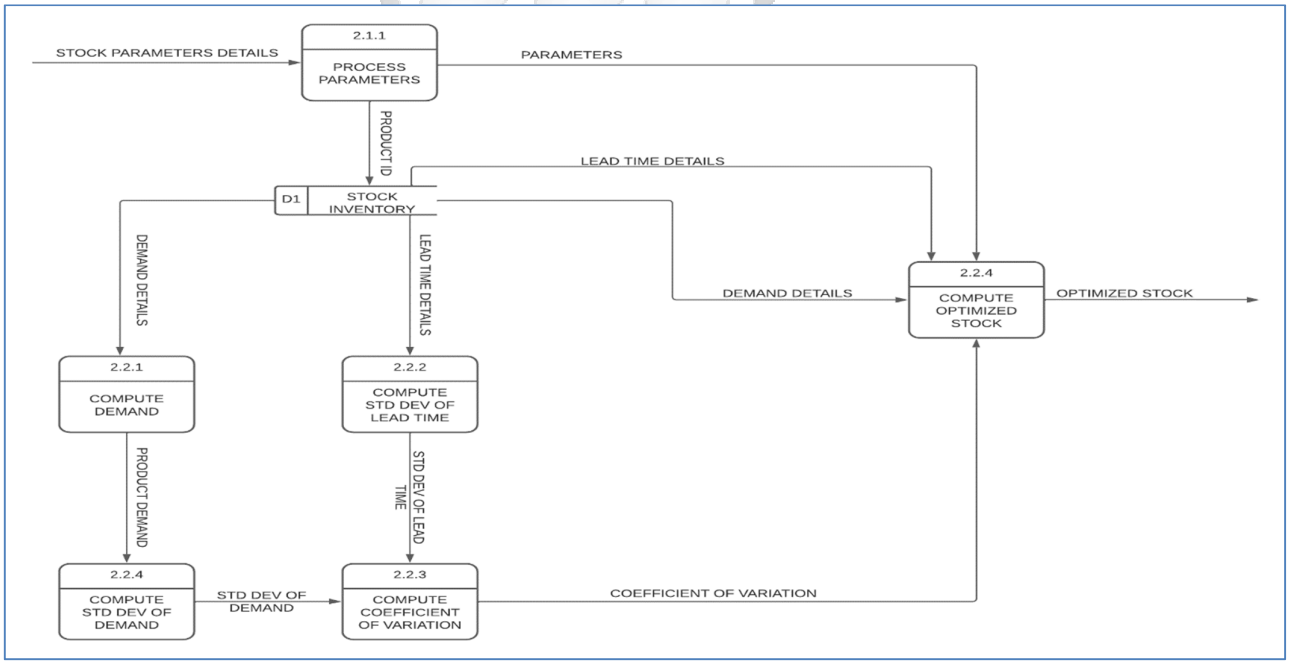


Figure 4.29: Level 2 DFD for Process 2.1

4.1.12 Entity Relationship Diagram

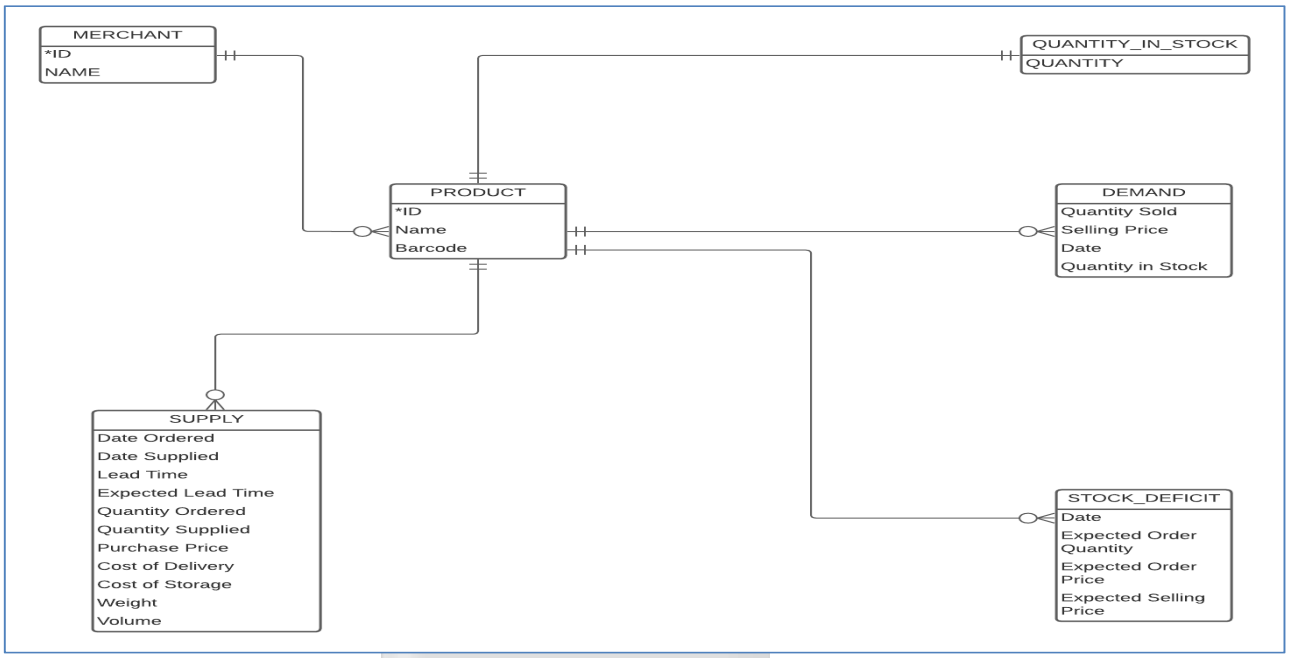


Figure 4.30: ERD

The Entity Relationship Diagram depicts the entities required for the functionality of the system. It represents the repository where data is streamed to from the ingestion layer. The merchant entity represents either a retailer or supplier, identified uniquely by an identification number and a name. The product entity stores product details used to uniquely identify it. The supply entity stores supply-related data of a particular product. The demand entity represents the sales data of the product. The stock deficit entity records any stockouts of a particular product experienced in the merchant premises.

4.1.13 Partial Data Model

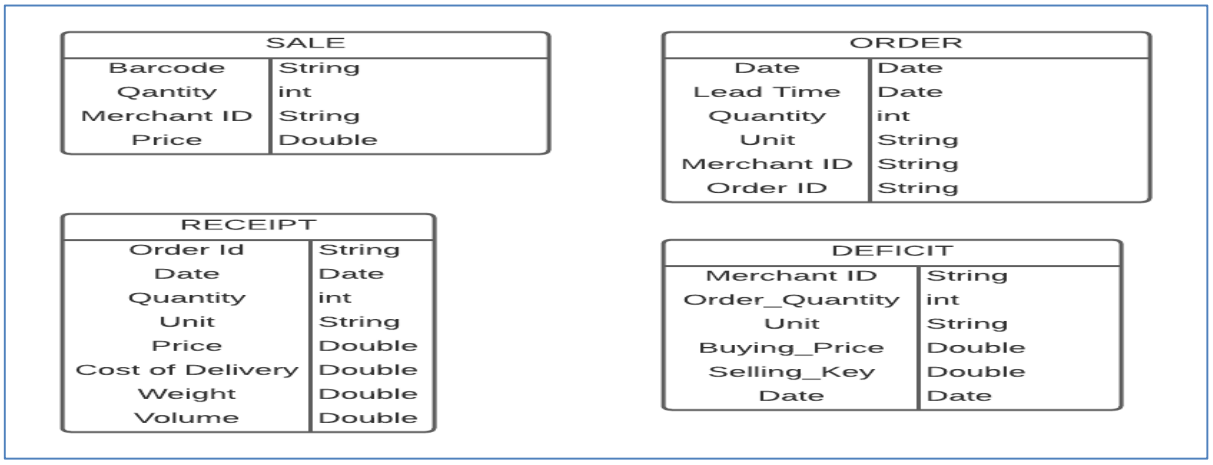


Figure 4.31: Partial Data Models

The partial data models represent the data received from the API from the merchant systems to the ingestion server, before being processed and streamed to the cluster database. The sale data model represents the sale transaction at the VMI system. The order represents a placed order for replenishment that allows to track lead time and differences in orders against quantity supplied. The receipt represents an event when the order is received. The deficit represents data regarding unavailability of a particular product either caused by stock-out or lack of supply.

Wire Frames

The user first registers to the system. The user may opt to sign-up using Google sign-in API. The user provides his/her name, email and password.

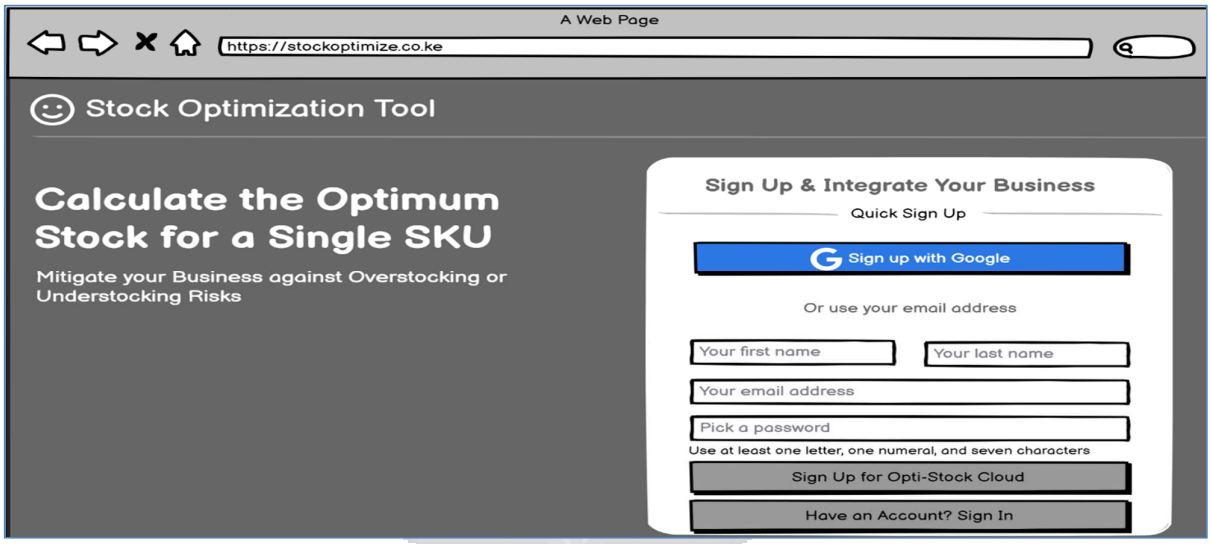


Figure 4.32: Web Sign-up Low-Level Mock-up

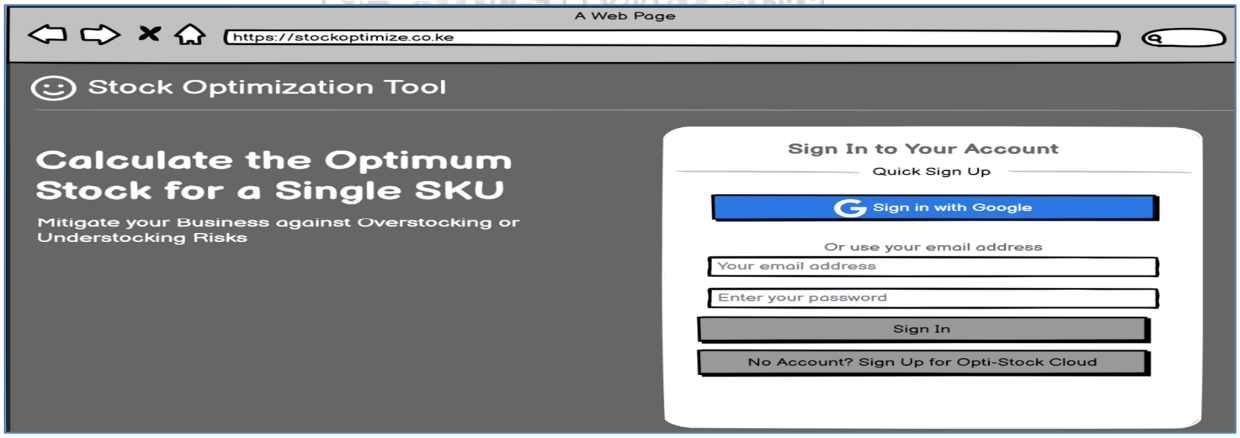


Figure 4.33: Web Sign in Low-Level Mock-up

After registration, the user can login through Google Sign-in API or directly into the system.

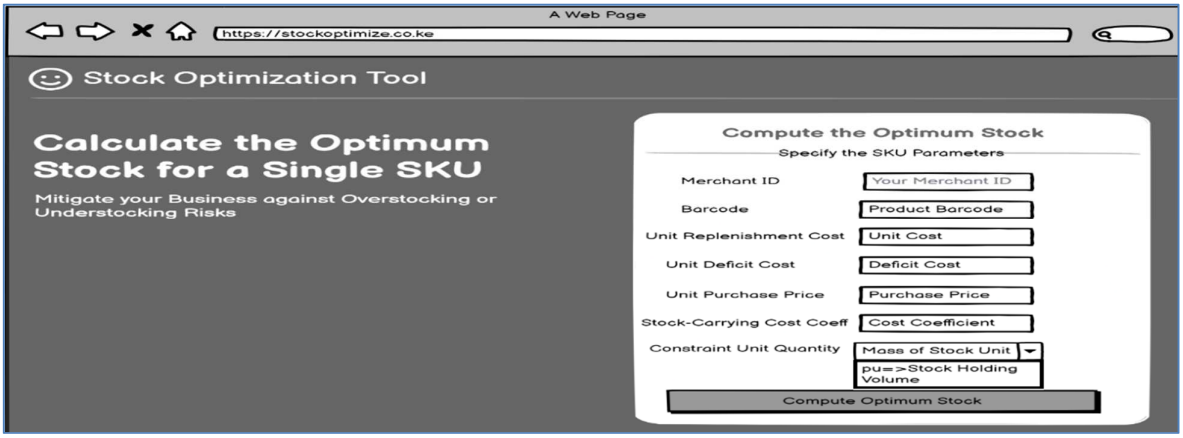


Figure 4.34: Web Optimization Parameters Interface Low-Fidelity Mockup

After login, the user specifies his/her merchant Id and the product barcode. This allows to select the respective product from the respective merchant in the database. The user then enters the relevant parameters for stock optimization.

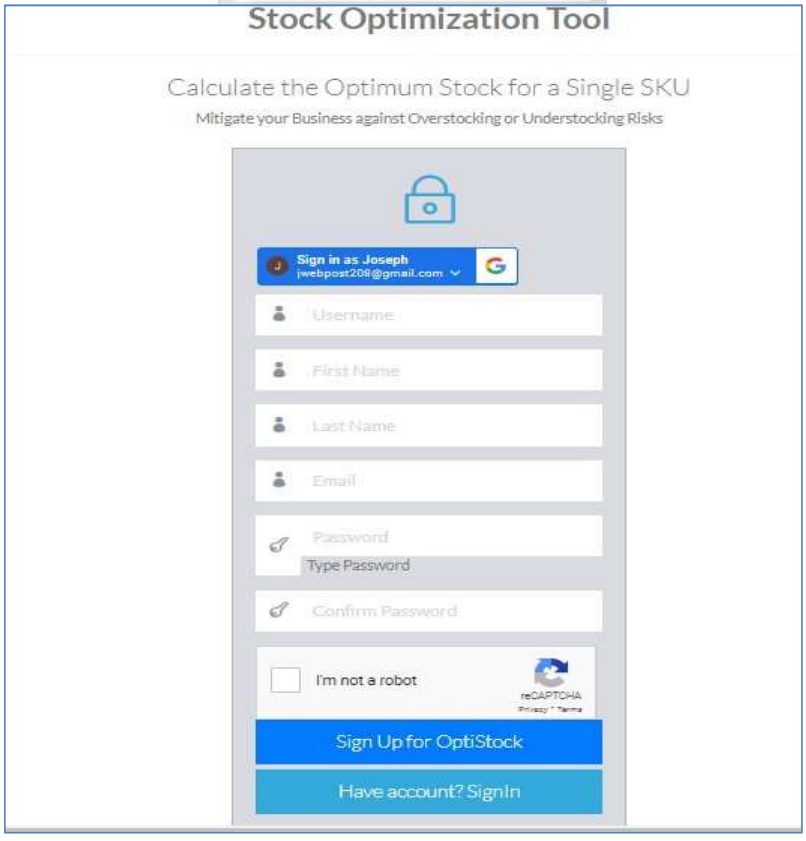


Figure 4.35: Web Sign-Up High-Fidelity Mockup

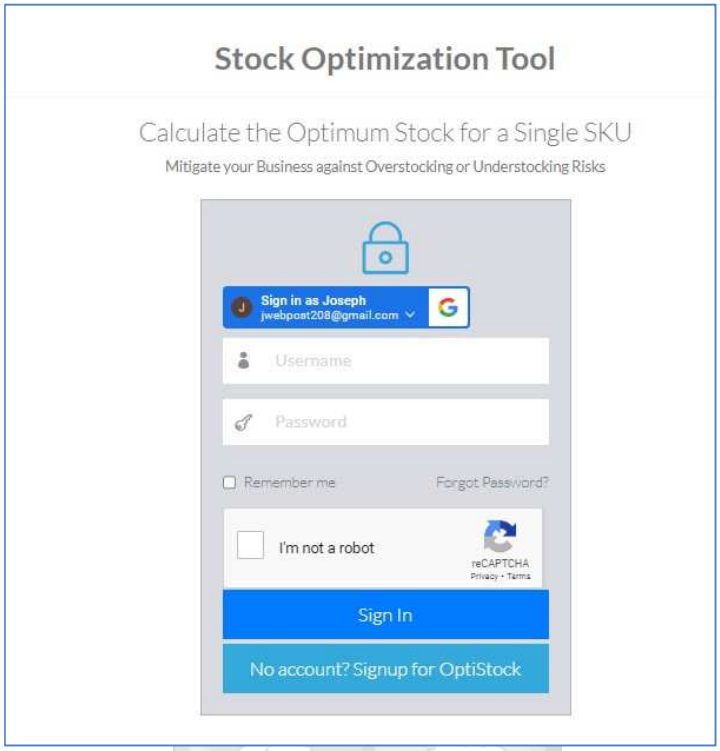


Figure 4.36: High Fidelity Web Login

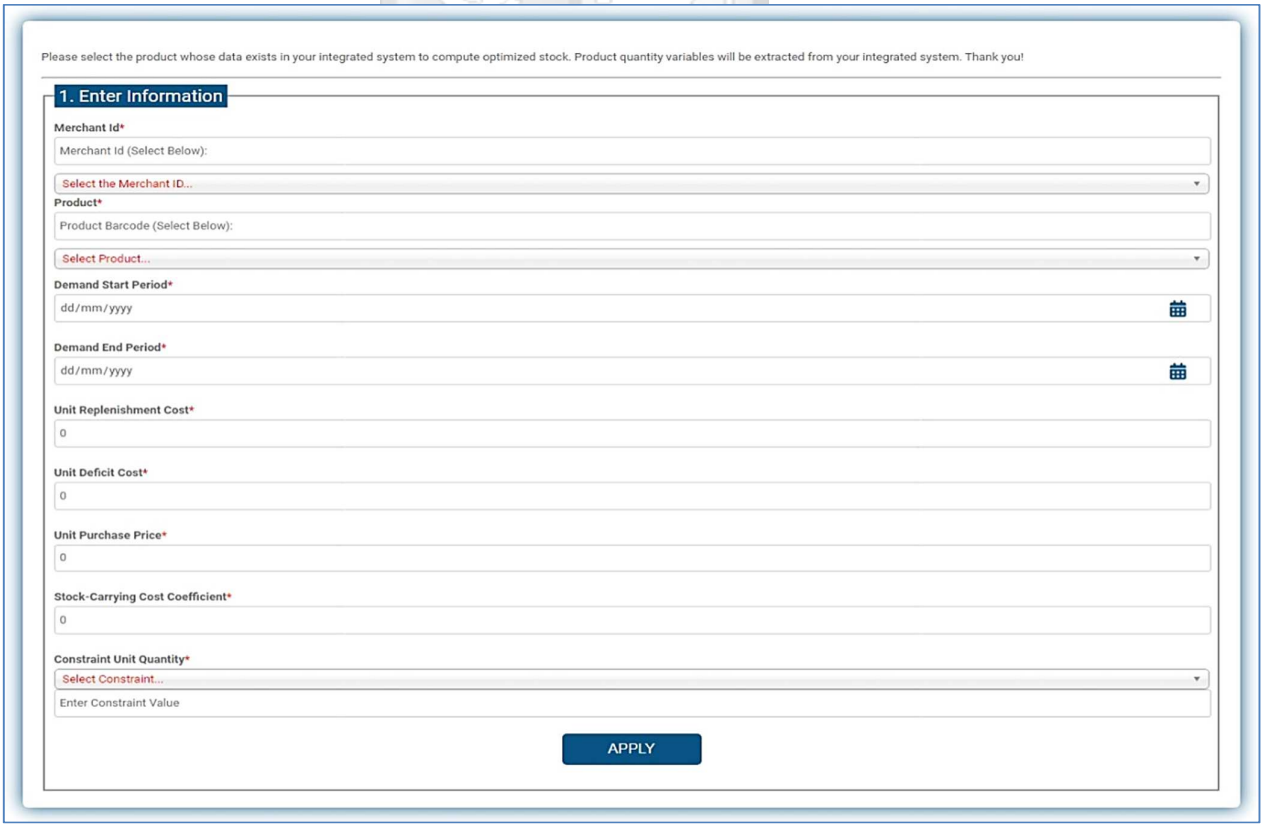


Figure 4.37: High Fidelity Web Stock Optimization Parameter Input Interface

Figure 4.38: Optimized Stock Output

After submitting the request, the user receives a response showing the stock related values computed by the system.

Chapter 5: System Implementation and Testing

5.1 Introduction

The main aim of the study was to develop a statistical model that allows retailers and suppliers to obtain optimized stock levels based on stock related parameters. The steps undertaken in the implementation of the system have been outlined. The historical data related to the selected product was used to develop the algorithms that are contained in the stock optimization model. The model was then validated by conducting comparisons of the model outcome to the historical data. Additionally, the model and system components testing and validation procedures have been described. Unit, integration and system tests were conducted on the system. Functional and non-Functional tests were conducted based on the aforementioned system requirements.

5.2 System Implementation

The system receives stock inventory data from retailer or supplier vending machines through an API. The data is transmitted to an ingestion server from which it is streamed to a cluster database server. The model then receives stock related parameters from the user through a web browser, augments it to the data in the repository related to the user, and computes the optimized stock. The system is made up of a front-end web application developed using HTML and Javascript, and the back-end sub-system based on Java Server Pages and Java-based modules.

5.2.1 Model Implementation

The statistical model adopted combines two algorithms; one for computing safety stock, and one for computing the replenishment order quantity. The safety stock algorithm was based on the demand-driven materials requirement planning (DDMRP) inventory replenishment approach. The replenishment order quantity algorithm was based on a Lagrange function augmented with constraints related to stock inventory structure. The function was used to calculate the optimal Lagrange multiplier value, which was then used to determine the components of the inventory structure. The function uses constraints, specifically, the service level, stock deficit costs and stock maintenance factors. The workflow of the model is illustrated in (**Figure 5.39**).

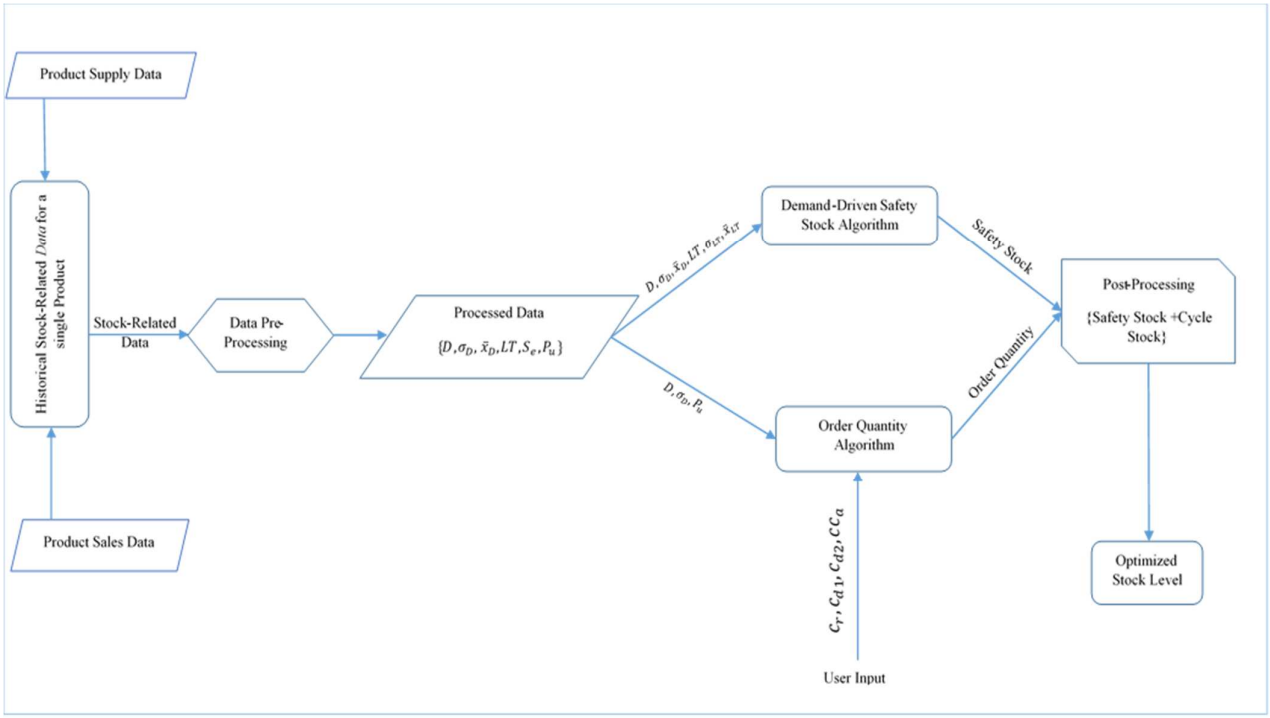


Figure 5.39: Model Workflow

The total demand was computed for the three-month period. The unit cost of replenishment (c_r) was computed from the product of average supply by a cost of one shilling. The cost related to stock deficit occurrence during the stock replenishment cycle (cd_1) was computed as the loss that would be incurred in case of a stock-out for a single day based on the average daily sales. The unit deficit cost (cd_2) for the item was taken as the profit margin of the product which would then be incurred as a loss. The standard deviation in the demand lead time σ_{DTL} follows equation 34. The purchase price (p_u) of the item was obtained from the historical data. The unit quantity of the assumed constrained (c) was set to one, assuming the constraint is the stock quantity in natural units that the retailer is able to stock in the retailer premises. The maximum admissible average stock level was based on the maximum supply quantity delivered within the three-month period. The values are as tabulated in Table 5.25.

Table 5.25: Optimized Stock Computation Parameters

D_{lt}	c_r	c_{d1}	c_{d2}	σ_{DTL}	p_u	cc_a	c	C
228	17	216	72	10	207	0.01	1	24

The initial Lagrange multiplier (λ_0) is set based on the formula in equation 35 so as to set a minimum acceptable value providing a positive value of the denominator. The first level of increment for the multiplier was set to 10, then incremented after each loop by a factor of 0.1. The number of iterations

were initially set to a hundred. With the initial value of lagrange multiplier, by iterating through the possible service levels, the first value of the safety coefficient ω, is computed by solving equation 36. Based on the pair (λ, ω), we solve for equation 36, such that Δq is minimum. The values corresponding to the minimum Δq are the optimum values used to compute the optimum order quantity, Q_{opt}.

$$\sigma_{DLT} = \sqrt{(\sigma_D^2 * LT + \sigma_D^2 * D^2} \ldots\ldots\ldots\ldots\ldots (34)$$

$$\lambda_0 = \frac{-p_u * cc_a}{c} \ldots\ldots\ldots\ldots\ldots (35)$$

$$\sqrt{\frac{2 * D_t * \{c_r + cd_1 * [1 - f(\omega)] + cd_2 * [F(\omega) - \omega * [1 - F(\omega)]] * \sigma_{DTL}\}}{p_u * cc_a + \lambda * c}}$$

$$= \frac{D_t[cd_1 * f(\omega) + cd_2\sigma_{DTL}(1 - F(\omega))]}{\sigma_{DTL} * (p_u * cc_a + \lambda * c)} \ldots\ldots\ldots\ldots\ldots (36)$$

where $F(\omega), f(\omega)$ are normal distribution functions of the safety coefficient; D_t is total demand.

The safety stock was computed based on the demand and the lead time, following equation 37. The optimum stock level is then computed based on equation 38.

$$SS = \left[D\left(1.02\sqrt{LT} + 1.15\right)\right]\left[1 + \sqrt{CV_D^2 + CV_{LT}^2 * LT}\right] \ldots\ldots\ldots\ldots\ldots (37)$$

$$Optimum\ Stock\ level = SS + Cycle\ Stock \ldots\ldots\ldots\ldots (38)$$

Where $cycle\ Stock = 0.5 * Q_{opt}$.

The model algorithm (Figure 5.40) was then coded in Java as a module. The pseudocode is illustrated below Figure 5.41. The Java module was then added to the web app. The Java.lang.Math class library was used for the computations.

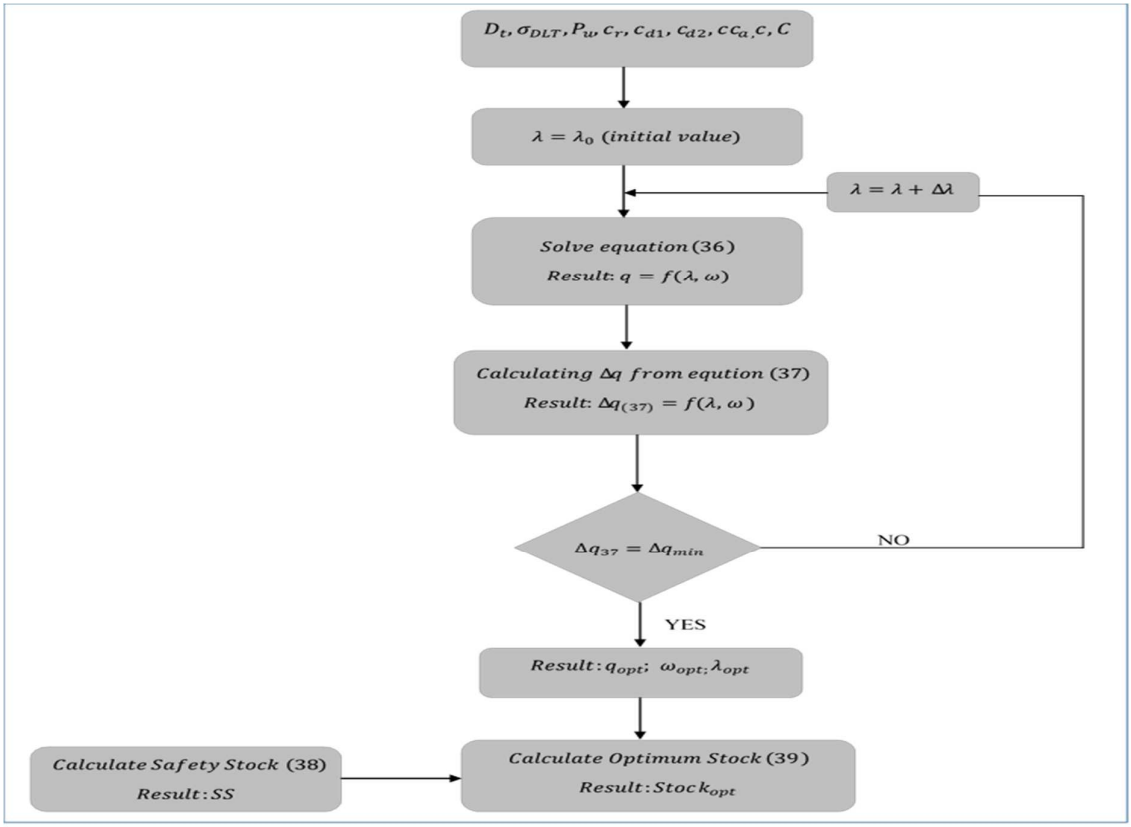


Figure 5.40: Model Algorithm

```
public class PseudoCode {

    val totalDemand, Leadtime, STDDEV_LT, STDDEV_D, lagrangeMultiplier,
    val lagrangeMultiplierIncrementValue, maxNoOfSteps, SafetyLevel;

    Array leadTimes = {};
    Array sales = {};
    Array supplies = {};

    void computeAverageLeadTime(){}
    void computeStdDevLeadTIme(){}
    void computeAverageDemand(){}
    void computeTotalDemand(){}
    void computeStdDevDemand(){}
    void computeStdDevDemandDuringLT(){}

    //initialze class constructor with required variables
    PseudoCode(double Cr; double Cd1, double Cd2, double Pu, DOuble CCa; double c, int C){
        compute lagrangeMultiplier;
    }
    //compute optimum order
    double optimumOrderQuantity(){
        while(currentStep<maxStep){
            increment lagrangeMultiplier by lagrangeMultiplierIncrementValue;

            for(range of service factor based on service levels) {
                compute Left Equation
                compute Right Equation
            }
            get the minimum difference of the equation solution;

            compute order quantity difference;
        }
        get the minimum order quantity difference;

        get corresponding lagrangeMultiplier;//optimum lagrangeMultiplier
        get corresponding SafetyCoefficient;//optimum safety coefficient
        calculate Service Level;//optimum service Level
        return computeOptimumOrderQuantity();
    }

    double computeSafetyStock(){};

    double computeOptimumStock(){};
}
```

Figure 5.41: Program Pseudo Code

5.2.2 Algorithm Testing

To validate the algorithm, the output values were compared to the historical data of the product within the given period. Firstly, we compare the computed optimum safety level to the service level computed from the historical data (Equation 39). Assuming all quantities were delivered in time, the service level of the product from hostorical data was computed. The computed service level from the model was then compared to the historical data service level. The average supply from the historical data was then compared to the computed order quantity from the model. To test the algorithm for reliability, the algorithm was tested with a different product and the output values compared to historical data.

$$\frac{the\ number\ of\ quantities\ delivered\ in\ time}{the\ total\ quantity\ of\ the\ demand} \ldots\ldots\ldots\ldots (39)$$

5.2.3 System Development

The Kappa big data deployment architecture was adopted for ingestion, streaming and repository (serving layer). Docker Desktop was used to deploy the system components. Apache Kafka was used as the ingestion layer. A single node Kafka with Zookeeper was configured as a docker file. Four topics, each with a single partition to ingest data from merchant remote applications, were created. Apache Spark was used as the streaming layer. YugabyteDB cluster database was used as the serving layer. Payara Glassfish server was used as the web server to host the streaming microservices as well as the java web application. A Java REST API was developed to receive data in JSON format. The streaming API was coded in Java and deployed as a microservice in the web server. The web application was developed as a Java web application and deployed in the Payara Glassfish Server. The web user interface was coded in Java Servlet Pages (JSP) with HTML and Javascript used for the frontend interface. The web application was then packaged as a war file and deployed.

5.3 Granularity Level System Testing

The system was tested based on the three granularity levels by conducting unit, integration and system tests. To accomplish a complete test plan on the system, an open-source POS system written in Java was downloaded and modified to send JSON data to the REST Java API (Appendix H through S).

5.3.1 Unit Test

Each module in the web app was tested by running the code in debug mode in Netbeans IDE, which allowed viewing the values at each stage of the code. Debugging break-points were set at lines that were considered significant to the functioning of the program (Figure 5.40). Thereafter, the program was run stepwise through the code, checking for output after each line. The registration module was tested for successful registration of users. The login module was tested for successful login. Wrong credentials were supplied to test if unauthorised user could access the system. The parameter input module was tested to ensure it captured and submitted supplied parameter values correctly. The model algorithm

module was tested for correct computation and output. The REST API was tested for ability to receive client data in JSON format and transmit it to the message broker successfully.

Black-box unit testing was done in the ingestion, streaming and service layer. Kafka broker was tested by creating a test topic through a command terminal (Figure 5.43), then running a consumer command on a separate terminal, and running a producer command on the previous terminal. The consumer terminal was checked if it was able to receive the message (Figure 5.44).

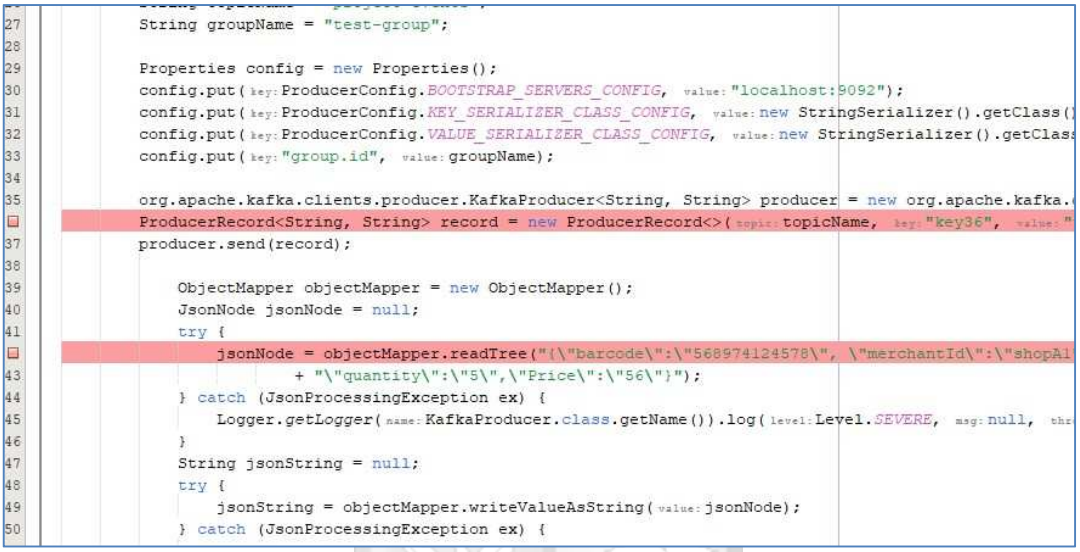


Figure 5.42: Code Snippet showing Debugging break points

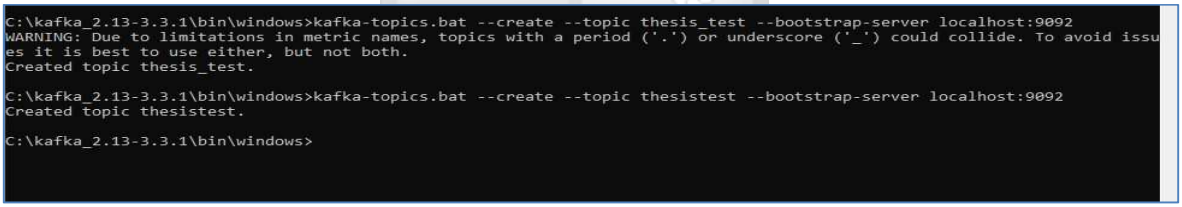


Figure 5.43: Kafka Topic Test

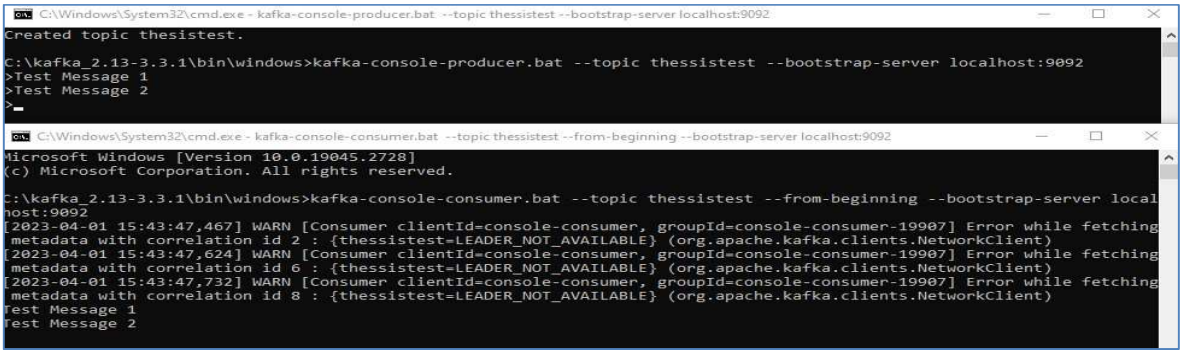


Figure 5.44: Kafka Producer and Consumer

Spark engine was tested by creating a Resilient Distributed Data (RDD) set and a Data Frame (DF). The RDD was created by parallelizing a list, which in turn was used to create a DF (Figure 5.45).

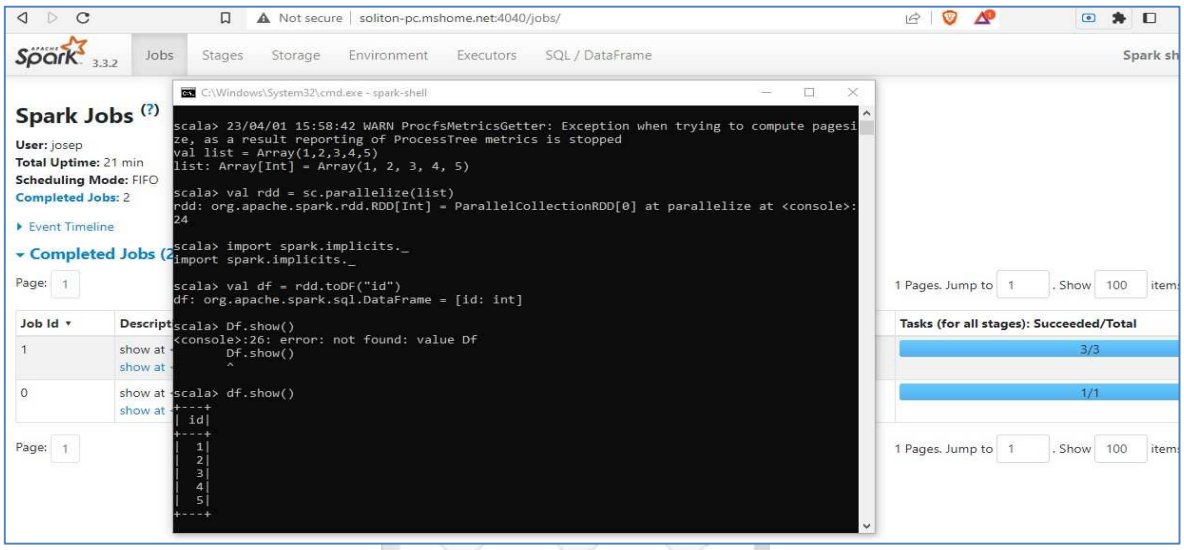


Figure 5.45: Spark Test

The Yugabyte clustered database was tested by creating the database for the system and posting test data. The data was then queried to check if it was successfully captured in the database.

5.3.2 Integration Testing

The POS system was deployed, and a transaction done on the application. The transaction details were recorded manually for comparison with the captured data in the database. The same data was queried from the database, and found to be similar to the transaction details recorded. This implied that the subsystems were successfully integrated since the data produced by the client was successfully stored in the message broker, and streamed through spark to the database. Further, through a kafka consumer console, the data was checked to be available in the broker topic.

5.3.3 System Testing

A number of transactions on a particular product were done on the POS system, and the transaction details recorded. Afterwards, through the user interface, a test merchant selected, the product transacted in the POS system was selected, test stock parameters were input, and the request submitted. The user was able to login to the system. The system successfully computed the optimized stock and gave the expected response through the user interface.

5.3.4 Summary of the Granularity Level System Testing

Test	Expected Outcome	Status
Unit Test	-Each module in the web app had no bugs and gave the correct output -The ingestion layer received data from client successfully and correctly -The streaming layer successfully streamed data from broker to database -The serving layer (database) received data successfully from data stream -The serving layer successfully responded to queries from the web app	Pass
Integration Test	-The subsystems successfully communicate with each other -Data from client successfully transitions from the client, to the ingestion layer, through the streaming layer and finally to the serving layer	Pass
System Test	-A user is able to interact and query the system for the optimized stock level -The system gives the user the correct response	Pass

5.4 System Validation

The system was validated based on the functional and non-functional requirements to ascertain whether all desired functionality was accomplished. The functional test cases are illustrated in the **Table 5.26**, showing how the developed system aligned with the system requirements.

Table 5.26: System Validation Test Cases

Reference Number	Functional Test Case	Significance	Results
FRQ1	Does the system capture the required stock parameters from the client correctly?	High	The system captured the product, quantity sold, lead time, purchase price, and constraints into the system for a single product.
FRQ2	Is the model able to process the client product data to compute the optimized stock level?	High	The model was able to process the data and compute demand, covariance and standard deviation of demand and lead time, incorporate the constraint parameters into the model and compute the optimum stock based on the model.
FRQ3	Are the suppliers and retailers able to obtain optimized stock quantity information for a single product?	High	The system was able to respond to the user query for optimized stock level for a single product.

Table 5.27: System Validation Test Cases (Continued)

Reference Number	Functional Test Case	Significance	Results
FRQ4	Is the system able to retain the client product data for future reference?	Medium	The system stores data in the ingestion layer and in the clustered database for as long as required.
FRQ5	Does the system have capability to keep real time inventory levels at the client premises?	High	The system, through the ingestion layer coupled with the REST API, is able to stream stock related data from client in real time
NFRQ1	Does the system provide the user with a user-centric interface to interact with the system?	High	The system, through the web application, provides the user with a simple, convenient interface to register, login, query the system for optimized stock and respond to the user with the optimized stock level for a product
NFRQ2	Is the user, when querying for optimized stock, able to enter the required stock parameters without technical assistance?	Medium	The system provides html contextual hints on each of the input fields. Further tests required to check if users enter correct parameters.
NFRQ3	Is the system interoperable with existing client systems?	High	The system has an API that is able to receive data in the common JSON format. Moreover, the system only requires the client system to transmit data to the REST API in JSON format. The API is compiled as an executable Java Jar file with all required dependencies.
NFRQ4	Can the system run without failure during normal working hours?	High	The system was allowed to run for 24 hours with simulated stream data successfully.
NFRQ5	Does the system ingest data from the client correctly?	High	The ingested data was checked against the data from the POS system and found to be accurately captured

Table 5.28: System Validation Test Cases (Continued)

Reference Number	Functional Test Case	Significance	Results
NFRQ6	Are the users able to interact with the system through their mobile or web platforms?	Medium	The system user interface is provided through a front-end web interface based on HTML and Javascript languages which are cross-platform
NFRQ7	Is the REST API cross-platform?	Medium	The API is packaged as an executable Java Jar File
NFRQ8	Can the users access the system from various browsers?	Low	The web interface was tested on Google Chrome, Brave, Safari, Firefox and Microsoft Edge
NFRQ9	Is the system able to update product details from streamed data?	High	Streamed data of a single product checked for update after a transaction in the POS successfully.
NFRQ10	Can the system support simultaneous users?	Medium	The Kafka Message Broker supports simultaneous ingestion of data from various sources. The web interface can be deployed as a microservice and scaled respectively.
NFRQ11	Are only authorized users allowed into the system?	High	System login testes with fake credentials without success. System further configured with two-step authentication. Users can also login using their Google account.
NFRQ12	Does the system ensure data integrity?	High	Only logged in users are allowed. Users can only modify their own data. The Message broker offers the required redundancy to the clustered database.
NFRQ13	Has the system demarcated data pertaining to each client?	High	Each product is uniquely related to its owner through the SQL clustered database

Chapter 6: Discussion

Introduction

This section describes the outcome of the developed solution with respect to the study research objectives and research questions. The purpose of this study was to develop a model for computing the optimized stock levels among retailers and suppliers. The results of the algorithms have been presented and reviewed.

Stock Optimization Model Based on Stock-Related Historical Data

The model parameters informed the quality of historical data to be collected. The model parameters were then computed from the data collected. The model developed computes the optimum order quantity based on a Lagrange function, as well as a DDMRP safety stock. Based on the model output, the optimized stock level is computed.

4.1.14 Model Results

The computed values for the selected product A20 are as tabulated below (Table 6.29). The product is a high moving product, therefore, the computed service level at 95.2% is a positive indicator that the retailer will not hit a stock-out during the subsequent replenishment cycle. Consequently, this lowers the probability of losing sales. Based on the optimum service level computed by the model, the probability of running out of stock is 0.048. On a probability scale of 0.01 to 0.99, the computed probability of stock-out is very low. Additionally, the risk of stock-out is mitigated through the computed safety stock. Moreover, with a safety stock of 14 units, the retailer is able to cater for varying vendor delivery periods, as well as meet the changing consumer demands. Further, from the data collected, it is observed that the retailer is able to sell 24 units weekly. Therefore, the safety stock bears no risk of product expiration. Also, the optimum order quantity lowers the delivery cost for the retailer in each replenishment cycle. To mitigate overstocking, the retailer can ensure that he maintains stock levels that relate to demand patterns. This can be achieved by ensuring s/he makes an order every time the cycle stock approaches depletion. Similarly, the retailer can further inform decision making in the reorder placements based on the average lead time, such that the delivery is made every 6 days.

Table 6.29: Product A20 Model Results

Optimum Order Quantity	Optimum Safety Stock	Cycle Stock	Optimum Stock	Optimum Service Level	Average Lead Time
15 units	14 units	7 units	21 units	95.20%	6 days

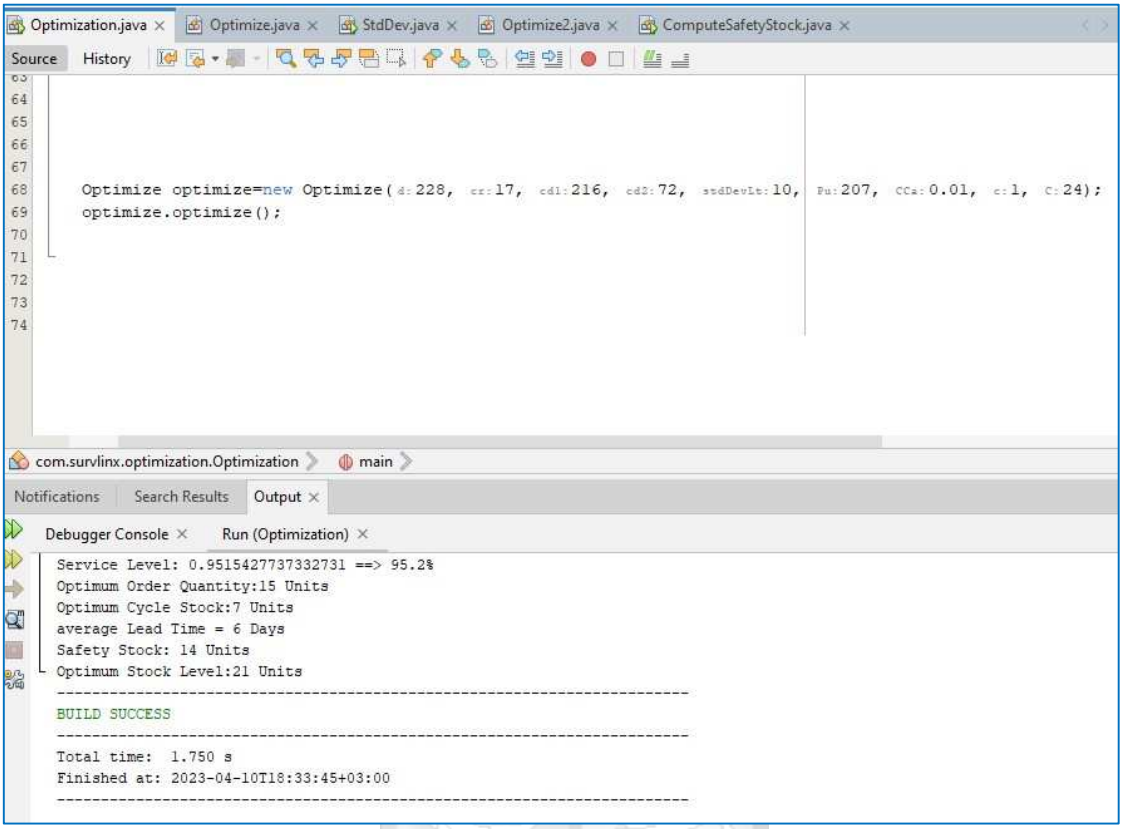


Figure 6.46: Module Results

Chapter 7: Conclusion and Recommendation

7.1 Conclusion

When commodity stock-related data is analysed properly, retailers and supplier can optimize their stock levels. Further, retailers and suppliers should adopt more efficient approaches in managing stock inventory at their premises. These approaches should meet the consumer demand as well as lower the cost of stock storage and stock delivery. The classical stockpile management technique aims at maintaining the ideal stock level from the lowest total expenditures. This implies that stock level optimization is viewed from the perspective of expenses. Accordingly, the suppliers strive to minimize delivery costs while maximizing the quantity delivered.

Lack of optimization of stock levels can be attributed to lack of efficient inventory management control strategies. Consequently, retailers and suppliers that lack such strategies suffer from inadequate data to optimize stock or lack efficient systems in place to collate such data. As a result, the crucial information needed to optimize stock is not shared between the retailers and the supplier. These limitations can be mitigated through efficient retailer-supplier integration support systems, adoption of technology that incorporates efficient stock optimization models, inventory control systems that are able to capture the required stock optimization parameter data and platforms that allow them to compute the stock optimization levels. The technologies adopted should enhance sharing of stock-optimization data, thus allowing for seamless collaboration to reduce waste. Additionally, retailers and suppliers should purpose to determine the optimal policies of inventory control to support maintaining the balance between satisfying the demand and optimum stock in the premises.

The adoption of technology through stock management systems is significant in chain stores, many of which have adopted the Vendor Managed Inventory systems. Although a significant number of retailers have not adopted technology in managing stock levels, there is a positive trend towards installation of Point-of-Service systems. However, managing of stock at optimum levels requires the retailers and suppliers to apply efficient stock optimization strategies. Just-in-Time approach has mainly been adopted by retailers who lack inventory systems, where orders are made after stock nears or is depleted. With installed VMI, suppliers are able to apply the EOQ model. However, significant number of retailers still prefer the ABC approach to maximize on profits. Various architectures are available for integrating retailer to supplier systems that would allow information sharing in real time. However, no single architecture achieves all the five requirements (volume, velocity, variety, variability, and veracity) of Big Data. Consequently, the challenge of selecting an appropriate design for a certain use case scenario would depend on the system requirements and the corresponding satisfaction of the requirements of each architecture.

The quality of stock data plays a key role when optimizing stock levels. Each individual stock optimization approach requires certain parameters. Therefore, retailers and suppliers should structure their stock inventory systems to align with the approach adopted, such that data required for optimization is available. It is possible to optimize stock levels by combining various models of stock optimization. This study combined two algorithms, one based on a Lagrange function and the other on a demand driven safety stock algorithm. The model yielded an optimum service level of 95%, implying that retailers and suppliers can mitigate against the risk of stock-out or understocking. The model further yielded an optimized order quantity that was less or close to the average supply level of the given product. The optimized level mitigates against frequent ordering and delivery costs that would otherwise be incurred by the retailers and suppliers. The computed safety stock mitigates against the risk of stock-out by acting as buffer stock in case of sudden increase in consumer demand. In addition, the model better informs decision making in placing orders by informing the retailer or supplier when stock levels fall below or nears depletion based on the cycle stock. Similarly, based on the average lead time, they are better informed on the suitable intervals of placing orders.

Based on the model, users are able to compute the optimized order quantity, the safety stock, the cycle stock, the average lead time and the optimum stock level. The model was also tested based on another high moving product, yielding a service level of 96%, implying that it can assist in stock optimization of various commodity products if supplied with the required parameter values.

7.2 Recommendations

Based on this study, the following are the recommendations for the model to work more effectively: Reduce the number of iterations that would reduce the time taken for the Lagrange function to converge by using data analysis (through machine learning) on historical sales data to determine the initial current values of service level and lagrange multiplier; To test the model with more data sets for improved results. The current model was only limited to a three-month period single data set from one retail outlet; To test the model for better results with more quality data such as daily demand, actual stock-out days or actual supply cost; The retailer or supplier systems should be improved to capture the relevant data needed in the parameters of the model.

7.3 Future Work

The current model was based on historical demand and supply data. Future studies can improve the model by: Using a predictive model to predict demand and supply; Investigate the significance of the parameters on the optimized stock level using data spread over a longer period such as three years; Using a predictive model to predict values of the initial service level and Lagrange Multiplier.

Bibliography

Arasa R. & Achuora J. (2020), Strategic Inventory Management Practices and the Performance of Supermarkets in Nairobi County, Kenya. *European Journal of Business and Management Research,* Vol. 5, No. 2, April 2020

Carrim I., Agigi A., Nieman W., & Mocke K. (2020), The role of buyer-supplier relationships in enhancing sustainable supply chain management in a logistics services context. *Journal of Contemporary Management,* Volume 17 Issue 1 2020 Pages 150-182

Chambers, D. & Lacey, N. (2011). Modern Corporate Finance, Sixth Edition. Reading: Hayden McNeil Publishing.

Choudhury M. & Mahata C. (2021), Sustainable Integrated and Pricing Decisions for Two-Echelon Supplier–Retailer Supply Chain of Growing Items. *RAIRO Operations Research,* RAIRO-Oper. Res. 55 (2021) 3171–3195

Craig N., DeHoratius N. & Raman A. (2016), The Impact of Supplier Inventory Service Level on Retailer Demand. *Harvard Business School,* Working Paper 11-034

Creswell J. & Poth C. (2018), Qualitative Inquiry and Research Design *Choosing Among Five Approaches.* Reading: SAGE Publications

Davoudian A. & Liu M. (2020). Big Data Systems: A Software Engineering Perspective. ACM Computing Surveys Vol. 53, No. 5, Article 110 (September 2020). https://doi.org/10.1145/3408314

Galbraith, J.R. (1973), Designing Complex Organizations. Reading, MA: Addison- Wesley

Hou R., Li W., Lin X. & Zhao Y. (2022), Impact of Quality Decisions On Information Sharing with Supplier Encroachment. *RAIRO Operations Research*, RAIRO-Oper. Res. 56 (2022) 145–164

Hoswari S., Gozali L., Marie I. & Sukania I. (2020), Comparison Study about Inventory Control System from Some Papers in Indonesian Case Study. IOP Conf. Series: Materials Science and Engineering 852 (2020) 012110

Huang S., Guan X. & Chen Y. (2017), Retailer information sharing with supplier encroachment

Jafarpour H., Desai R. & Guy D. (2019), KSQL: Streaming SQL Engine for Apache Kafka. *Industry and Applications Paper, Open Proceedings*

Kerin R., Berkwitz E., Hartley S. & Rudelius W. (2002), Marketing, Seventh Edition. Reading: McGraw-Hill Irwin

Kmetz, John L. (2021), Everytime, Baby: Systems, Information, Complexity, and Technology

Korponai J., Toth A. & Illes B. (2017), Effect of the Safety Stock on the Probability of Occurrence of the Stock Shortage. *7th International Conference on Engineering, Project, and Production Management.* Procedia Engineering 182 335 – 341

Kothari C. R. (2004), Research Methodology-Methods and Techniques. Reading: New Age Int. Ltd Publishers

Krzyżaniak S. (2022), Optimisation of the stock structure of a single stock item taking into account stock quantity constraints, using a lagrange multiplier. LogForum 18 (2), 261-269, http://doi.org/10.17270/J.LOG.2021.730

Krzyżaniak, S., (2017). An Attempt Towards a Model Approach to Choosing a Stock Replenishment System Under Conditions of Independent Demand. Transport Economics and Logistics, 68, pp.39-48.

Krzyżaniak St., (2017). Influence of delivery quantity on service level in the stock replenishment system based on reorder level for irregular distributions of demand. *LogForum* 13 (4), 415-427

Kumar R. (2005) Research Methodology, A Step-By-Step Guide for Beginners. Second Edition. Reading: Sage Publications

Lee C. & Rim S. (2019), A Mathematical Safety Stock Model for DDMRP Inventory Replenishment. *Hindawi Mathematical Problems in Engineering,* Volume 2019, Article ID 6496309

Levy M., Weitz B. & Beitelspacher L. (2012), Retailing Management, Eighth Edition. Reading: McGraw Hill Irwin

Logility Handbook (2022): The advanced Inventory Optimization Handbook: Application of Machine Learning to Boost MEIO

Lukitosari V. & Subriadi A. (2020), Inventory models for short life cycle clothing products use a logistic growth model. *Journal of Physics: Conference Series.* 1490 (2020) 012060

Maina, Z. K., & Ngugi, P. K. (2019), Factors affecting stock levels in retail supermarket outlets in Nairobi County, Kenya. The *Strategic Journal of Business & Change Management*, 6 (4), 869 – 884

Makori W., Magutu P., Omai K. and Akello E. (2016), The Relationship Between Real-Time Information Processing and Supply Chain Optimization Among Supermarkets in Nairobi, Kenya. *International Journal of Economics, Commerce and Management UK.* Vol. IV, Issue 2, February 2016 ISSN 2348 0386

Marz N. & Warren J. (2015), Big Data: Principles and Best Practices of Scalable Real-time Data Systems

Muhoza P., Koffi A., Anglewicz P., Gichangi P, Guiella G., OlaOlorun F., Omoluabi E., Sodani P., Thiongo M., Akilimali P., Tsui A. and Radloff S. (2021), Modern contraceptive availability and stock-outs: a multi-country analysis of trends in supply and consumption. *Health Policy and Planning*, 36, 2021, 273–287 doi: 10.1093/heapol/czaa197

NetStock (2021), Netstock brochure US edition

Ndwiga, J., M. & Kiarie, D., M. (2017). Influence of Inventory Control Techniques on Performance of Retail Chain Stores in Nairobi City County, Kenya. International Journal of Human Resources and Procurement. Vol. 6 (5) PP 163 – 192.

Nzioka J. & Were S. (2017), Effect of Inventory Management On Performance of the Education Sector In Kenya. *International Journal of Novel Research in Education and Learning*, Vol. 4, Issue 6, pp: (48-59), ISSN 2394-9686

Obayi, R., Koh, S.C., Oglethorpe, D. et al. (1 more author) (2017) Improving Retail Supply Flexibility using Buyer-Supplier Relational Capabilities. *International Journal of Operations and Production Management*, 37 (3). pp. 343-362. ISSN 0144-3577

Olang'o A., (2018). Influence of Inventory Management Practices on Stock Out of Human Immunodeficiency Virus Rapid Test Kits in Health Care Facilities in Kenya. African Journal of Business and Industry, 1(4), 187-198.

Ochelle C., Muturi W. & Atambo W. (2017), Effect of Inventory Control Methods On the Performance of Procurement Function in Sugar Manufacturing Firms in Western Kenya. *International Journal of Social Sciences and Information Technology.* Vol III Issue II, May 2017 ISSN 2412-0294

Olow R., Abdi N., Malicha F., Mohamed M., Mwangangi M., Magutu P. & Mutunga J., Stock Management Practices and Supply Chain Performance of Pharmaceutical Companies in Nairobi, Kenya. Noble International Journal of Business and Management Research. ISSN(e): 2520-4521 ISSN(p): 2522- 606 Vol. 04, No. 04, pp: 26-37, 2020. Accessed via http://napublisher.org/?ic=journals&id=2

Ongbali O., Afolalu A., Fayomi O. & Oladipupo S. (2019), Inventory replenishment in multi-stage production setting under stochastic demand: a review. *Journal of Physics: Conference Series.* 1378 032072

Ooms G., Kibira D., Reed T., Ham H., Mantel-Teeuwisse A. & Buckland-Merret G. (2020), Access to Sexual and Reproductive Health Commodities in East and Southern Africa: A Cross-Country Comparison of Availability, Affordability and Stock-Outs in Kenya, Tanzania, Uganda and Zambia. *BMC Public Health.* (2020) 20:1053. Accessed through https://doi.org/10.1186/s12889-020-09155-w

Ooms GI, van Oirschot J, Okemo D, Waldmann B, Erulu E, Mantel-Teeuwisse AK, et al. (2021) Availability, affordability and stock-outs of commodities for the treatment of snakebite in Kenya. PLoS Negl Trop Dis 15(8): e0009702. https://doi.org/10.1371/journal.pntd.0009702

Ovidiu-Cristian Marcu, Alexandru Costan, Gabriel Antoniu, María Pérez-Hernández, Radu Tudoran, et al., Storage and Ingestion Systems in Support of Stream Processing: A Survey. [Technical Report] RT-0501, INRIA Rennes - Bretagne Atlantique and University of Rennes 1, France. 2018, pp.1-33. ffhal-01939280v2

Ozsu, M. T., Valduriez, P., 2011. Principles of Distributed Database Systems, 1190 Third Edition. Reading: Springer.

Philipp Z. & Dominik R. (2017), StreamConnect: Ingesting Historic and Real-Time Data into Unified Streaming Architectures. FZI Research Center for Information Technology, Germany

Premkumar, G., Ramamurthy, K., & Saunders, C. S. (2005). Information processing view of organizations: An exploratory examination of fit in the context of interorganizational relationships. *Journal of Management Information Systems*, 22(1), 257-294)

Radasanu A. (2016), Inventory Management, Service Level and Safety Stock. *Journal of Public Administration, Finance and Law*. Issue 9/2016

Richardson C. (2019), Microservices Patterns with Examples in Java

Roth R., Dennis A. & Wixom B. (2013), System Analysis and Design. *International Student Version.* Fifth Edition. Reading: John Wiley

Sergi N., Victor H., Oscar R., Alberto A., Xavier F., Stijn V., Danilo V. (2017), A Software Reference Architecture for Semantic-Aware Big Data Systems

Shajema, I. (2018). Effect of Inventory Control Practices on Performance of Retail Chain Stores in Nairobi County, Kenya, *Journal of International Business, Innovation and Strategic Management*, 1(5), 18 - 38

Singh A., Masuku M. (2014), Sampling Techniques & Determination of Sample Size in Applied Statistics Research: An Overview. *International Journal of Economics, Commerce and Management, UK.* Vol. II, Issue 11, Nov 2014, ISSN 2348 0386

Sporta F. (2018), Effect of Inventory Control Techniques on Organization's Performance at Kenya Medical Supplies Agencies. *The International Journal of Business & Management.* ISSN 2321–8916

SYSPRO, Inventory Optimization for Better Supply Chain Management: Whitepaper

Tarigan Z., Jiputra J. & Siagian H. (2020), The Effect of Supply Chain Practices On Retailer Performance with Information Technology as Moderating Variable. *International Journal of Data and Network Science* 5 (2021) 47–54

Tongya Z., Gang C., Xinyu W., Chun C., Xingen W. & Sihui L. (2021), Real-time Intelligent Big Data Processing: technology, platform, and applications (Research Paper). College of Computer Science and Technology, Zhejiang University, China

Weibhuhn S. & Hoberg K. (2021), Designing Smart Replenishment Systems: Internet-Of-Things Technology for Vendor-Managed Inventory at End Consumers. *European Journal of Operational Research* 295 (2021) 949–964

Appendices

Appendix A: Data Request Format

Supply Data Attributes of a product

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Product Name	Product ID/Barcode	Date Ordered	Date Supplied	Lead Time (Time Taken from Order to Delivery)	Expected Delivery Timeline	Quantity Ordered	Quantity Supplied	Purchase Price	Selling Price	Total Cost of Delivery	Total Cost of Storage	Total Weight	Total Volume
2														
3														
4														
5														

Sales Data Attributes of a product

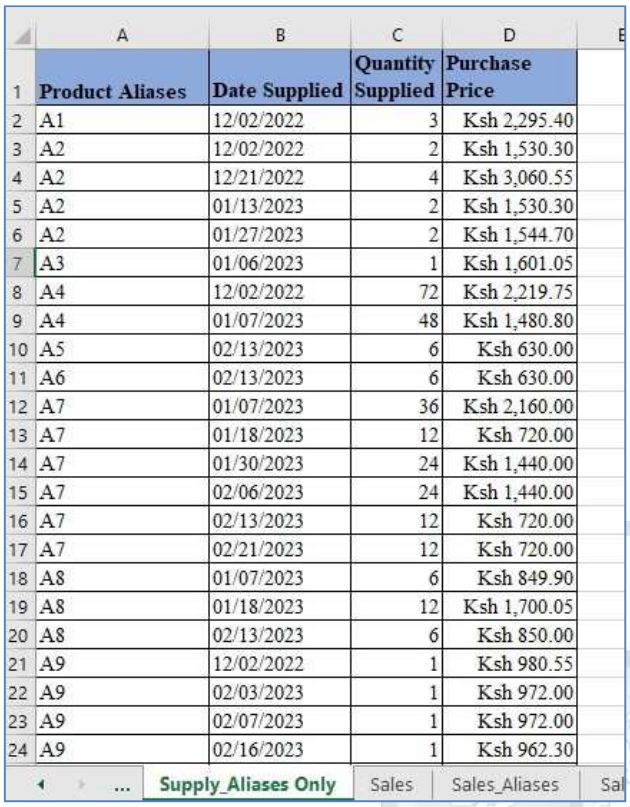
	Product Name	Product ID/Barcode	Quantity Sold/Released/ Picked	Selling Price	Date	Quantity in Stock
1						
2						
3						
4						

Appendix B: Data Received Snapshot

Date	Transaction Ty	Category	Transaction Code	Product Code	Quantity	Invoice N	Purchase Price	Vat	Net Purchase Price
02/01/2023									
1	Received	Bread A	1	white bread 800g	4.00	A1	440.00	0.00	440.00
2	Received	Bread B	2	White bread 600g	5.00	A1	425.00	0.00	425.00
3	Received	Bread C	3	white bread 400g	12.00	A1	660.00	0.00	660.00
4	Received	Milk A	4	longlife 500ml	12.00	A2	610.35	97.65	708.00
5	Received	Milk B	5	fresh milk 500ml	12.00	A2	568.95	91.05	660.00
6	Received	Milk C	6	vanilla 250ml	12.00	A2	517.25	82.75	600.00
7	Received	Confectionery A	7	Queen cakes 200g	4.00	A3	237.95	38.05	276.00
8	Received	Confectionery B	8	Tea cake	4.00	A3	137.95	22.05	160.00
9	Received	Confectionery C	9	Mini 350g	5.00	A3	426.70	68.30	495.00
10	Received	Confectionery D	10	Soft scones	5.00	A3	215.50	34.50	250.00
11	Received	Confectionery E	11	mandazi 200g	20.00	A3	724.15	115.85	840.00
12	Received	Confectionery F	12	muffins 300g	30.00	A3	2,793.10	446.90	3,240.00
13	Received	Drink A	13	Red 375ML	2.00	A4	1,489.65	238.35	1,728.00
14	Received	Drink B	14	Gin 75OML	3.00	A4	3,000.00	480.00	3,480.00
15	Received	Drink C	15	Wine 350ml	4.00	A4	1,931.05	308.95	2,240.00
16	Received	Drink D	16	Gin 350ML	4.00	A4	1,931.05	308.95	2,240.00
17	Received	Drink E	17	Beer Can 500ml	12.00	A4	1,962.10	313.95	2,276.05
18	Received	Drink F	18	Beer Malt Can 500ml	12.00	A4	2,404.35	384.70	2,789.05

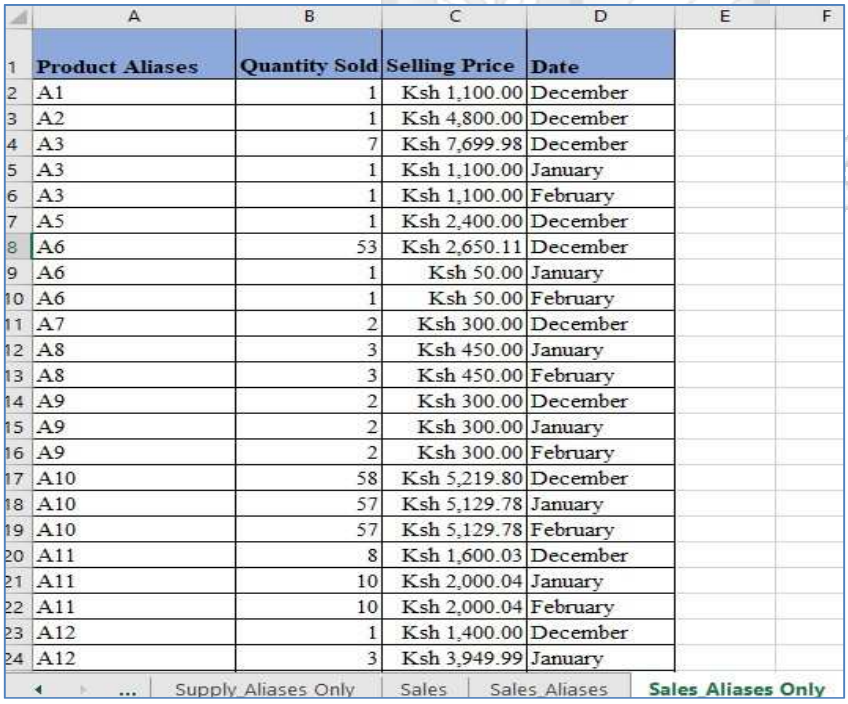
Code	Description	Category	Qty Sold	Gross Sales	VAT	Net Sales
1	Hanan Serviette	Kitchen and Catering	2.00	280.00	19.31	241.38
2	Exe All Purpose 2kg	Flours	2.00	520.00	0.00	520.00
3	Zesta Jam Red plum 500g	Jams & Spreads	2.00	499.99	34.48	431.03
4	Blue Band Original 250g	Jams & Spreads	4.00	800.02	27.59	689.66
5	Party Cups 500ml	Kitchen and Catering	78.00	780.03	1.38	672.41
6	Ketepa Tea Leaves 250g	Kitchen and Catering	1.00	160.00	22.07	137.93
7	Table Salt 500gms	Kitchen and Catering	2.00	60.00	4.14	51.72
8	Ketepa Tea Bag 50s	Kitchen and Catering	1.00	220.00	30.34	189.66
9	Ndhiwa Sugar 2kg	Kitchen and Catering	1.00	350.00	48.28	301.72
10	Velvex Super Soft Serviettes	Kitchen and Catering	1.00	150.00	20.69	129.31
11	Daawat Traditional biryani Rice 1kg	Rice	2.00	700.01	48.28	603.45
12	Cadburys Drinking Chocolate 125g	Kitchen and Catering	1.00	180.00	24.83	155.17
13	Indomie noodles chichken 70gms	Noodies & Pasta	14.00	560.01	5.52	482.76
14	kabras 2kg	Kitchen and Catering	1.00	350.00	48.28	301.72
15	kabras 1kg	Kitchen and Catering	3.00	600.01	27.59	517.24
16	Weetabix 112g	Kitchen and Catering	1.00	140.00	19.31	120.69

Appendix C: Edited Data Snapshots



	A	B	C	D
1	Product Aliases	Date Supplied	Quantity Supplied	Purchase Price
2	A1	12/02/2022	3	Ksh 2,295.40
3	A2	12/02/2022	2	Ksh 1,530.30
4	A2	12/21/2022	4	Ksh 3,060.55
5	A2	01/13/2023	2	Ksh 1,530.30
6	A2	01/27/2023	2	Ksh 1,544.70
7	A3	01/06/2023	1	Ksh 1,601.05
8	A4	12/02/2022	72	Ksh 2,219.75
9	A4	01/07/2023	48	Ksh 1,480.80
10	A5	02/13/2023	6	Ksh 630.00
11	A6	02/13/2023	6	Ksh 630.00
12	A7	01/07/2023	36	Ksh 2,160.00
13	A7	01/18/2023	12	Ksh 720.00
14	A7	01/30/2023	24	Ksh 1,440.00
15	A7	02/06/2023	24	Ksh 1,440.00
16	A7	02/13/2023	12	Ksh 720.00
17	A7	02/21/2023	12	Ksh 720.00
18	A8	01/07/2023	6	Ksh 849.90
19	A8	01/18/2023	12	Ksh 1,700.05
20	A8	02/13/2023	6	Ksh 850.00
21	A9	12/02/2022	1	Ksh 980.55
22	A9	02/03/2023	1	Ksh 972.00
23	A9	02/07/2023	1	Ksh 972.00
24	A9	02/16/2023	1	Ksh 962.30

... Supply_Aliases Only | Sales | Sales_Aliases | Sal



	A	B	C	D	E	F
1	Product Aliases	Quantity Sold	Selling Price	Date		
2	A1	1	Ksh 1,100.00	December		
3	A2	1	Ksh 4,800.00	December		
4	A3	7	Ksh 7,699.98	December		
5	A3	1	Ksh 1,100.00	January		
6	A3	1	Ksh 1,100.00	February		
7	A5	1	Ksh 2,400.00	December		
8	A6	53	Ksh 2,650.11	December		
9	A6	1	Ksh 50.00	January		
10	A6	1	Ksh 50.00	February		
11	A7	2	Ksh 300.00	December		
12	A8	3	Ksh 450.00	January		
13	A8	3	Ksh 450.00	February		
14	A9	2	Ksh 300.00	December		
15	A9	2	Ksh 300.00	January		
16	A9	2	Ksh 300.00	February		
17	A10	58	Ksh 5,219.80	December		
18	A10	57	Ksh 5,129.78	January		
19	A10	57	Ksh 5,129.78	February		
20	A11	8	Ksh 1,600.03	December		
21	A11	10	Ksh 2,000.04	January		
22	A11	10	Ksh 2,000.04	February		
23	A12	1	Ksh 1,400.00	December		
24	A12	3	Ksh 3,949.99	January		

... Supply_Aliases Only | Sales | Sales_Aliases | Sales_Aliases Only

Appendix D: Proportionate Sampling

	A	B	C	D	E
1	**Product Alias**	Agg Quantity Sold	Agg Sales	% of Product	Proportionate Sample (k)
2	A1	785	Ksh 3,925.24	0.11	0.73
3	A2	845	Ksh 4,225.28	0.12	0.79
4	A3	588	Ksh 29,401.75	0.82	5.50
5	A4	599	Ksh 11,980.83	0.33	2.24
6	A5	562	Ksh 40,025.00	1.12	7.48
7	A6	557	Ksh 36,205.00	1.01	6.77
8	A7	442	Ksh 26,521.56	0.74	4.96
9	A8	458	Ksh 29,771.09	0.83	5.57
10	A9	437	Ksh 17,480.63	0.49	3.27
11	A10	421	Ksh 12,630.71	0.35	2.36
12	A11	393	Ksh 23,580.00	0.66	4.41
13	A12	308	Ksh 24,638.91	0.69	4.61
14	A13	345	Ksh 27,598.72	0.77	5.16
15	A14	309	Ksh 21,591.33	0.60	4.04
16	A15	331	Ksh 26,478.78	0.74	4.95
17	A16	271	Ksh 2,710.14	0.08	0.51
18	A17	323	Ksh 48,410.09	1.35	9.05
19	A18	271	Ksh 2,710.17	0.08	0.51
20	A19	300	Ksh 45,000.09	1.25	8.41
21	A20	228	Ksh 63,839.86	1.78	11.94
22	A21	267	Ksh 13,350.67	0.37	2.50

... | Sales Agg | Sales Agg Summary | **Sales Summary Aliases**

Appendix E: Ethical Approval

Strathmore UNIVERSITY

10th March 2023

Mr Ndana Joe,
joseph.ndana@strathmore.edu

Dear Mr Ndana,

RE: Retailer-Supplier Stock Levels Optimization Tool

This is to inform you that SU-ISERC has reviewed and **approved** your above **SU- master's** research proposal. Your application reference number is **SU-ISERC1618/23.** The approval period is from **10th March 2023 to 9th March 2024.**

This approval is subject to compliance with the following requirements:

i. Only approved documents including (informed consents, study instruments, MTA) will be used
ii. All changes including (amendments, deviations, and violations) are submitted for review and approval by SU-ISERC.
iii. Death and life-threatening problems and serious adverse events or unexpected adverse events whether related or unrelated to the study must be reported to SU-ISERC within 48 hours of notification
iv. Any changes, anticipated or otherwise that may increase the risks or affected safety or welfare of study participants and others or affect the integrity of the research must be reported to SU-ISERC within 48 hours
v. Clearance for export of biological specimens must be obtained from relevant institutions.
vi. Submission of a request for renewal of approval at least 60 days prior to expiry of the approval period. Attach a comprehensive progress report to support the renewal.
vii. Submission of an executive summary report within 90 days upon completion of the study to SU-ISERC.

Prior to commencing your study, you will be expected to obtain a research license from National Commission for Science, Technology, and Innovation (NACOSTI) https://research-portal.nacosti.go.kc/ and obtain other clearances needed.

Yours sincerely,

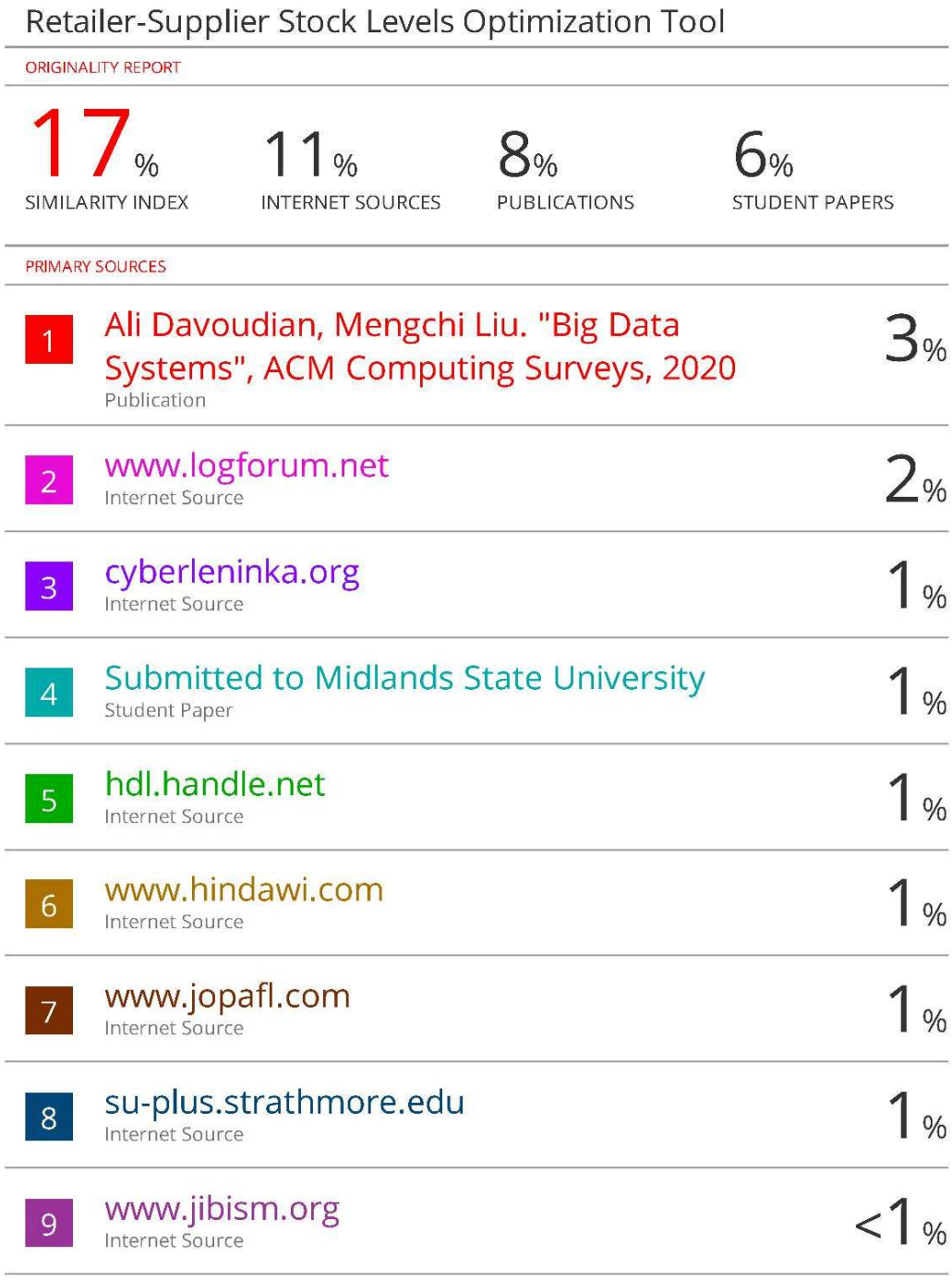
for: **Dr Ben Ngoye,**
Secretary; SU-ISERC

Cc: Mr Ambrose Rachier,
Chairperson; SU-ISERC

STRATHMORE UNIVERSITY INSTITUTIONAL SCIENTIFIC AND ECHICAL REVIEW COMMITTEE
(SU-ISERC)
10-Mar-2023
Email:ethicsreview@strathmore.edu
P.O BOX 59857-00200
NAIROBI-KENYA

Ole Sangale Rd, Madaraka Estate. PO Box 59857-00200, Nairobi, Kenya. Tel +254 (0)703 034000
Email admissions@strathmore.edu www.strathmore.edu

Appendix F: Similarity Check



Retailer-Supplier Stock Levels Optimization Tool

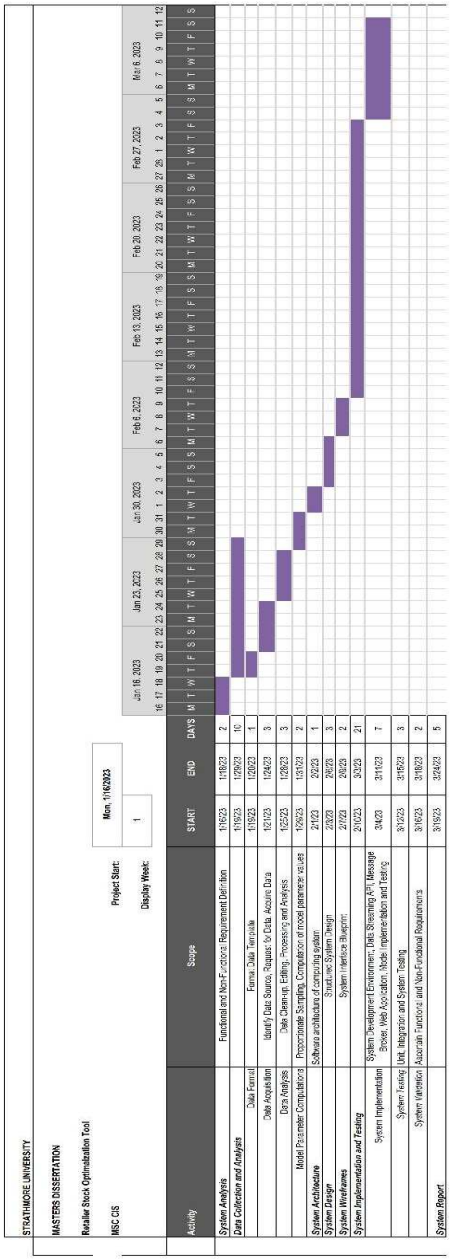
ORIGINALITY REPORT

17%	11%	8%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

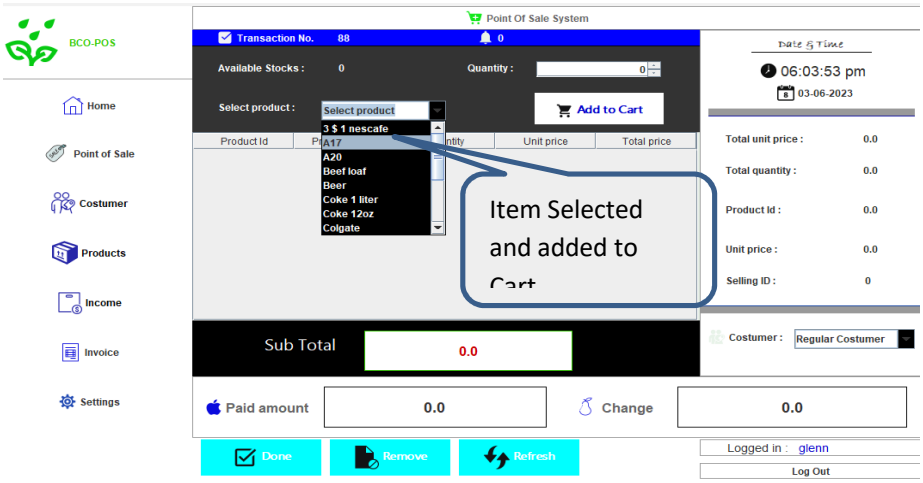
PRIMARY SOURCES

1	Ali Davoudian, Mengchi Liu. "Big Data Systems", ACM Computing Surveys, 2020 Publication	3%
2	www.logforum.net Internet Source	2%
3	cyberleninka.org Internet Source	1%
4	Submitted to Midlands State University Student Paper	1%
5	hdl.handle.net Internet Source	1%
6	www.hindawi.com Internet Source	1%
7	www.jopafl.com Internet Source	1%
8	su-plus.strathmore.edu Internet Source	1%
9	www.jibism.org Internet Source	<1%

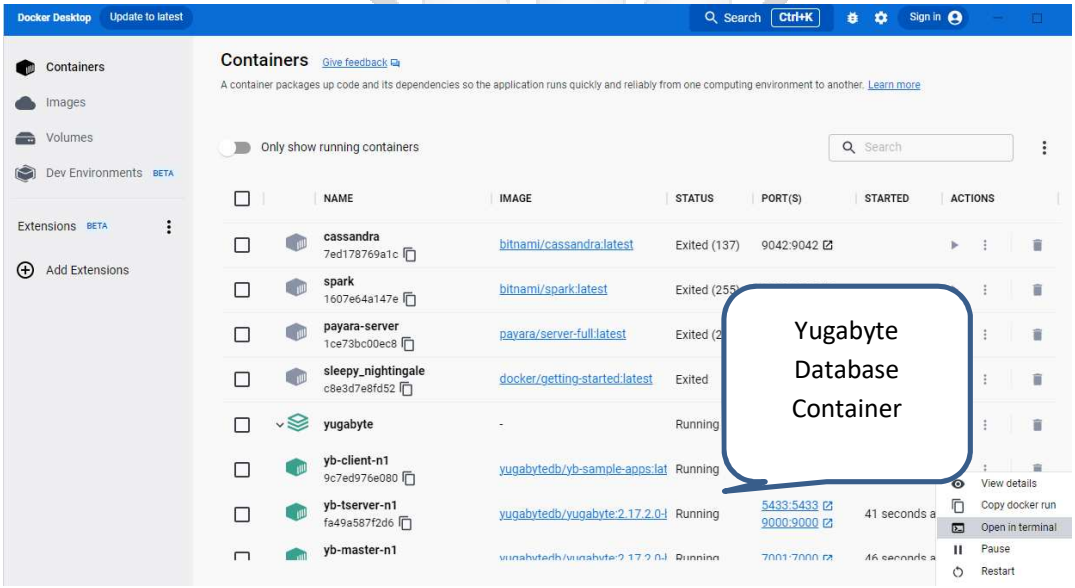
Appendix G: Project Plan Gantt Chart



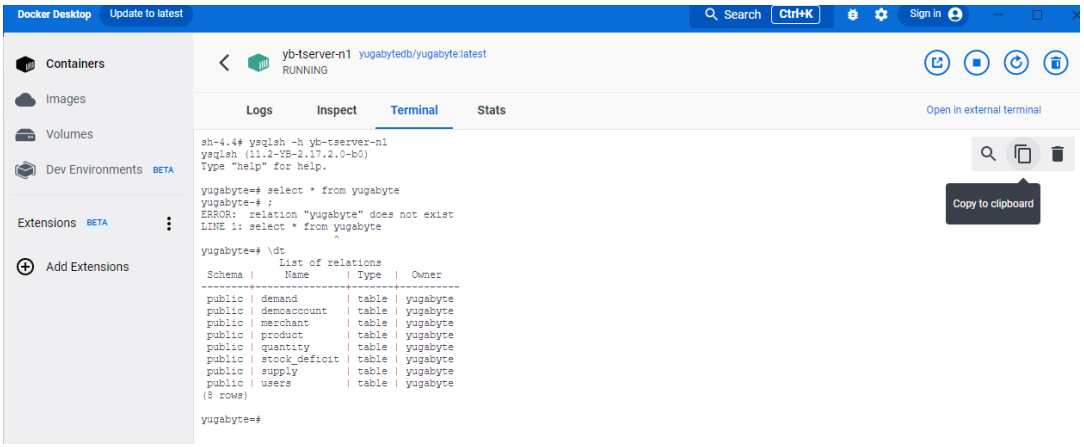
Appendix H: Open Source POS



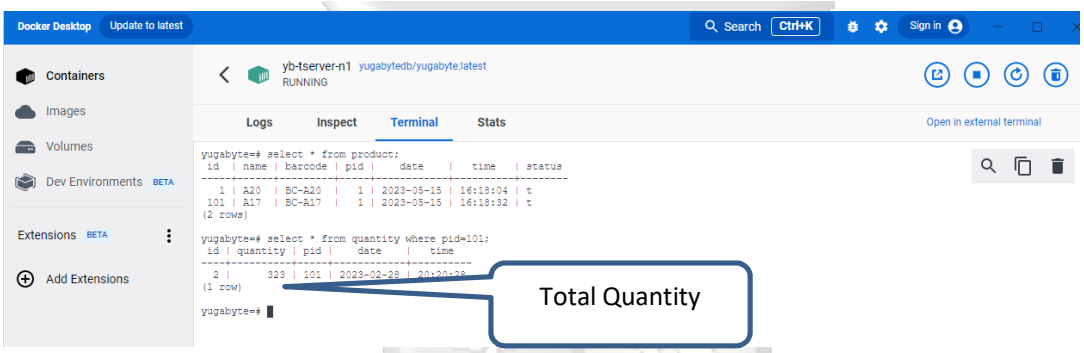
Appendix I: Docker Yugabyte Database container



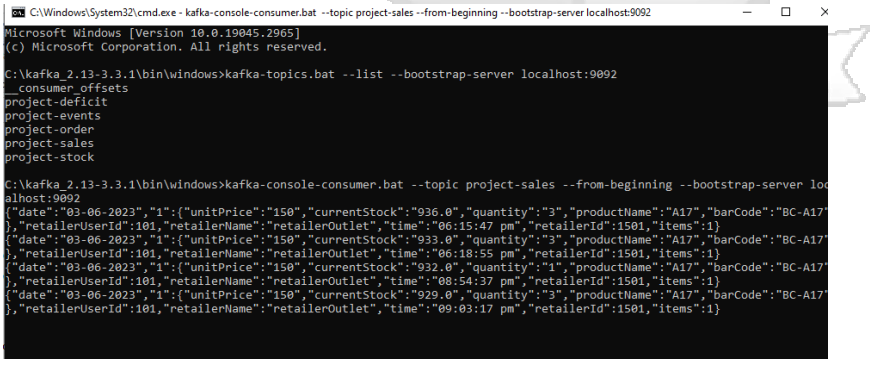
Appendix J: Tables in Yugabyte Docker Terminal Console



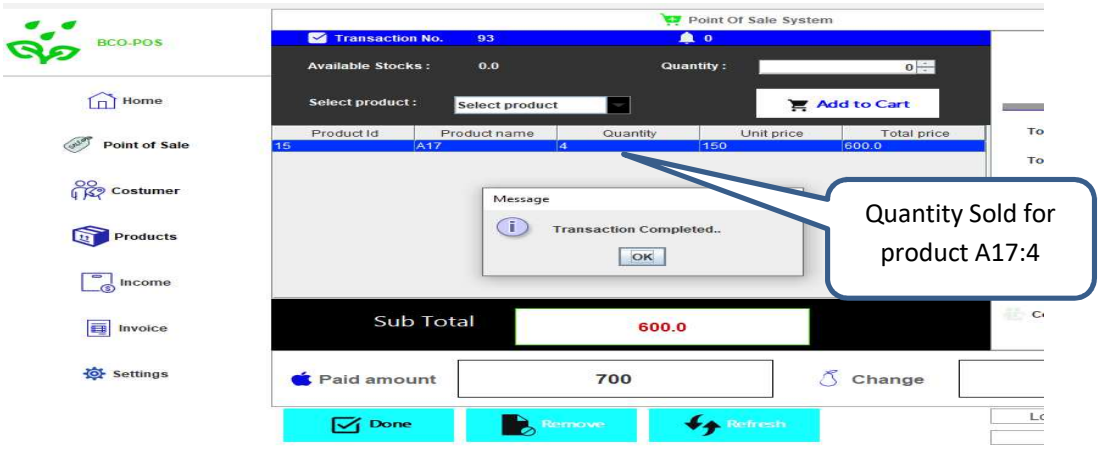
Appendix K: Product Data in Yugabyte DB before Ingestion



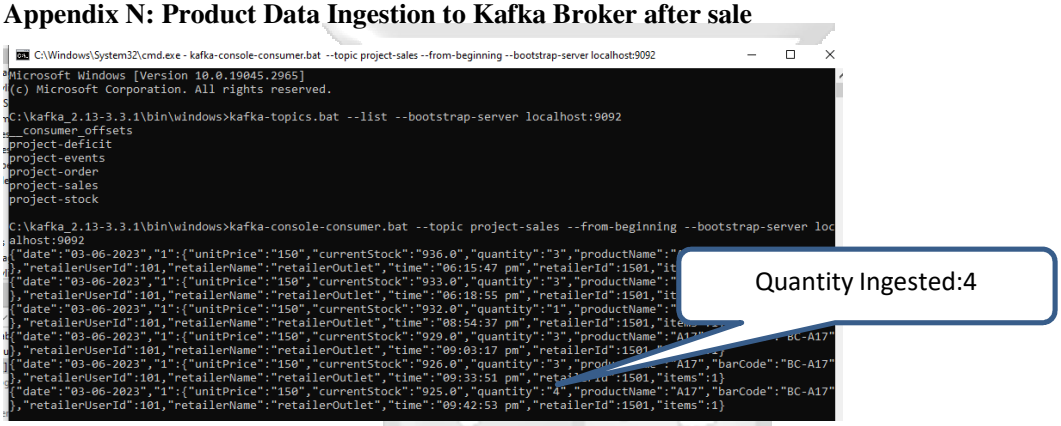
Appendix L: Product Data in Broker before Ingestion



Appendix M: Product Sale in POS

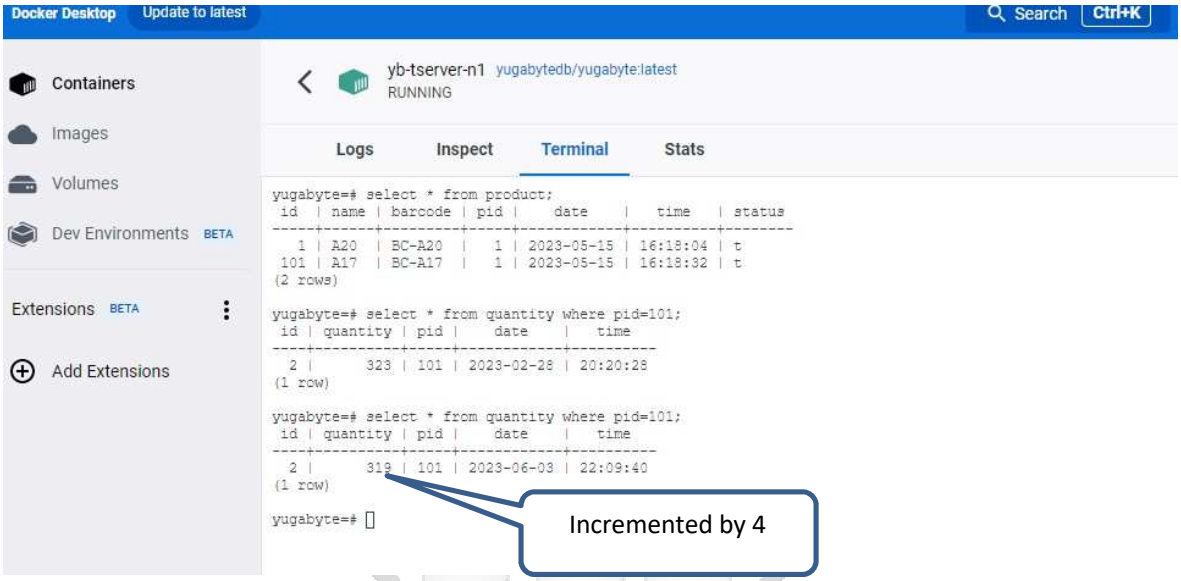


Appendix N: Product Data Ingestion to Kafka Broker after sale

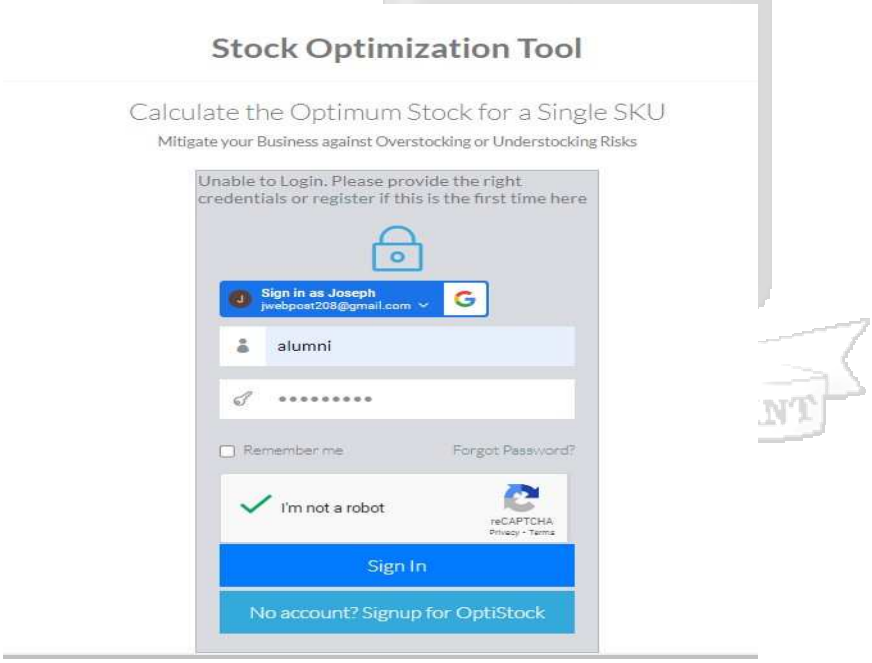


Appendix O: Product Data in JSON format streamed by spark to database

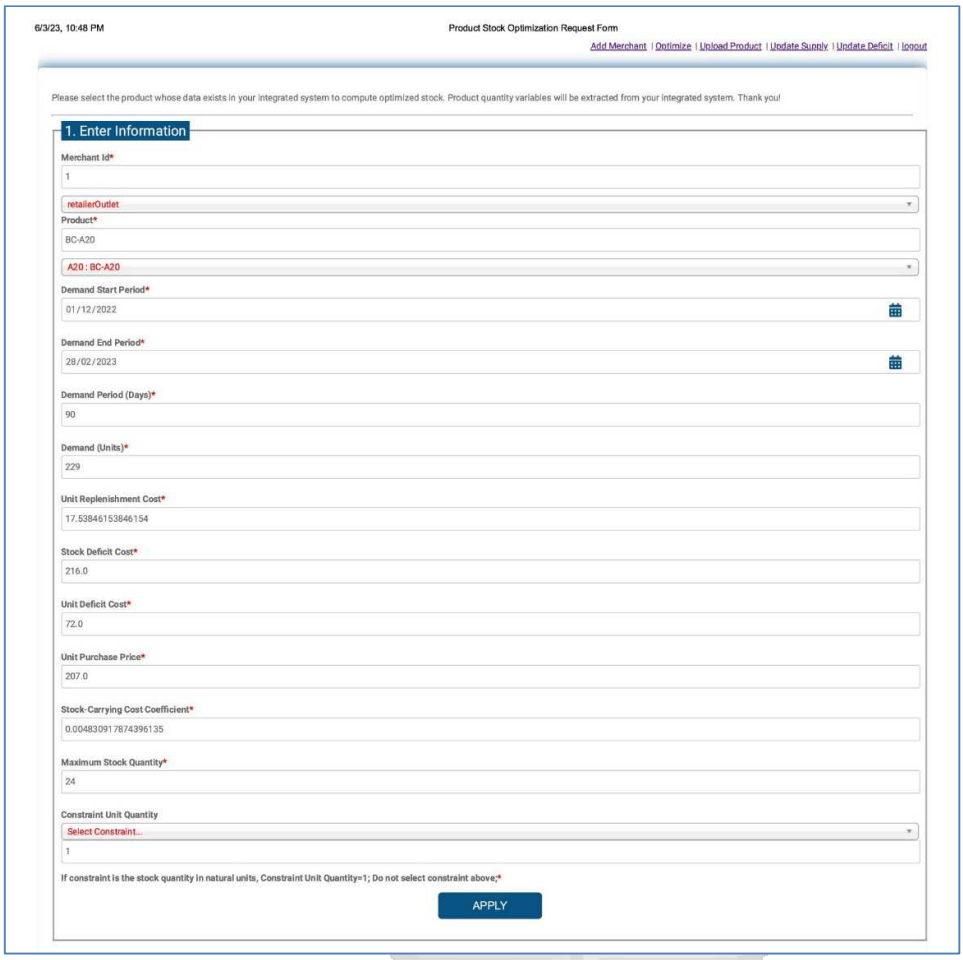
Appendix P: Product update after data streaming to database



Appendix Q: System Unregistered User Login



Appendix R: System User Query and Parameter Input



Appendix S: System Optimum Stock Level Output