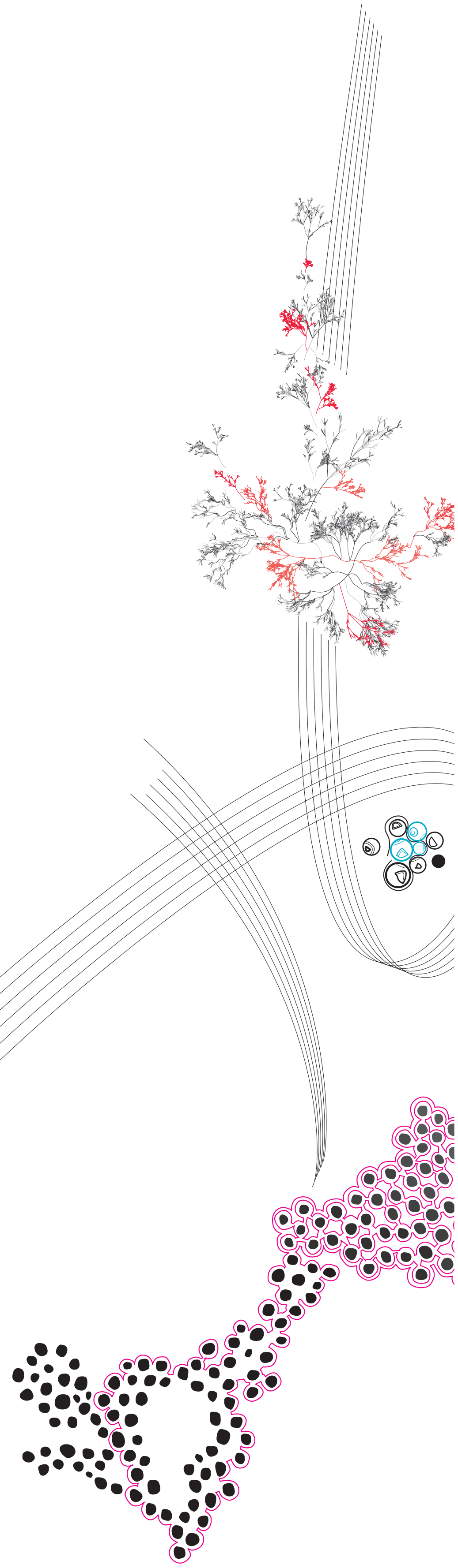


MSc Business Information Technology Thesis

Improving Decision Making in Warehouse: Data-Driven Forecasting and Storage Simulation

Jiayu Li

Supervisors:
dr. Luís Ferreira Pires
prof. dr. ir. Martijn Mes
dr. João Luiz Rebelo Moreira

November, 2023

Department of Computer Science
Faculty of Electrical Engineering,
Mathematics and Computer Science,
University of Twente

UNIVERSITY OF TWENTE.

Contents

Glossary

SKU (Stock Keeping Unit): A unique identifier for each distinct product. It helps in the management and tracking of inventory.

ABC Storage Policy: A method of managing inventory where products are categorized into three categories (A, B, and C) based on their importance. A-products are the most important, while C-products are the least. Products with higher importance are stored closer to pick/delivery point (e.g. arrival dock and departure dock for products).

DES (Discrete Event Simulation): A type of simulation that models the operation of a system as a discrete sequence of events in time. Each event occurs at a particular instant and marks a change of state in the system.

ABS (Agent-Based Simulation): A type of simulation where entities with certain behaviors are modeled as agents, interacting with each other to assess their impact on the system as a whole.

Digital Twin: A digital replica of a physical entity. By bridging the physical and the virtual world, data is transmitted seamlessly allowing the virtual entity to mimic the physical entity.

Data Warehouse: A centralized repository for storing large volumes of data from multiple sources. Data is organised and tailored to business analysis.

Worker Utilisation: A measure of how efficient workers are used in a warehouse, calculated by dividing the total working time by the total available time of the workers.

Waiting Time: The time it takes for a product before it can be processed by a worker in a particular task due to the queue it is waiting on.

Data Tool: A software tool with capability of using Python, R, or other programming languages that supports data retrieval, manipulation, and export.

Storage Simulator: A simulation model for a warehouse enabled by a simulation software.

Preface

This thesis is carried out at Bricklog, an IT service company serving the logistics industry. The company offers software and data-driven solutions for small-and-medium sized transport companies. We participated in an initiative project which revolves around building data-driven solutions for warehousing companies.

Abstract

The thesis presents a data-driven solution that utilises a data warehouse and several software tools to help decision making in warehousing business. The solution consists of the forecasting service and the storage simulator service. In warehousing, forecasting is often desired to predict the daily workload to meet the demand for picking and to have an appropriate number of workers available for the day. Storage policy is a method to allocate products in different sections of the storage primarily based on their demand in order for pickers to reach these products faster. Forecasting the picking demand with higher accuracy and choosing a suitable storage policy help improve efficiency and reduce costs in warehousing operations. We illustrated how these two services are realised with an architecture based on a data warehouse. Use cases of the services were created to help warehousing managers and supervisors make improved data-driven decisions.

Chapter 1

Introduction

This chapter introduces the context of the problems addressed in the thesis and the approach to solve them. The following subsections are included: Problem Description, Research Questions, Methodology, Approach, and Structure of the Thesis.

1.1 Problem Description

Many SME warehouses or companies with warehouses do not yet have or fully use data-driven technology, such as data warehouse, dashboard or simulation applications, to help them improve operational effectiveness or make more informed decisions. What they usually have is a collection of enterprise software and databases that are dedicated to specific business processes and hold records of operational data such as customers, orders, financials, inventory, transports, and planning. These data could be put together in a data factory to build a data warehouse. Inmon defines a data warehouse as “a subject-oriented, integrated, time-variant, and non-volatile collection of data in support of management’s decision-making process" [29]. A data warehouse can be used by various software tools to provide various data-driven services such as data quality improvement, customer and inventory analysis, forecasting, and storage optimisation.

Two common challenges in warehousing are addressed in the thesis: demand forecast and ABC storage policy.

The forecast of demand for workload [21] [7] and products [27] helps warehouse managers and supervisors to anticipate demand and plan resources in advance. Studies [21] [7] [27] [84] [25] have shown that forecasts backed by historic data and refined forecasting algorithms can improve forecasting accuracy. With a data warehouse, the data required for forecasting algorithms can be easily retrieved. There are a myriad of studies that apply different algorithms for demand forecasting with real case studies, which range from custom-built algorithms[27] to complex Machine Learning algorithms[70]. Traditional models such as ARIMA and Exponential Smoothing are usually the choices for many forecasting applications. However, case studies of applying newer algorithms such as XGBoost and LightGBM (both introduced in 2016) in warehousing context are still lacking. So, we aim to include both traditional algorithms and new algorithms to compare their performance in the forecasting solution.

A storage policy determines how the stored products are allocated or organised in a storage area. This is a common concern for a warehouse because it affects the picking time and travelling distance[82], which are the KPIs that translate into the labour cost and the performance cost[16]. In general, products with higher demand should be more easily and quickly accessed for picking. To solve this, ABC storage policy is a commonly used method and is proven to be effective in reducing travelling distance and picking time in several studies [76] [46] [73] [11]. These studies also showed the viability of using the simulation approach, which produces quantitative assessment of the impact of different policy settings on the travel distance and picking time. For example, a real case study [73]showed that a proposed ABC policy combined with a change of storage layout could reduce the distance travelled and the picking time by more than 40% on average in a simulation model for a warehouse. The advantage of using simulation is the ability to perform experiments that are otherwise not feasible in the real world. For example, a simulation study [76]designed a model to balance the cost and pick-up efficiency of the forward and reserve zones in a warehouse by varying the number of storage bins. When multiple experiments were performed, an optimal number of bins was found for the fast-pick zone. Another paper [46] showed a comparison of the impact of ten different arrangements of products based on different ABC classification methods on the average picking time in a case study. However, many of these studies did not mention the methods of measuring the KPIs. For example, some studies did not say whether the picking time include the time for a forklift to search, elevate, or retrieve a pallet. Our goal is to design a simulation model for a warehousing storage environment with clear measurement methods for KPIs, which are used to evaluate variants of ABC storage policies.

To solve these two problems, we used a data warehouse containing synthetic data of the movement history of bulk products in a hypothetical warehouse. With the architecture shown in Figure 2, we generated forecast and made a simulation model that ran experiments on the variants of ABC storage policy. The architecture consists of a Data Tool that carries out forecasting and a Storage Simulator that carries out the evaluation of the storage policies. The key data resulting from the solutions are presented to stakeholders with a visualisation tool. The forecasting solution is to be used daily for approximating the picking demand. The connection between the picking workload forecasting and the Storage Simulator in the current design is that the forecast should provide a reference to how many pickers are needed in a simulation experiment with a certain picking workload given a fixed time horizon. The Storage Simulator is to be used quarterly to determine the best storage policy for the current quarter. There are two reasons for it to be used quarterly: the first is that a warehouse should not change the storage policy too frequent because it would add regular operational and managerial expense; the second is that we assume the warehouse have a seasonal demand pattern that could be summarised on quarterly basis, and that if there are newly introduced products in the next quarter, the policy should be changed accordingly.

1.2 Research Questions

The main research question we answer in this thesis is:
How can a warehouse improve decision making by using data-driven services consisting of picking demand forecast and storage simulation?
The main question is decomposed into the following sub-questions:

- RQ1: Which forecasting method is the most suitable to predict daily picking demand?
- RQ2: Which factors affect the effectiveness of an ABC storage policy?
- RQ3: How to model an environment with storage and other areas of a warehouse and measure KPIs including the travel distance and picking time in the picking process?
- RQ4: How do warehousing stakeholders use the forecast and storage simulation services for decision making?

1.3 Methodology

This section describes the process of designing the solution with Design Science Methodology along with the solution's components.

1.3.1 Using Design Science Methodology

Figure 1 shows the steps of applying Design Science Methodology for our work. We reviewed the literature related to the forecasting and ABC storage policy. Based on the literature review findings, we carried out design & development comprising two services. The first phase of the design is the forecasting service. The forecasting service is about making workload forecasts through statistical and Machine Learning methods and present forecasts to stakeholders with intelligible visualisation. The second phase of the design is a storage evaluation service built upon a simulation model we developed. The service consists of the estimation of picking time and travelling distance resulting from different variations of ABC storage policies and of a resource monitor measuring utilisation of workers and waiting time in the picking process. The Demonstration & Evaluation step includes use case description of the artefact and the expert survey.

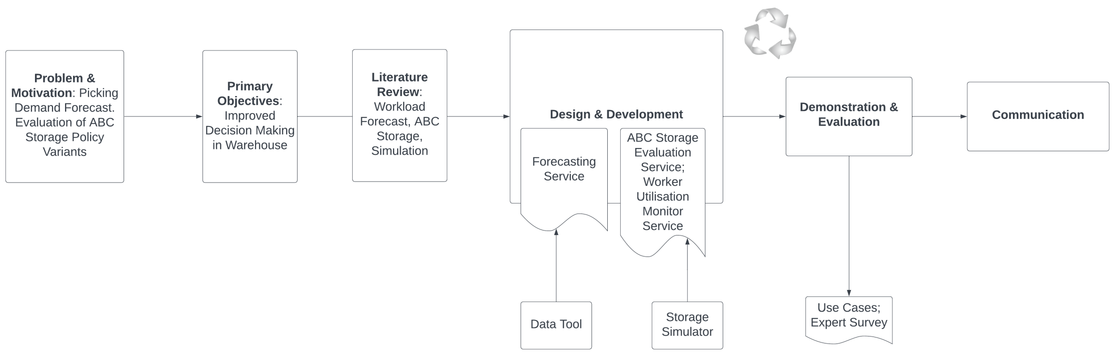

Problem & Motivation: Picking Demand Forecast. Evaluation of ABC Storage Policy Variants
Primary Objectives: Improved Decision Making in Warehouse
Literature Review: Workload Forecast, ABC Storage, Simulation
Design & Development
Forecasting Service
ABC Storage Evaluation Service; Worker Utilisation Monitor Service
Data Tool
Storage Simulator
Demonstration & Evaluation
Use Cases; Expert Survey
Communication

Figure 1: Steps of Applying Methodology

1.3.2 Literature Review

We reviewed the literature related to warehousing management, demand forecasting, ABC storage policy, and simulation to gain an understanding of these subjects and solutions to similar problems from other studies and to form connections with these papers. The literature review addressed the following aspects:

- L1. Common data-driven forecasting methods in a warehouse.
- L2. Forecasting methods with relatively better performance than others in case studies.
- L3. The common variants of ABC storage policy and the variation of ABC storage policy that may lead to minimal travel distance and picking time in the picking process.
- L4. Besides variants of policy, the other factors that may affect the efficacy of an ABC storage.
- L5. Simulation models suitable for the evaluation of ABC storage policies.
- L6. Use cases or features that a simulation application can offer to help decision making in a warehouse.
- L7. Communication medium of forecasting and simulation solutions in the case studies with warehousing stakeholders.

L1 and L2 are connected to RQ1; L3 and L4 are connected to RQ2; L5 is connected to RQ3. L6 and L7 are connected to RQ4.

1.3.3 Scope and Components

The components of the solution are shown in Figure 2. We assume that a Data Tool is available and able to retrieve data, manipulate data, and apply algorithms for forecasts. A Data Tool can be a development environment with Python or R language which supports this kind of operations. We also built a simulation model with which we can run experiments and evaluate different ABC storage policies. The results of forecasts and storage evaluation are formatted and presented in various graphs or charts through a visualisation tool to help decision support for warehousing stakeholders.

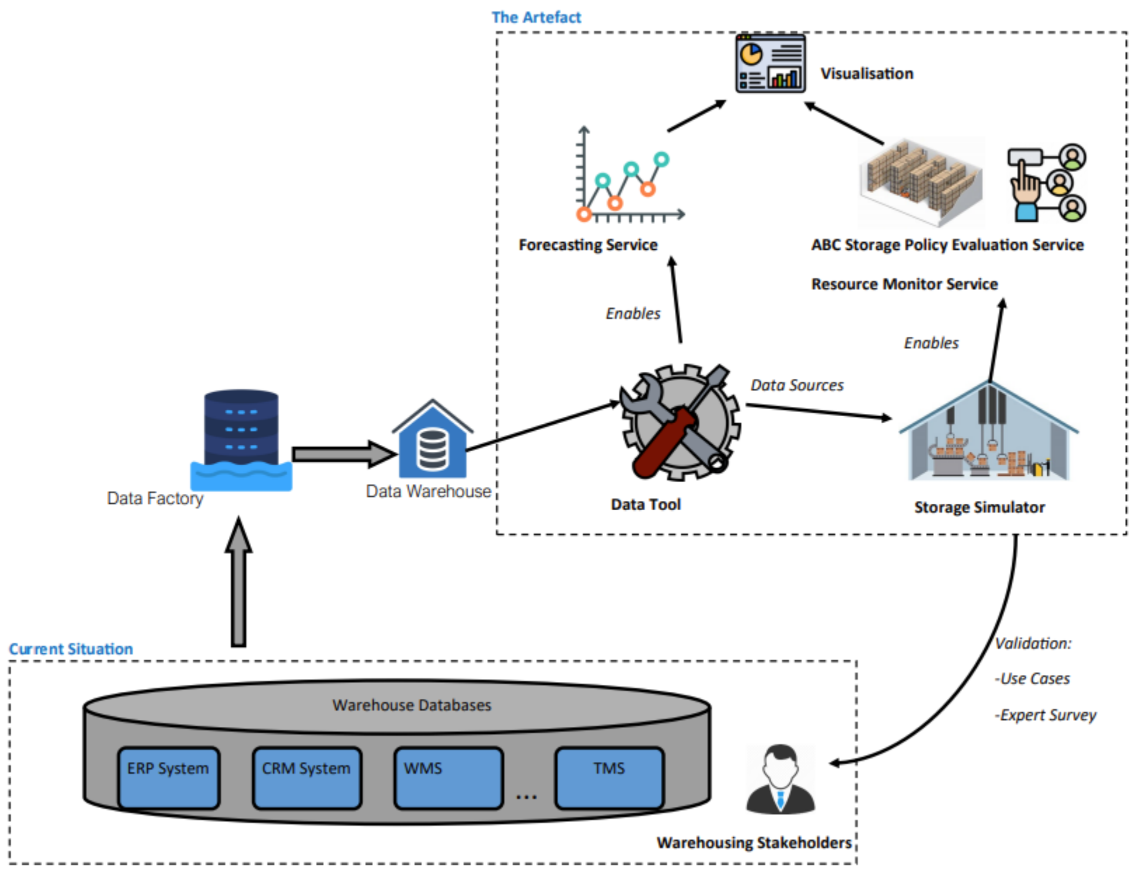


FIGURE 2: Conceptual Design of the Solution

1.3.4 Forecasting Service

Synthetic data provided by the company is used to train and test the forecasting methods: SARIMA, exponential smoothing, LightGBM, and XGBoost. The data includes movement history of SKUs from 2020 to 2022, indicating the daily workload in this period. Data analysis includes graphs of daily workload, histogram, and seasonality decomposition. The data pre-processing removed Sundays since there is no movement on these days. The train set includes daily workload history up until 30 days before the end of history; the test set is the last 30 days in the history. The forecasting service consists of the calculated workload forecasts and their visualisation based on use cases.

1.3.5 Storage Simulator Service

The steps to build the Storage Simulator include building a conceptual model, defining KPIs, and designing experiments.

Conceptual Model for Simulation

We employed the conceptual model framework proposed in[61] (see Figure 20) for the simulation model, and used a mixture of Discrete Event and Agent-based approach.

KPIs

KPIs are defined for travelling distance, total picking time, for ABC storage evaluation service. Worker utilisation, and total waiting time are defined for the resource monitoring service.

Experiment Design

Two experiments were designed. (1) the evaluation of the ABC storage policy; we constructed horizontal, low-level, diagonal, and random policies, then measured travelling distance and picking time for each policy. (2) resource monitor; we measured the utilisation of the workers for the checking task in the picking process. We also measured total waiting time of SKUs for the checking task. In addition, a chart was drawn and configured in the simulation software to show congestion resulting from waiting SKUs.

Verification and Validation

Our verification included extreme conditions, modularised testing, debugging, animation, and graphic inspection. Validation is based on the framework provided by North & Macal (2007) [53]. The following validations were performed: data validation, process validation, face validation, theory validation, and requirements validation.

1.4 Structure of the Thesis

The thesis is structured with the following chapters.

Chapter 2 explains the approach of the literature search and answers the literature questions from the materials consulted about picking demand forecasts, ABC storage policy, simulation applications, simulation methods, and how these applications benefit warehousing stakeholders.

Chapter 3 presents the workflow and the development of the forecasting service by analysing data and applying algorithms in a Python development environment; After comparing the performances of the algorithms, SARIMA forecasting method is found to be the best and is selected for daily picking workload demand.

Chapter 4 presents the modelling and implementation of the Storage Simulator that enables the evaluation of ABC storage policy variants and a simple resource monitor in a hypothetical warehouse.

Chapter 5 illustrates the experiments conducted in the Storage Simulator to assess total distance travelled and picking time in the variants of ABC storage policy, and to assess the workers' utilisation and waiting time during the picking process.

Chapter 6 discusses how stakeholders can benefit from the designed solution through use cases, and discusses the evaluation of our visualisation by expert surveys.

Chapter 7 concludes the thesis by discussing each research questions, related works, and contributions. Recommendations are also given for future work.

Chapter 2

Literature Review

This chapter explains the search strategy for the literature review and presents an overview of topics related to research questions. The subjects include warehouse management matters related to forecasts and storage, workload and product demand forecasting, simulation methods, simulation applications in warehousing, ABC storage policy, and mechanisms with which stakeholders interact with forecasts and simulation applications for decision making.

2.1 Literature Search Strategy

To answer the research questions, we consulted academic papers, conference articles, grey literature, books, and website articles. The academic papers were selected from various sources including IEEE, Springer, ResearchGate, Elsevier and other journal websites. Snowballing method is also used to find more relevant articles by reading and selecting related articles from the reference list of the current article. General search engines such as Bing and Google were also employed to help locate relevant papers and articles. Grey literature is sourced from companies such as Anylogic and DHL who are active in the supply chain domain.

Subjects	**Keywords used for Literature Search**
Aspects of Warehouse Management	Warehouse operations, inventory management, warehouse layout, material handling, picking process, human resources, labour cost, labour utilisation, KPI.
Demand forecast	Workload demand forecast, product demand forecast, inventory forecast, warehouse workload forecast, sales forecast, picking demand forecast
Simulation Methods	Discrete event simulation, agent-based modeling, hybrid simulation
ABC Storage Policy	Class-based storage, ABC storage, SKU classification, product classification, storage policy, slotting policy
Simulation Applications in Warehousing	Warehouse simulation, Digital Twin simulation, warehouse process simulation

Table 1: Subjects and Corresponding Keywords for Literature Search

2.2 Demand Forecasting

In this section we explain demand forecasting of workload and products. Product demand forecast is not directly related to our design, it is reviewed for contextual information on warehousing management.

2.2.1 Workload Demand Forecast

Daily workload in a warehouse can be approximated by the number of order lines as it implicates the number of pickers needed for a given day[21]. Both statistical and machine learning methods can be used to predict the workload demand. For example, a study designed a composite forecasting method [21] to predict the workload for zones in a real warehouse in Belgium. It achieved higher forecasting accuracy than before, when the predictions were based on supervisors' experience and judgement. By employing statistical methods including exponential smoothing, SARIMA, and ARIMAX, it reduced MASE error from 0.33 to 0.29. Additionally, the study compared the top-down approach (which forecast all zones), and the bottom-up approach (which forecast individual zones) and found that the bottom-up approach outperformed except for one zone which has the least activities. It was because of the strong fluctuating demand and lack of recurring weekly cycle in that zone. Paper[37] argues that a combination of statistical methods and empirical methods may enhance the forecasting performance. It suggests using a linear model to include the prediction from the statistical method and the prediction from managers to optimise the forecasts based on the weighting of these two factors. Another study [7] compares Prophet, ARIMA, SARIMA, and LightGBM methods. They were used to predict picking volume of several warehousing sites and the performance of Prophet was comparable to ARIMA and SARIMA. LightGBM notably showed improved results in short-term horizon up to ten to fourteen days and then its accuracy became almost the same as the other methods. The LightGBM model used features such as upfront forecasts, open orders and open shipments. Another study found that Generalised Linear Model was the best performing model compared to Deep Learning, Random Forest, and Gradient Boost Trees (which is what LightGBM and XGBoost based on) in predicting the workload in a Dutch supermarket [27] It used shift, date specifics, service percentage and planned hours, and so on as coefficients for the model.

When it comes to the impact of forecasts, a study argued that some bias for forecasting is desired. If the bias is related to labour and inventory costs, it can reduce costs [41]. Under-forecasting is supreme if labour cost is an important consideration and over-forecasting is desired if the main costs are delay panties for stock-outs [37].

2.2.2 Product Demand Forecast

Products are stored in the inventory of a warehouse. For some types of warehouses such as spare parts warehouse or manufacturer's warehouse, they usually try to keep the optimise inventory level for each product to balance supply and demand. An excessive inventory of a product means that there is oversupply and it occupies the storage that could be otherwise used for other products. Conversely, if the inventory of the product is low, there is a danger of losing revenue because it may run out of products that could have been sold to customers. Therefore, each product must have certain amount of storage level in order to meet demand [52] while it is also monitored for oversupply [36]. For other types of warehouse such as third-party warehouses, they generate revenue by storing products for customers. They try to store as many as products as possible and expand inventory's capacity when there are more products coming in to meet the demand.

There are studies that used other machine learning methods for product demand forecast. For example, A study [70] compared the performance of convolution neural network (CNN), long short term memory (LSTM), and artificial neural network (ANN) and found CNN to be the highest performer for outbound product demand with long term horizon. The downside is that a neural network is not easily understandable or explainable and the study used no exogenous features. Peak Range Prediction was used in another study [13] to predict shipment demand of items in a warehouse. It demonstrated that it performed better than ARIMA and Peak Auto

Prediction; it also handles the issue of shifting demand. This issue occurs for example, the peak demand occurring in 51st week in 2018 could also occur in 52nd week in 2019. The authors provided structure and steps of applying the Peak Auto Range algorithm. However, they did not mention the mathematical model behind to compute peak range, so replication using this method is not possible until the model is specified.

Product demand forecast is a complex subject since the demand may be influenced by many factors such as customers' behaviour, product life cycle, economic conditions, and so on. We touched on this subject to give some contextual information about warehousing management. It requires a more comprehensive review in future study and is outside the scope of the forecasting service in this thesis.

2.2.3 Summary of Demand Forecast

To summarise demand forecasting, statistical methods and machine learning methods are applied in various case studies. A warehouse may use statistical methods such as ARIMA and exponential smoothing without external features, and a more advanced warehouse may prefer more complex algorithms such as Prophet and LightGBM with exogenous features such as incoming orders, date specifics, previous predictions, and so on.

The review addresses L1: common data-driven forecasting methods in a warehouse. The answer is: common demand forecasting methods may include ARIMA, exponential smoothing, gradient boosting tree algorithms such as LightGBM and XGBoost. It also addresses L2: the forecasting methods showing relatively better performance than others in the case studies. The answer is: using composite ARIMA and exponential smoothing method to make workload predictions showed the best performance among statistical methods in one study. Using LightGBM showed better results in short-term forecast than statistical methods and prophet in another study.

2.3 ABC Storage Policy

This section reviews ABC division methods, variations of ABC storage policy and other factors that may affect the efficacy of an ABC storage.

2.3.1 ABC Classification Methods

ABC category of a product can be classified through various methods. For example, if categorised by movement frequency, the products that are moved in or out the most should be classified as A Class and stored in the storage area where it is closest to the pick/delivery points (which would take pickers the least amount of time and travelling distance to reach these products). In this way, picking efficiency is improved. Class B products are located further than A products, and Class C products are the furthest away from the pick/delivery points[16]. Other methods to categorise ABC category include quantity [60], profit, COL (Cube-per-Order Index)[30] [46], EIQ analysis [47], and weight & volume of the product [12][54]. Some argued that classifying ABC products in sub-categories offer more benefits: as seen from Figure 3, each one division offers actionable insight in relation to the business strategy. Similarly, another paper[30] used a recursive method to classify products in sub-categories to reduce picking costs further compared to using just three categories.

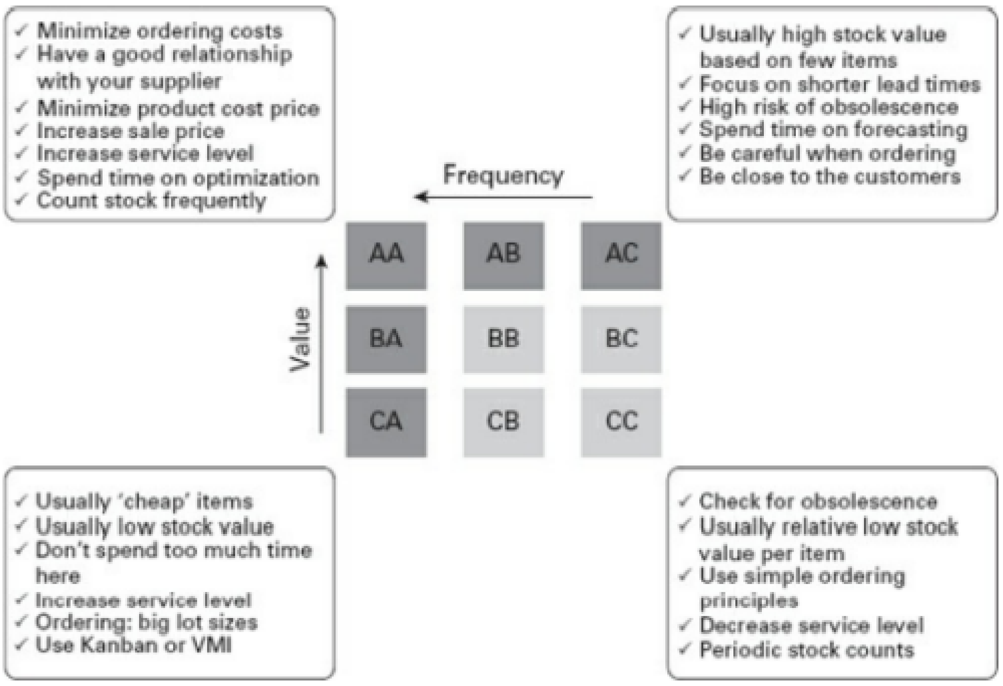

✓ Minimize ordering costs
✓ Have a good relationship with your supplier
✓ Minimize product cost price
✓ Increase sale price
✓ Increase service level
✓ Spend time on optimization
✓ Count stock frequently
✓ Usually high stock value based on few items
✓ Focus on shorter lead times
✓ High risk of obsolescence
✓ Spend time on forecasting
✓ Be careful when ordering
✓ Be close to the customers
Frequency
Value
AA
AB
AC
BA
BB
BC
CA
CB
CC
✓ Usually 'cheap' items
✓ Usually low stock value
✓ Don't spend too much time here
✓ Increase service level
✓ Ordering: big lot sizes
✓ Use Kanban or VMI
✓ Check for obsolescence
✓ Usually relative low stock value per item
✓ Use simple ordering principles
✓ Decrease service level
✓ Periodic stock counts

FIGURE 3: ABC Product Classification [60]

2.3.2 Variants of the ABC Storage Policy

A specific arrangement of A, B, C areas/zones on a storage is called a variant. Common variants of the ABC storage policy include horizontal (across-aisle), vertical (within-aisle), diagonal, and perimeter policies [12, 56]. These policies may perform differently [71], based on pick/delivery points. Figure 4 shows the visualisation of some variants of ABC policies. Area A, B, and C are allocated differently for each variation, but their common goal is to minimise the distance from Area A (in pitch black) to the P/D point. Area B (in grey) is located further, followed by Area C.

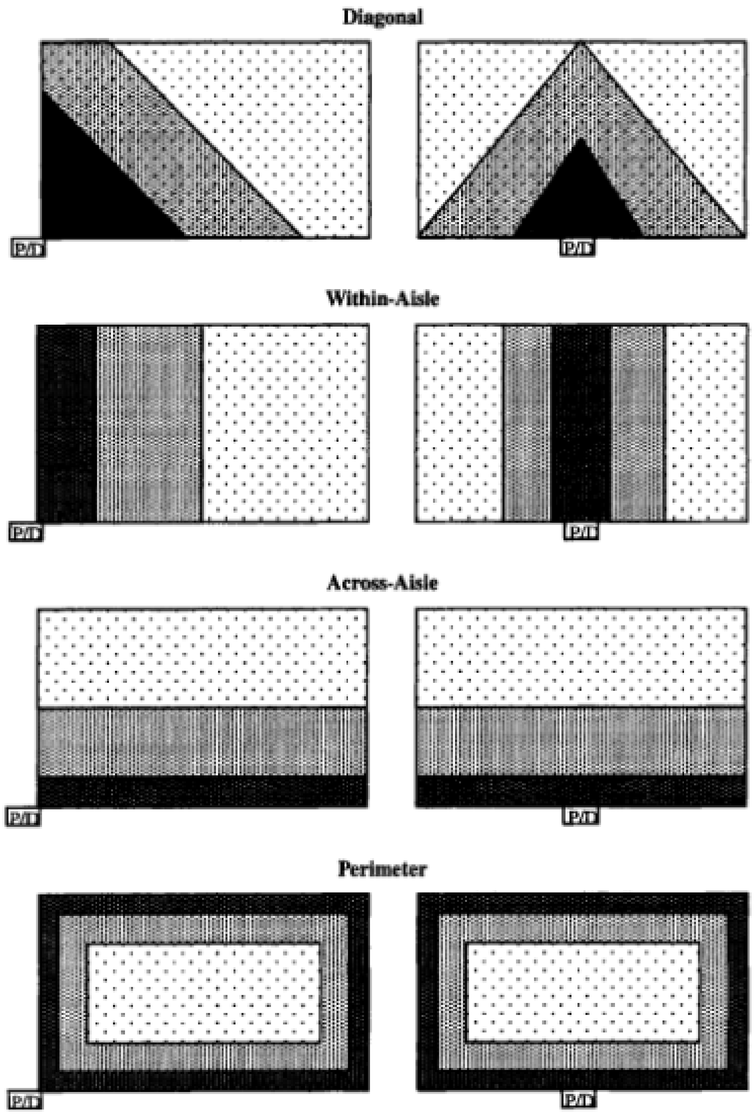


FIGURE 4: Variations of ABC Policy [56]

2.3.3 Layout and ABC Zone Sizes

Layout design can affect the effectiveness of an ABC Policy. For example, an ABC storage policy does not work well with a drive-in storage. A drive-in storage is that in which the products that are stored the first will be retrieved the last. It makes it difficult to retrieve a selected product in the middle or in the end of the storage lane. Changing it to a rack storage allows products to be retrieved straightaway via the access lane. Because of this reason, a study [74] changed the layout of a warehouse from a drive-in storage to a rack storage in order to apply an ABC storage policy. Another layout factors are the storage lane depth, explained by the study [15]; it argued that a warehouse should determine suitable lane depth and the size of ABC zones to prevent overstocking which occurs when there are excessive items forced to be stored in another zone. When it comes to the sizing of ABC zone, there are heuristics with the 20/80 rule and the 20/30/50 rule. For instance, the 20/30/50 rule assigns 20% zone space for A products, 30% for B, and 50% for C. These rules assign an arbitrary sizes for A, B, and C zones. However, they are not the optimal size allocation: a study [67] aimed to optimise ABC zone sizes using a simulator method originated from paper [62] and then applied Machine Learning methods to determine optimal zone sizes and showed highly improved picking efficiency (measured by average route length for picking) compared to arbitrary sizing. The features used in the Machine Learning model include: number of aisles. aisle length & width, routing policy, storage policy variation, and so on. Among the algorithms, random forest and multi-layer perception methods performed better than ordinary least square and regression tree. Machine Learning methods are fast and fairly accurate, it argued. It avoided the downside of having to keep running the simulator, which requires huge amount of data to simulate the warehouse settings such as routing policies, pick lists, and zone sizes. It was extremely slow (as it took 12,000 CPU hours to simulate 16,000 instances). Applying Machine Learning method would save significant amount of time and they could still achieve high accuracy in determining optimal zone sizes.

2.3.4 Real-time Assignment of Products Based on ABC Storage Policy

When it comes to assigning incoming products to specific slot/cell of a rack storage, there are real-time methods available. In a warehouse with stable demand and supply pattern, storage allocation of individual product can be done using a linear programming method that can satisfy constraints such as storage space utilisation and travelling distance[73]. If the demand pattern of products is erratic, a dynamic assignment can be achieved through the method proposed by paper [57]. Another method is affinity-based [76] [45] (which was shown to reduce travelling distance significantly in a case study[45]). Affinity method stores products that are frequently ordered together close to each other. Combining ABC storage policy and real-time assignment methods such as the affinity method would likely to improve the effectiveness further as shown by a case study using simulation[45].

2.3.5 Summary of ABC Storage Policy

The section addresses L3: common variations of ABC policy are: diagonal, within-aisle, across-aisle, and perimeter. L4: in addition to the variation of policy, the other factors that may affect the efficacy of an ABC storage include layout specification, ABC zone sizes, and real-time allocation methods.

2.4 Simulation Applications in Warehouse

This section reviews case studies of simulation applications that help improve decision making of warehouses.

Some of the simulation applications for warehousing are based on Digital Twin (DT), In 2002, Michael Grieves formally introduced the Digital Twin concept aimed at Product Lifecycle Management [23]. Grieves defined Digital twin as a virtual representation of the real system [22]. The main characteristics of a Digital Twin is its ability to exchange real-time data with the real counterpart with the possibility to interact or control the real system [65][17][19]. The case studies with Digital Twin applications utilise the real-time characteristics of Digital Twin for improved decision making.

The research in [39] outlines an architecture (shown in Figure5) for a warehouse Digital Twin that focuses on simulation. It divides types decision support services into three categories for a Digital Twin. This architecture is similar to our model depicted in Figure 2 with decision support services. The services from the Figure 5 is based on a DT while our model is based on a data warehouse. With the DT architecture, a DT was designed and tested for a warehouse to support decisions on the allocation of employees during shifts in different areas and it showed the benefit of using this DT is that it enabled inexperienced planners to plan and learn more effectively in order to reduce total man-hours without orders being delayed.

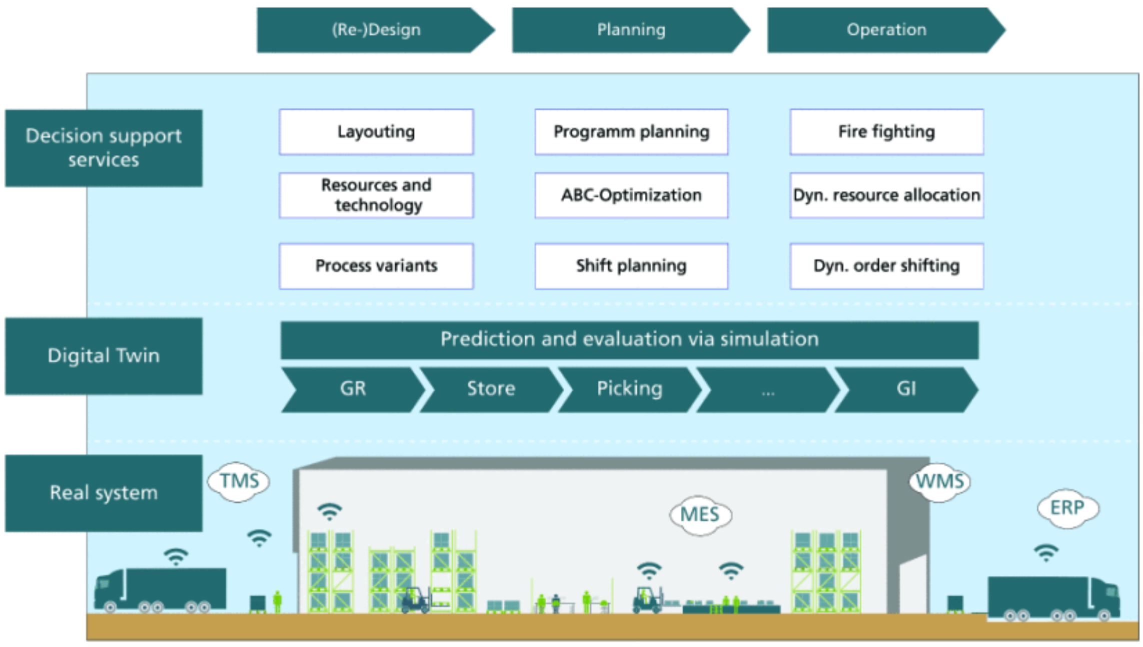


FIGURE 5: Application Architecture of a Digital Twin in Warehousing[39]

A DT was built for a warehouse which handles 10,000 to 20,000 orders per day [35]. The aim of the implementation is to assign an optimal number of workers to each zone in an automated goods-to-person setting. It is done by forecasting the number of orders for each zone from the simulation tool. The tool then prescribes the picking time and number of workers needed for the next day. Users can decide whether to take the advice from the tool. The author noted that the order picking process is complex since it is subject to individual decisions by human actors. Because of the complexity, deciding on the level of detail that should be modelled in the simulation environment of a DT remains a challenge. These factors affect the accuracy of the model and thus its performance. The next step to improve the model is to estimate the quality of the decision making support with the use of cost functions, which are used in a feedback loop to tune the algorithm.

In an automated storage facility, a DT prototype [44] was designed for optimising the packing and storage process. 3D models of the packing station and the storage are made in the open source engine called Unity3D. The DT retrieves data from sources including sensors through USB, Ethernet, and serial port connections, incoming and outgoing goods, and so forth. It optimises the packing, storing, and picking processes by calculating the best strategy for packing and put-away, and instruct the physical system to perform them. The main limitation of the proposed DT framework is that it does not examine the impact of random or unforeseen physical disruptions on the physical systems. It is therefore essential to predict and handle bottlenecks and deadlocks in the future. This is an example of how simulation can optimise the real-time operations in a warehouse. Users can feed in specific workload information into the model and it automatically instructs the system to perform the work with optimised handling processes.

A study [20] used a simulation to assign an optimal number of workers to different areas of warehouses for different types of works. Every day there are thirty one workers, each with ten possible skills. The skills are the constraint for the system because not everyone is suited for a given task. Gamma distribution was used to model the time consumed for each task. By using a combination of the O2DES (object-oriented discrete event simulation) framework and a random neighborhood search method, the model determined the best number of workers for each task and achieved significant improvement inbound and outbound service productivity. However, the study did not model any physical aspects of the warehouse.

Another study[42] presented a data-driven discrete-event simulation modeling approach to serve as an analysis tool for determining the appropriate mix of bulk and rack storage locations to utilize warehouse space effectively. The model evaluates system configurations including the number of racks and storage locations, the number of bulk lanes and lane depth, etc. This approach should help design a suitable layout for a warehouse.

Table 2 summarises the simulation applications reviewed in warehousing. In addition to simulation applications for storage policy, we found several applications help improve assignment (who to what problem) and allocation (who and when problem) of workers in warehousing processes; this is one of the reasons we decided to implement a resource monitor function for workers in our Storage Simulator to measure the impact of scenarios where assignments and allocation of workers are different.

Decision Factors	**References**
Worker Assignment and Allocation	[20, 35, 39, 80]
Process Optimisation	[44] (automated workload distribution)
Layout Optimisation	[42]
Storage Policy	[11, 15, 30, 45, 46, 50, 62, 73, 75, 76]

TABLE 2: Reviewed Simulation Applications in Warehousing

This section addressed L6: Use cases or features that a simulation application can offer to help decision making in a warehouse. It is mainly the assignment and allocation of workers. Few studies presented use cases for process optimisation or layout optimisation.

2.5 Configuration of a Warehouse in a Simulation Model

When modelling a warehouse for simulation, there are a number of configurations to be considered. They are divided into Physical Assets, Picking & Put-away Processes, and Human Resources.

2.5.1 Physical Assets

Physical assets include storage, layout and equipment selection. Rack storage and drive-in storage are the two of the common types of a storage system. A rack storage (see Figure6) allows pallets to be accessed from the aisle and a drive-in storage allows a forklift to drive directly into the lane of stacked rows. It is a first-in, last-out method of storage (see Figure7) with advantage of utilising the space efficiently by eliminating picking aisles. A limitation is that each bay is usually dedicated to a single product type for picking efficiency.

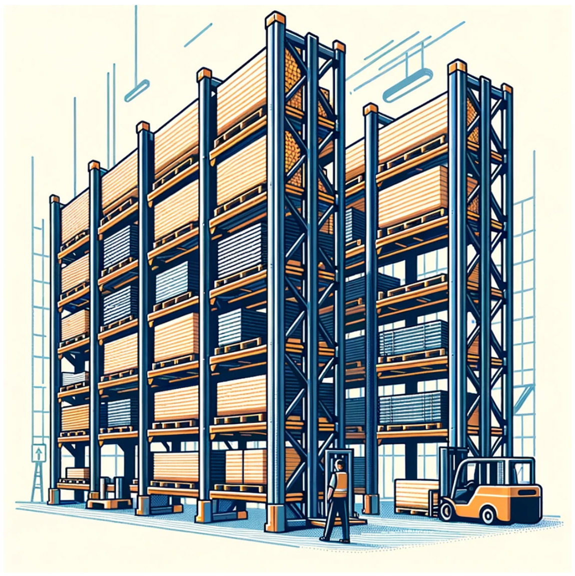


FIGURE 6: A Common Rack Storage System

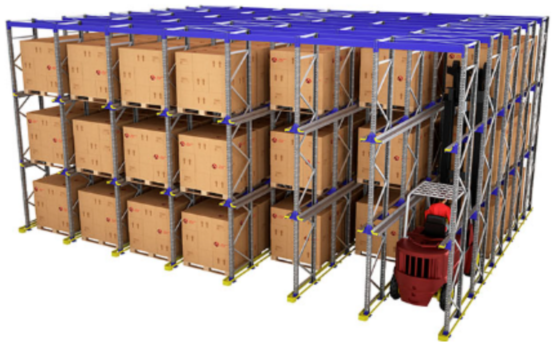


FIGURE 7: A Drive-in Storage System[51]

The layout specifies the placement of the assets and the physical dimensions of the physical assets. A good design can improve the efficiency of warehouse processes and maximise the value of the space[34]. There are traditional layouts [36][24]and new Flying-V layouts[34]. They will affect the routing choices of the workers. This is usually done during design phase of a warehouse. The layout should be built or modified to meet the type of operations, constraints and efficiencies of the warehouse. The dimensions of the zones, rack and aisle [24] should be suitable for operators, machines, and designated products. The appropriate zoning of the space should also ensure the smooth transition of products during each stage of handling.

Equipment selection [36] depends on the size of the warehouse, volume of products, types of products, etc. A warehouse that must handle tens of thousands of products every day, for example, would be better off with automated storage system. For a small and medium-sized warehouse with low volume, manual storage by labour is enough. In e-commerce warehouses, automated guided vehicles and even drones may be chosen to transport goods.

2.5.2 Picking and Put-away Processes

Put-away is the process of placing the received goods into storage locations. Sometimes it is equivalent to replenishment process where products are restocked. Picking is the process of retrieving the goods and preparing them for dispatch. Common picking methods include zone picking (divided into sequential and parallel), wave picking, and batch picking, etc. [36][24][3]. Picking is time consuming and can account for up to 60% of all warehouse time of labour activities and cost[49][31]. The efficiency of picking depends on the size of the warehouse, the picking system, the storage policy, the layout, and the routing strategies [10]. There are many routing methods. Basic routing methods such as traversal, midpoint, return, and combined policies are specified by a paper[63]. Most routing optimisation are based on Travelling Sales Man (TSP) problem and heuristics [36]. Route determination can be done by using algorithms to determine optimal routes for workers in the picking or put-away process.

2.5.3 Human Resources

The costs associated with labour can amount to about 80% of total expenses in warehouses [60]. Therefore, labour management should be prioritised for optimisation in warehousing. The description below highlights the main concerns of labour management.

Labour scheduling and assignment On the basis of incoming and outgoing orders, a suitable number of workers must be assigned to fulfil the total demand. In warehouses with dedicated zones, the number of workers for each zone should also be considered[39][35].

Labour performance measurement These are KPI metrics such as pick accuracy, processing time of receiving, storing, picking, and shipping, costs, and units/cases per man hour. [60][86] [58]They are often used for bench-marking to identify inefficient parts of material handling processes.

Balance utilisation and cost Normally high utilisation of workers in picking and put-away process is desired; this implies that workers are actively doing tasks, but the throughput rate or cycle time [40] could be negatively affected since the queue could increase in size and possibly create congestion or bottleneck.

2.6 Simulation Methods for Storage Policy

This section discusses the simulation methods for storage policies.

Discrete-Event Simulation DES models the operation of a system as a sequence of events in time. Each event occurs at an instant in time and triggers activities or processes in the simulated system[79]. A DES model is a system of queues and processes[8]. In a warehouse scenario, the arrival of an order is a discrete event that can be modelled by the rate of arrival per hour, by Poisson or gamma distribution, by an arrival schedule, etc. The system responds to each of these events by, for instance, triggering a process flow.

Agent-Based Simulation ABS models individual agents and their interactions with each other and their environment. ABS can provide a detailed, bottom-up perspective on the operation by modelling the actions and interactions of individual agents in the warehouse environment. An ABS model contains a simulated environment filled with autonomous entities [8].

Using an ABS approach, the following agents can be defined in a warehouse simulation.

① Workers: Warehouse workers are responsible for tasks such as picking, packing, and storing. Their efficiencies can be affected by their skill level and level of fatigue.

② Products: Although they are passive entities, products can be assigned characteristics such as size, weight, fragility, location where it is stored, customer owning them, and the order assigned to them.

③ Vehicles: Forklifts, Automated Guided Vehicles (AGVs), or drones in warehouses can be modeled as agents with their own navigation logic and loading capacities.

④ Storage Locations: Locations such as shelves or bins can be considered agents, with properties such as capacity, accessibility, or climate conditions.

A study [80] designed an ABS model similar to the above with autonomous vehicle and workers agents in an e-commerce warehouse. It is an ABS model with DES elements: the robots and workers pick orders generated by the system. After picking tasks are finished, the system checks the orders and summarises data. Experiments were conducted to measure throughput and costs. It found that costs could be reduced by having human operators and robots work together and assigning them to certain areas. It also found that routing choices can impact costs and that Largest Gap method performs better than with the S shape and Return method.

Hybrid simulation with DES and ABS addresses the shortcomings of DES, which lacks the ability to model the behaviour, properties, and states of an individual entity. An application is demonstrated by a case study in a radiology center [1] using AnyLogic software that models the processes and behaviours for patients, nurses, receptionists, doctors, and X-ray technicians. After running the model for a week, the length of stay for patients from the model was in line with the actual value from the real world. This model can be helpful to monitor resource utilisation and predict the length of stay for each patient given the arrival patterns of patients. Another paper[55] proposed a hybrid DES and ABS modelling approach to simulate the picking activity of a warehouse using FlexSim software. With the schedule generator and schedule control agents, picking tasks can be accepted or refused by the forklift agents based on their own logic. A state-of-art review [8]was conducted in 2018 and showed that hybrid simulation was a rapid developing field with many promising new opportunities; it reviewed 69 papers about hybrid simulation applications and there were 13 papers using a combination of ABS and DES. It also mentioned that AnyLogic was a promising tool for future hybrid simulation development, but cautioned that it has limitations of being a jack-of-all-trade software because it requires Java programming skills to implement a hybrid model.

A hybrid DES and ABS model would allow for both systematic view and bottom-up view of a warehouse. In a warehouse, resources (such as workers and forklifts), storage slot, and products could be modelled as agents with their own properties, states, and actions based on events, allowing us to track and manage them. When considering the travel distance of a forklift, for example, it can be measured regularly at a short interval; the total distance can be obtained by summing up the distances travelled by all forklifts.

This section addressed L5: the simulation methods that are suitable for a storage simulator. We look at the strength and weakness of DES and ABS simulation methods and review studies using these methods. A combination of DES and ABS would allow for a more accurate measure of the travel distance and picking time during the picking process.

2.7 Communication of Solutions to Warehousing Stakeholders

We investigated how forecasting and simulation solutions from case studies were communicated with or used by stakeholders in warehouse, and we found two main types of interfaces: (1) web-based or custom software applications. For example, a web-based application [39] was designed for dispatchers in a warehouse to view scenarios and help allocate employees' shifts. (2) Visualisation tools that has built-in features of data retrieval, drag-and-drop charts, or even data manipulation. Power BI and Tableau are examples: they allow for faster development without having to build a custom software or web application from ground up. Other studies presented results & recommendation but did not mention how the solutions are used by stakeholders. Table 3 summarises the findings.

Category	**References**
Web-based or custom software	[20, 35, 39]
Using a Visualisation Tool	[27]
Simply Presenting Results & Recommendation	[7, 11, 15, 30, 46, 50, 73, 76, 80]

Table 3: Communication with Warehousing Stakeholders

This section addressed L7: the communication methods of forecasting and simulation solutions in case studies with warehousing stakeholders. They include custom software, web applications, using visualisation tools such as Power BI, or presenting results & recommendations straightaway.

Chapter 3

Forecasting Service

This chapter describes the forecasting service for the daily picking demand. First, the workflow enabling the service is explained. Second, an analysis of the demand data is carried out. Third, the algorithms to be applied are introduced and discussed. Fourth, error metrics measuring the performance of the forecasting methods are explained along with the objective for our forecast error. Fifth, the process of applying algorithms for forecast is explained. Finally, the results are described. The objective, as will be shown in Section 3.4, is to achieve prediction error measured by MAPE to be less than 15% for an algorithm to be acceptable. Also, the MASE error should be less than 0.8 to be acceptable.

3.1 Workflow

Figure 8 outlines the workflow, summarising which data is retrieved and processed for the forecasting service. As seen from figure, the process of the data flow begins with data retrieval from the data warehouse with synthetic data provided by the company. It is done through a Python develop environment through the Spyder software. After retrieving data, it is cleaned and transformed to reflect the picking demand. The transformed data set is then used to train the forecasting algorithms; these details can be found in Section 3.2. After comparing performance of the algorithms, the most suitable one according to the objective is used to predict daily picking demand; the details are found in Section 3.5. The forecasts are visualised in Power BI.

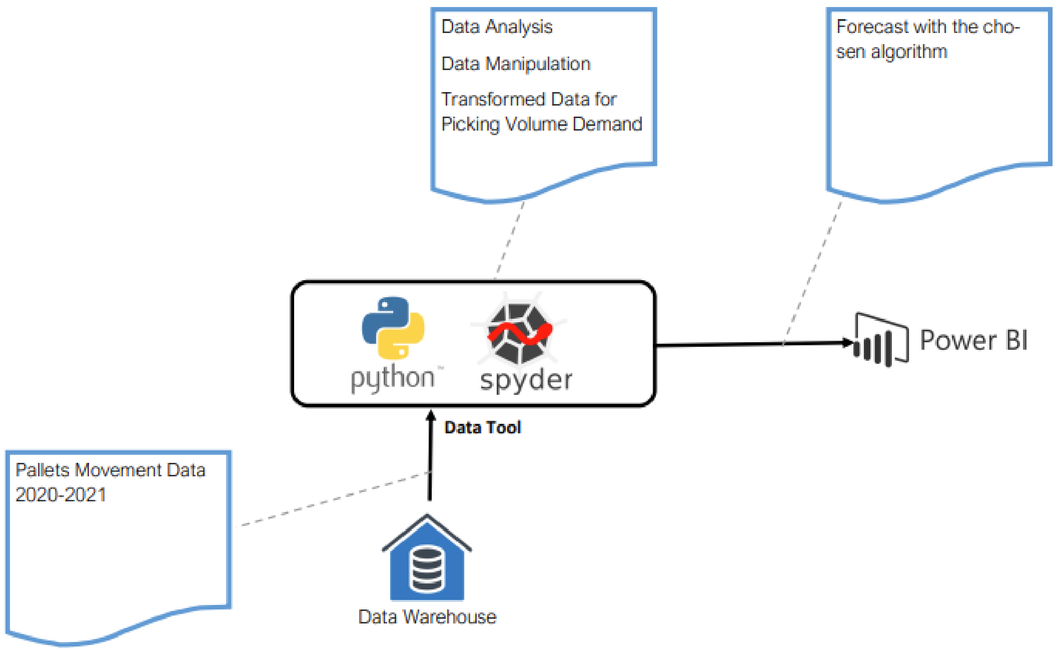

Data Analysis
Data Manipulation
Transformed Data for Picking Volume Demand
Forecast with the chosen algorithm
python
spyder
Power BI
Data Tool
Pallets Movement Data 2020-2021
Data Warehouse

Figure 8: Workflow for Forecasting Service

3.2 Data Analysis

As Table 4 shows, the original data contains the movement history of pallets starting from January 2020 and ending in June 2022 in a hypothetical warehouse that deals with bulk items. The products are stored in bulk on pallets (as Figure 9 shows) and the picking volume is treated as the number of pallets that are sent out for orders. To extract the outgoing workload demand, the number of pallets being moved out each day are summed up and aggregated for each day, as Figure 5 shows. This represents the daily picking volume each day in the warehouse. The aggregated data upon visual inspection, as Figure 10 shows, suggests a slight volume decrease in 2021 compared to the previous year. On any given day with picking activities, there were always three pickers.

Movement Date	**Movement ID**	**Code**	**Employee ID**	**Pallet ID**	**Product ID**
2020-04-11T00:00:00	17685	I	5	n/a	10007
2020-04-11T00:00:00	17881	I	5	n/a	10010
2020-04-11T00:00:00	17931	I	5	n/a	10003
2020-04-11T00:00:00	144044	O	6	10482	10010
2020-04-11T00:00:00	144134	O	6	10425	10003

Table 4: Table of The Pallet Movement History

Figure 9: Bulk Pallet

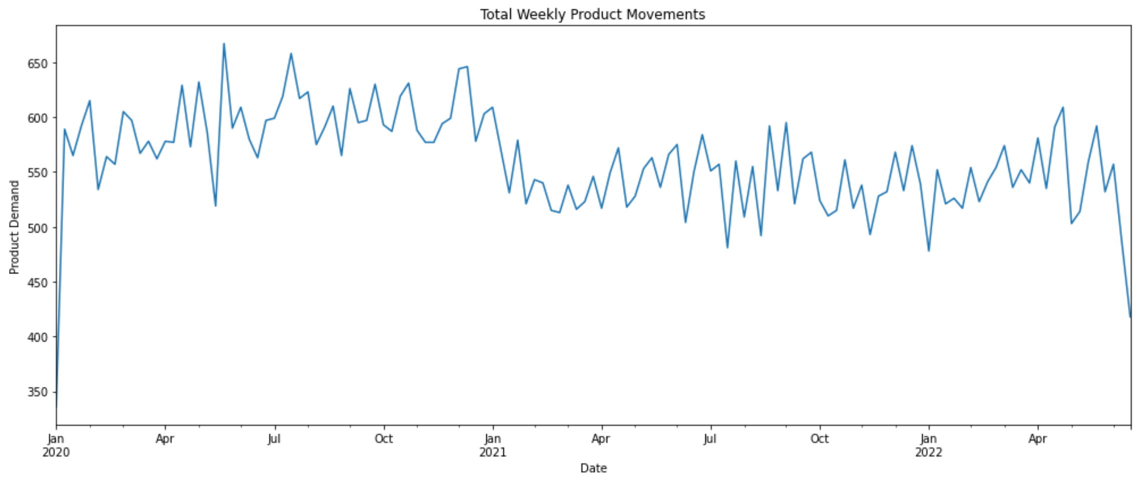


FIGURE 10: Weekly Demand Over Time

When we look at the daily picking volume on weekly basis, we found that in 2020 Tuesdays, Wednesdays, Thursdays, and Fridays follow a similar distribution pattern. On Mondays, there were less than 10 pallets being picked, as Figure 11 shows. Furthermore, in 2021 there were no more picking volume on Mondays, as Table 5 and Figure 12 show.

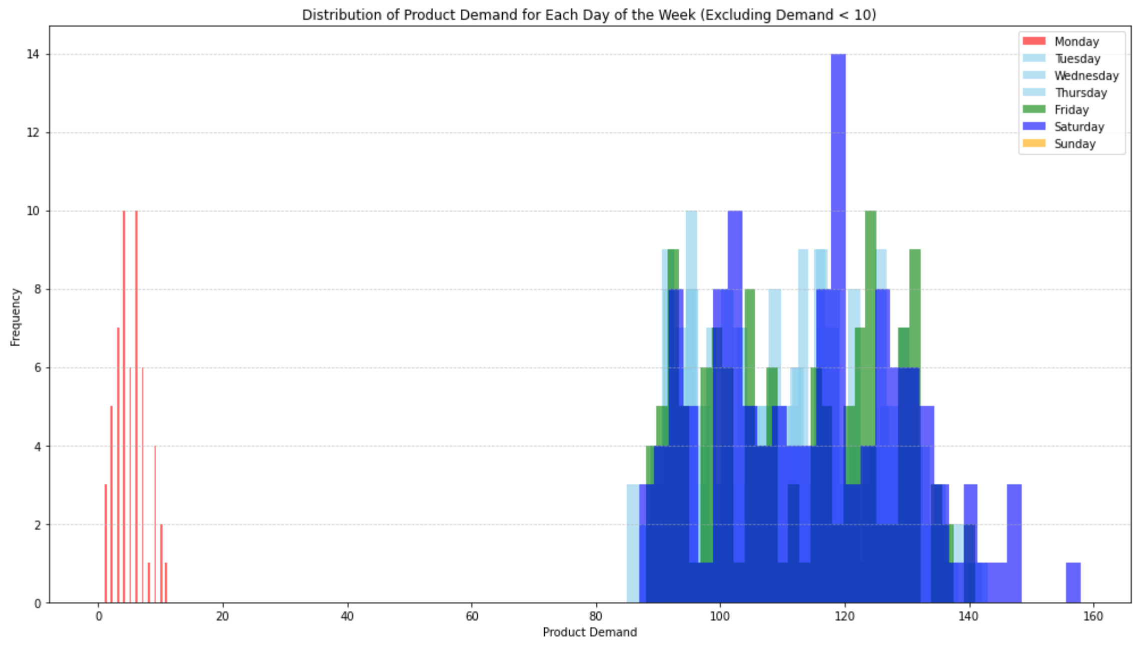


FIGURE 11: Weekly Demand Distribution

MovementDate	Count
2021-08-31	109
2021-09-01	111
2021-09-02	111
2021-09-03	130
2021-09-04	134
2021-09-05	0
2021-09-06	0
2021-09-07	97
2021-09-08	128
2021-09-09	102
2021-09-10	90
2021-09-11	104
2021-09-12	0
2021-09-13	0

Table 5: Aggregated Picking Volume

A seasonal decompose is performed, which is a method to help understand a time series[9], as Figure12 shows. A seasonal decompose is a useful method to break a time series into three components: trend, seasonal, and residual. The trend is the increasing or decreasing values in the series. The seasonality is the repeating short-term cycle in the series. The residuals (a.k.a the noise) are the random variations in the series. As seen from the decomposition, there is a seasonal pattern with some random variation of the data across the history.

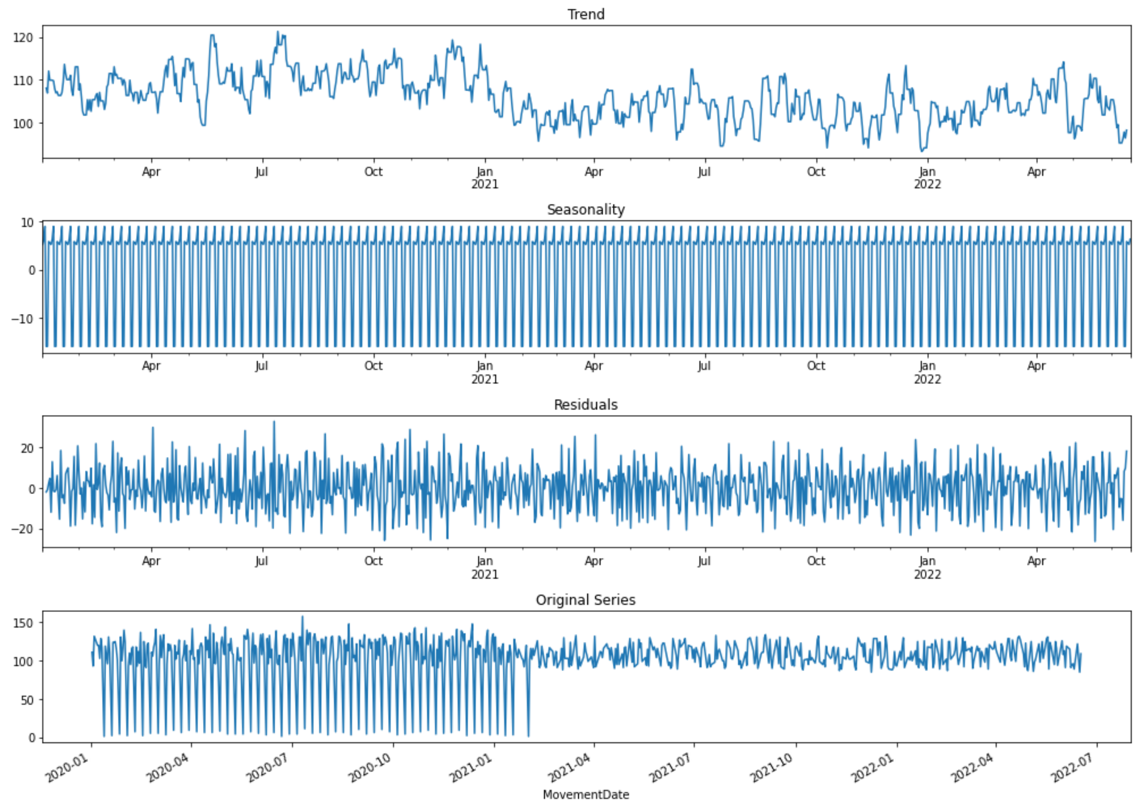


FIGURE 12: Seasonal Decompose

To clean the data, Sundays and Mondays are removed since there is no picking volume on these two days from 2021. With the cleaned data, we applied four forecasting methods as explained in the next section.

3.3 Forecasting Algorithms

In this section, we discuss the forecast algorithms to be applied to the demand data set. From the literature review we found three algorithms showing exceptional performance from studies [7, 21]: triple exponential smoothing, SARIMA, and LightGBM; so, they are going to be trained and tested on the data set. In addition, we used simple moving averages as the baseline algorithm to be compared to those three methods. The reason is to check whether those three algorithms could outperform the baseline model.

3.3.1 Simple Moving Averages

We used simple 3-day, 6-day, 10-day, 20-day moving average as a baseline models. The Simple Moving Average (SMA) is calculated by adding up a set of data points within a specific window size N, and then dividing the sum by the window size. The formula is as follows:

$$\text{Forecast}_{t+1} = \frac{1}{N}\sum_{i=0}^{N-1} X_{t-i}$$

Where N = 3, 6, 10, 20 respectively. The SMA is a simple and straightforward method to make forecast based on the moving averages in the past.

3.3.2 Triple Exponential Smoothing

Triple Exponential Smoothing (also called Holt-Winter method) can account for trend and seasonality. It captures three components in time series data: level, trend, and seasonality. There are two variations: Additive model and multiplicative model [26, 28, 81].

Additive method

$$\begin{aligned}
\hat{y}_{t+h|t} &= \ell_t + hb_t + s_{t+h-m(k+1)} \\
\ell_t &= \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m},
\end{aligned}$$

Multiplicative method

$$\begin{aligned}
\hat{y}_{t+h|t} &= (\ell_t + hb_t)s_{t+h-m(k+1)} \\
\ell_t &= \alpha\frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma\frac{y_t}{(\ell_{t-1} + b_{t-1})} + (1-\gamma)s_{t-m}
\end{aligned}$$

Where [18]:

Seasonal Component Adjustment: The term k is derived from the formula $\frac{(h-1)}{m}$, where h represents the forecast horizon, and m represents the number of seasons per period. The expression $s_{t+h-m(k+1)}$ ensures that the seasonal component is always derived from the corresponding season in the previous cycle, thus maintaining seasonal consistency in the forecasts.

Level Component (l_t): The level component, denoted by l_t, is essentially a weighted average. The weight, α, is applied to the seasonally adjusted observation while the remainder of the weight, $1-\alpha$, is applied to the non-seasonal forecast from the previous period. This process helps adjust the level component based on the most recent observation while considering past forecasting information.

Trend Component(β_t): is the smoothing parameter for the trend, which determines how quickly the trend component responds to changes in the underlying trend of the data. The value of β lies between 0 and 1; a higher value of trend allows the trend component to adapt more quickly to changes in the trend, while a lower value makes the adaptation slower.

Seasonal Component (s_t): The seasonal component, s_t, is computed as a weighted average with a weight of γ. The weight is applied between the current seasonal index and the seasonal index of the same season from the previous cycle. This mechanism helps update the seasonal component based on the latest seasonal patterns while considering past seasonal information.

In the additive method, the forecasted value is the sum of the level, trend, and seasonal components. The equations adjust the level, trend and seasonal components based on the observed data and previous estimates.

In the multiplicative method, the forecasted value is the product of the level and trend components, and the seasonal component. Similar to the additive method, the equations adjust the level, trend, and seasonal components based on the observed data and previous estimates, but with the seasonal component being a ratio rather than a difference.

Table 6 explains the two models.

Component	**Additive**	**Multiplicative**
Trend	Trend is linear. Difference between periods is roughly constant.	Trend is exponential. Increases or decreases at an increasing rate. [69]
Seasonality	Seasonal variations are roughly constant. Magnitude of seasonality doesn't change. [69, 83]	Seasonal variations change proportionally. Magnitude of seasonality changes with series level. [4, 83]

Table 6: Comparison of Additive and Multiplicative Trend and Seasonality in Holt-Winters Method

3.3.3 SARIMA

SARIMA is an extension of ARIMA that includes seasonal parameters. So, we used it to account for the seasonality. The ARIMA model is given as:

$$\text{ARIMA}(p, d, q)$$

where:

$$\begin{aligned} p &: \text{Order of the autoregressive (AR) term.} \\ d &: \text{Degree of differencing.} \\ q &: \text{Order of the moving average (MA) term.} \end{aligned}$$

Autoregressive Component (p)

The AR component assumes that the current value of the series is a linear combination of its previous values.

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$$

where $\phi_1, \phi_2, \ldots$ are AR coefficients.

Moving Average Component (q)

The MA component represents the influence of the previous white noise error terms on the current value.

$$y_t = \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

where $\theta_1, \theta_2, \ldots$ are MA coefficients.

Integrated Component (d)

This component makes the series stationary by subtracting. Stationary means the mean, variance, and auto-covariance of the series are constant. It is done by subtracting the current observation by its previous value. The series is subtracted d times. For example, the equation for a differenced series Y_t at lag-1 is:

$$Y_t = X_t - X_{t-1}$$

ARIMA Prediction

Assuming the series is differnced and stationary, the prediction becomes a combination of Auto Regressive and Moving Average components [2, 6]:

$$y_t = \alpha + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

Where

$$\alpha$$

is a constant.

SARIMA

The SARIMA model is given as:

$$\text{SARIMA}(p, d, q) \times (P, D, Q)_m$$

where:

$$\begin{aligned} p &: \text{Order of the autoregressive (AR) term.} \\ d &: \text{Degree of differencing.} \\ q &: \text{Order of the moving average (MA) term.} \\ P &: \text{Order of the seasonal autoregressive (SAR) term.} \\ D &: \text{Degree of seasonal differencing.} \\ Q &: \text{Order of the seasonal moving average (SMA) term.} \\ m &: \text{Number of periods per season.} \end{aligned}$$

P, D, Q, and m are introduced. These are seasonal components.

Seasonal Components

Seasonal AR and MA components (P and Q) account for seasonality, with lags (m) based on the season (for example, m = 12 for monthly data with yearly seasonality). Integral component means D times for seasonal differencing. The Seasonal AR component (P) is mathematically represented as follows:

$$Y_t = \Phi_1 y_{t-m} + \Phi_2 y_{t-2m} + \cdots + \Phi_P y_{t-Pm}$$

Seasonal Moving Average Component (Q) The Seasonal MA component is mathematically represented as follows:

$$Y_t = \Theta_1 \varepsilon_{t-m} + \Theta_2 \varepsilon_{t-2m} + \cdots + \Theta_Q \varepsilon_{t-Qm}$$

SARIMA prediction

The SARIMA model's prediction, which extends ARIMA by including seasonal components, is as follows:

$$\begin{aligned} y_t = \alpha &+ \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} \\ &+ \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} \\ &+ \Phi_1 y_{t-m} + \Phi_2 y_{t-2m} + \cdots + \Phi_P y_{t-Pm} \\ &+ \Theta_1 \varepsilon_{t-m} + \Theta_2 \varepsilon_{t-2m} + \cdots + \Theta_Q \varepsilon_{t-Qm} \\ &+ \varepsilon_t \end{aligned}$$

3.3.4 LightGBM

LightGBM is based on gradient boost trees [78]. It is an ensemble model, which means it builds multiple decision trees and aggregates their outputs for better prediction accuracy. Specifically, LightGBM constructs an ensemble of decision trees for tasks like classification and regression prediction. The common and most important parameters [5] are summarised in Table 29 in the appendix.

Two methods are available in LightGBM to determine the importance of feature: gain and split [66].

Gain Each feature has its importance level. Gain is the relative contribution of a feature in a particular tree. A higher gain means the feature contributes more to making better predictions.

Split Split for LightGBM calculates the relative count of times a feature occurs in all splits of the model's trees. A downside to this method is that it is subject to more bias when there are many categorical features.

3.4 Error Metrics and the Objective

Error metrics measure the accuracy of the predictions by comparing them with and the actual values. For example, A higher value of Root-Mean-Square Error means that the prediction is less accurate. In this section five error metrics are explained. Table 7 shows the methods of measurement with description of each method. We explained why we chose RMSE, MAPE, and MASE as the metrics for our evaluation of forecasting methods.

Error Measure	**Formula**	**Description**
Mean Absolute Error (MAE)	$\frac{1}{n}\sum_{i=1}^{n} \lvert y_i - \hat{y}_i \rvert$	MAE measures the average magnitude of the errors between the predicted values and the actual values.
Mean Squared Error (MSE)	$\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$	MSE squares the errors before averaging them, placing more weight on larger errors.
Root Mean Squared Error (RMSE)	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$	RMSE is the square root of the mean squared errors. It has the same unit as the output, which can be useful for interpretation.
Mean Absolute Percentage Error (MAPE)	$\frac{1}{n}\sum_{i=1}^{n}\left\lvert \frac{y_i-\hat{y}_i}{y_i} \right\rvert$	MAPE expresses errors as a percentage; it is the error percentage of the prediction compared to the actual values.
Mean Absolute Scaled Error (MASE)	$\frac{\frac{1}{n}\sum_{i=1}^{n} \lvert y_i-\hat{y}_i \rvert}{\frac{1}{n-1}\sum_{i=2}^{n} \lvert y_i-y_{i-1} \rvert}$	A MASE value less than 1 indicates that the forecast is better than a naive forecast, while a value greater than 1 indicates the forecast is worse.

TABLE 7: Summary of common error metrics used in forecasting with machine learning.

where:

- y_i: Actual value
- $\hat{y}_i$: Predicted value
- n: Total number of observations
- i: Index of each observation

As described above, MAE, MSE, RMSE, and MAPE are straightforward metrics. MASE needs more explanation: its numerator calculates the Mean Absolute Error (MAE) between the actual values y_i and the predicted values $\hat{y}_i$ over n observations. The denominator is a naive baseline model that computes the mean absolute error of a naive forecast which simply takes the previous observed value y_{i-1} as the forecast for the current value y_i. This ratio compares the error of the predictions to the error of a naive baseline model. Here are the implications of the MASE value:

- **MASE $<$ 1**: The model has lower error compared to the naive model. This is generally indicative of a good model.
- **MASE $=$ 1**: The model's error is equal to that of the naive model.
- **MASE $>$ 1**: The model has higher error compared to the naive model. This is generally indicative of a poor model.

MAE and RMSE are in the original units of the data, making them more interpretable than MSE. MAPE and MASE provide percentage-based and scale-independent error metrics, respectively, offering a relative measure of accuracy.

Judging from the characteristics of the error metrics. We chose MAPE, MASE, and RMSE to measure the performance of the algorithms. The objective is to achieve MAPE error of less than 15% for an algorithm to be acceptable. MASE should be less than 0.8 to be acceptable. MAPE and RMSE are used to measure the performance among the algorithms.

3.5 Forecast and Observation

In this section, we describe the application of the forecasting algorithms including Simple Moving Averages, Triple Exponential Smoothing, SARIMA, and LightGBM. The prediction and test data are compared to each other with graphs and with the error metrics: RMSE, MAPE, and MASE. The training data is the demand data from the start date of the data set up until 30 days before the end date. The test set is the last 30 days of the data set. This applies to all algorithms.

3.5.1 Simple Moving Averages

We used simple 3-day, 6-day, 10-day, 20-day moving average as a baseline models. Figure 13 shows the comparison of test data and predicted values.

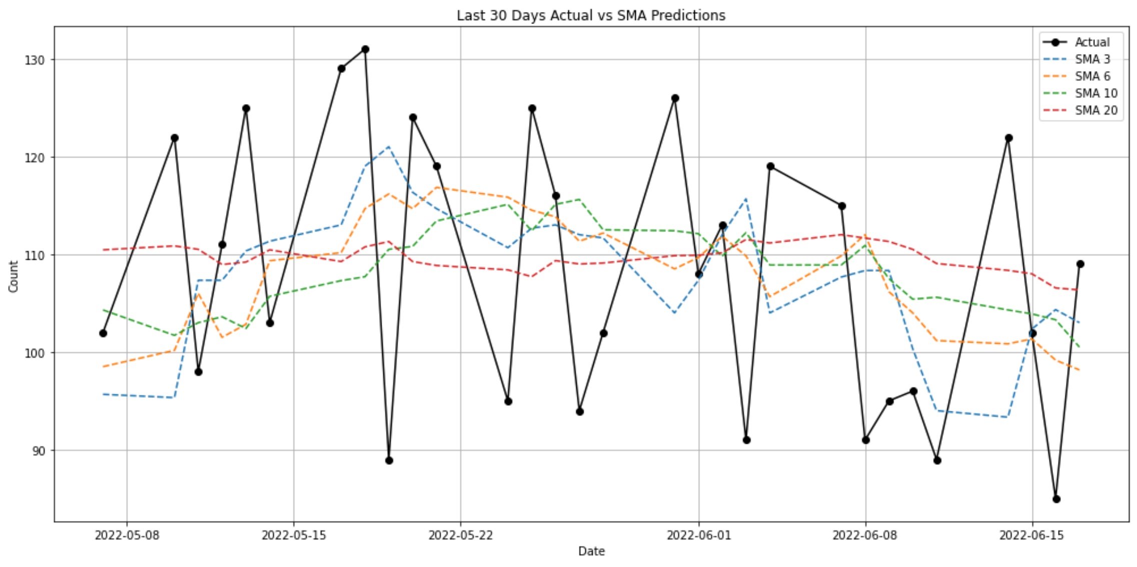


FIGURE 13: Test Set vs. Simple Moving Average (SMA) Forecast

SMA	MAPE (%)	RMSE	MASE
SMA_3	11.53	14.81	0.6689
SMA_6	11.45	14.13	0.6658
SMA_10	11.81	14.46	0.6867
SMA_20	11.86	13.95	0.6817

TABLE 8: Error metrics for different Simple Moving Averages (SMA)

All moving average methods yielded low MAPE, lower than 15% and MASE less than 0.8. From graphical inspection only the 3-day moving average roughly captured the peak and trough of the demand. The other moving average methods reflect the overall trend, but did not capture well the fluctuations like 3-day moving -average does. The result suggests that the 3-day SMA might be suitable to estimate the demand in the short term for this data set.

3.5.2 Exponential Smoothing

Grid search [72] is a practical method to find optimal parameters for Exponential Smoothing methods. Two grid searches as described below were conducted to find optimal parameters that minimises RMSE.

① **Grid Search for the Linear and Seasonal Model**:

- Different trends (‘additive‘ or ‘multiplicative‘) are tested using the ‘ExponentialSmoothing‘ model.

- For each trend, the model is trained on the training data, and the RMSE between the forecasted and actual test data is computed.
- An iteration within the trend configuration is carried out with seasonal configurations.
- The 'seasonal_periods' parameter is fixed at 5 (since Mondays and Sundays are removed).
- If a lower RMSE is found, the 'best_holt_rmse' and 'best_holt_params' are updated.

② **Parameter Optimization using a Coarser Grid**: A coarser grid implies a less granular, broader step size between the values in the grid. It covers a wider range of values but with less precision. The search setting is as follows:
Trend and Seasonality: additive trend, additive seasonality
Seasonal period: 5 days (weekly seasonality with Sundays and Mondays removed).
Searched parameters: α,β,γ, each ranges from 0.01 to 0.5. 10 sample values equally spaced between this range are used, specifically: 0.01, 0.065, 0.12, 0.175, 0.23, 0.285, 0.34, 0.395, 0.45, 0.5.

- This grid search is performed to optimize the smoothing parameters: 'alpha', 'beta', and 'gamma'.
- The Root Mean Squared Error (RMSE) is used as the evaluation metric.
- Lower RMSE values update the 'best_rmse', 'best_alpha', 'best_beta', and 'best_gamma' variables.

The search yield the following parameters that minimizes RMSE error:

- α for level smoothing: 0.01
- β for trend smoothing: 0.01
- γ for seasonal smoothing: 0.28

The search was resource and time-consuming for a regular computer. We attempted to increase the number of sample values from 10 to 15, the search would not complete for half an hour.

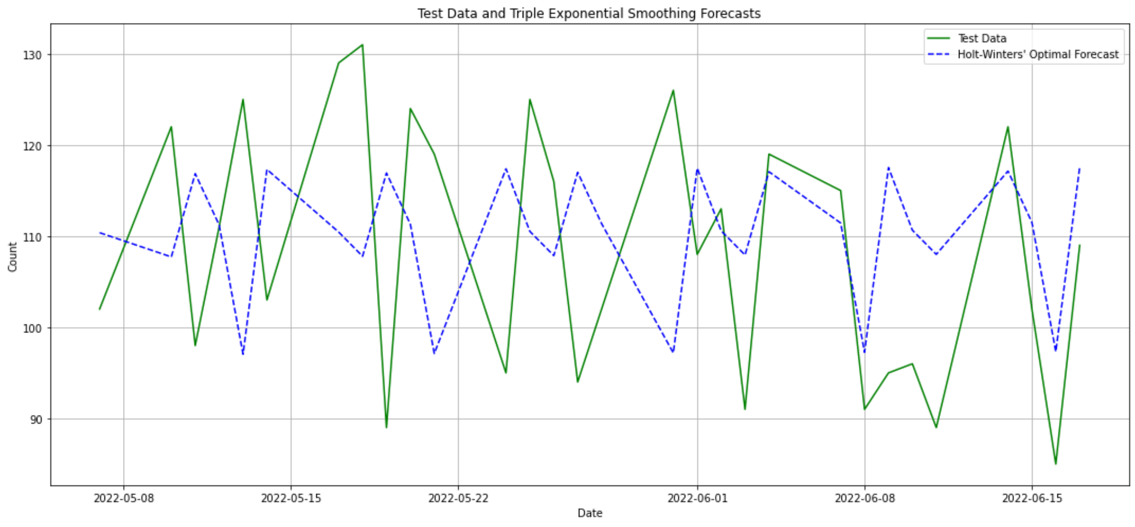


Figure 14: Test Set vs. Triple Exponential Smoothing Forecast

Metric	**Test Set**
MAPE	13.4%
MASE	0.7823
RMSE	16.33

Table 9: Triple Exponential Smoothing Forecasting Errors on the Test Set

Judging from MAPE and RMSE, triple exponential smoothing performed worse than moving average methods, but from the graphic review it appears to be capturing the fluctuations in the last one-third of the test set.

3.5.3 SARIMA

To estimate the optimal parameters of p, d, q, P, D, Q parameters, a grid search is a practical approach [9, 68]. The AutoARIMA library in Python was imported and applied to search the parameters. AIC (Akaike information criterion) was used as error metrics by AutoARIMA by default. The formula for AIC is given by:

$$\text{AIC} = 2k - 2\ln(L)$$

where k is the number of estimated parameters in the model and L is the likelihood, and $\ln(L)$ is the maximum log-likelihood estimate for the model [38, 59]. Compared to RMSE, AIC seeks a balance between the fit of the model and its complexity. It penalizes models with more parameters, aiming to balance complexity and accuracy[32].

The search setting is as follows:

- **seasonal**: This parameter is set to `True` to indicate that the data has a seasonal component.
- **m**: Specifies the seasonal period; in this case, it is set to 5, indicating a seasonal cycle every 5 observations (since Sundays and Mondays are excluded).
- **trace**: Set to `True`, this parameter enables the output of convergence information during the fit.
- **error_action**: The value `'ignore'` tells the function to ignore errors and continue the search.
- **suppress_warnings**: This parameter is set as `True`, suppressing warnings generated during the fit process.
- **step-wise**: This parameter is set to `True`, enabling a step-wise search over the model parameters which can significantly speed up the search process by evaluating only a subset of models.

A SARIMA configuration of (5,1,1), (1,0,1,6) results in the lowest AIC, with the following results in Table 10:

Metric	**Test Set**
MAPE	12.51%
MASE	0.728
RMSE	15.37

TABLE 10: SARIMA Forecasting Errors for Training and Test Sets

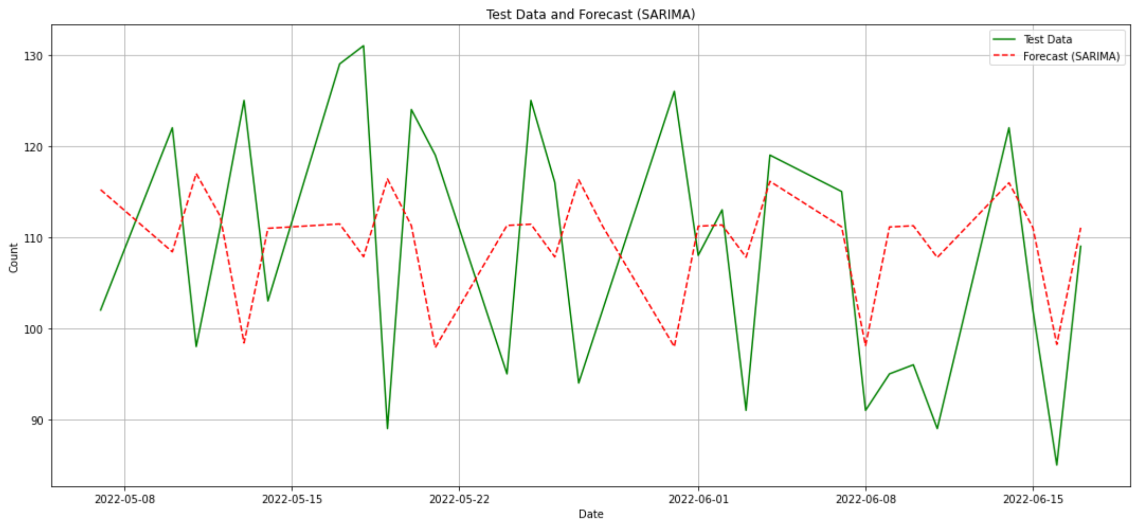


FIGURE 15: Test Set vs. SARIMA Forecast

SARIMA model appears to capture the peaks and troughs in the last one-third of the test set, and it has lower MAPE and RMSE than Triple Exponential Smoothing.

3.5.4 LightGBM

LightGBM requires feature engineering for the data to be trained. We derived the following time-related features [7] [27] from the time series. The features are explained and shown in Table 11.

Feature	Description
year	The year of the movement date
month	The month of the movement date
week	The week number of the movement date
day	The day of the month of the movement date
day_of_week	The day of the week of the movement date (0=Monday, 6=Sunday)
holiday	Binary feature indicating if it's a holiday in the Netherlands (1 for holiday, 0 otherwise)
lag_1	Value of the demand from 1 day prior
lag_3	Value of the demand from 3 days prior
lag_5	Value of the demand from 5 days prior
lag_6	Value of the demand from 6 days prior
rolling_mean_3	Rolling average of the demand over the past 3 days
rolling_mean_5	Rolling average of the demand over the past 5 days
rolling_mean_6	Rolling average of the demand over the past 6 days
rolling_mean_10	Rolling average of the demand over the past 10 days
rolling_mean_15	Rolling average of the demand over the past 15 days
rolling_mean_20	Rolling average of the demand over the past 20 days
rolling_mean_60	Rolling average of the demand over the past 60 days

TABLE 11: Features Used in The LightGBM Model

To train the model, we chose a range of parameters [7] to form a grid, which is used for a grid search to find the best performing configuration. With the grid parameters shown in Table 12, training and testing were done using loops to find the configuration that results in the least RMSE error. The best configuration with the least RMSE error is found to be the parameters indicated in Table 13.

By using the best configuration, forecast was applied to the test set, and the results are shown in Figure 16. Like the other algorithms, predictions of LightGBM in the

Parameter	**Values**
objective	regression
boosting	gbdt
metric	rmse
learning_rate	0.10
num_leaves	11, 21, 31, 41
max_depth	6, 8, 10, 12, 14, 16
min_data_in_leaf	2, 4, 5, 10, 15, 20
bagging_fraction	0.9
bagging_freq	10
early_stopping_rounds	50
verbose	-1
seed	1

TABLE 12: Grid Parameters

Parameter	**Value**
bagging_fraction	0.9
bagging_freq	10
boosting	gbdt
early_stopping_rounds	50
learning_rate	0.1
max_depth	16
metric	rmse
min_data_in_leaf	4
num_leaves	41
objective	regression
seed	1
verbose	-1

TABLE 13: The Best Configuration found for the LightGBM Model

short-term (the first 7 days of the test set) did not line up with the actual values. However, it did capture the direction of the fluctuation in the last one-third of the test set.

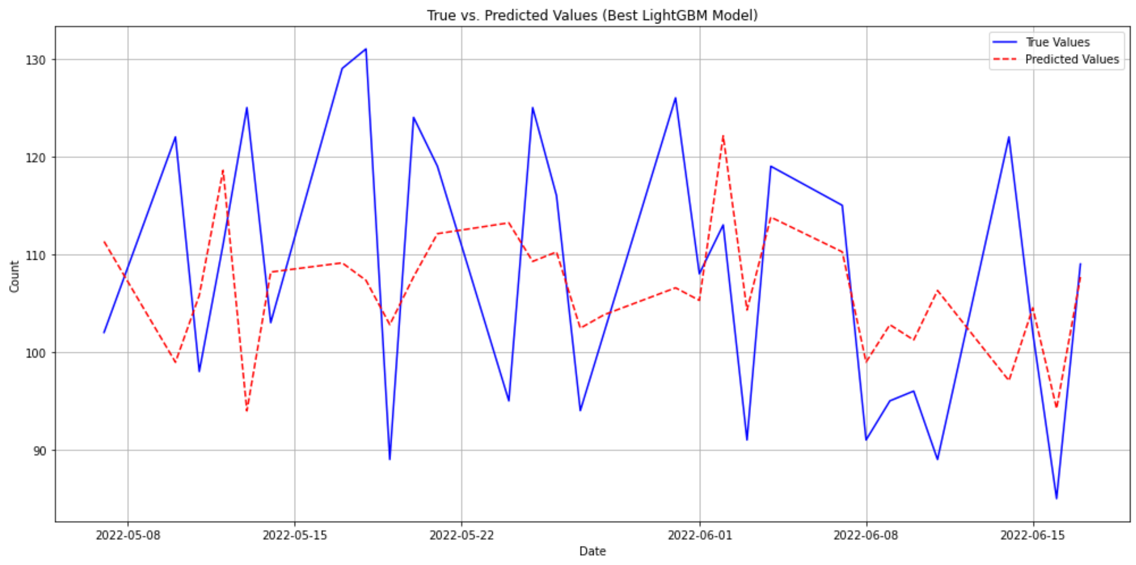


FIGURE 16: Predictions from LightGBM

The error metrics of the forecast from the LightGBM model are shown in Table 14.

Metric	**Test Set**
MAPE	10.46%
MASE	0.69
RMSE	13.82

TABLE 14: LightGBM Forecasting Errors for Training and Test Sets

The features' importance can be obtained by calling a built-in function from the LightGBM library: Figures 17 18 show the importance of features based on splits and gains respectively. The top 3 features based on split importance, are lag_1, lag_3, and day; Split importance measures how frequently a feature is used to split data across all trees. A feature with higher split importance is used more often in making binary splits in trees[48]. Based on gain importance, the most important parameters are "day of week", lag_1, and lag_3. The gain is a measure of the improvement in accuracy from a feature in its branches [14]. In other words, it's a sum of the decreases in loss function brought by that feature.

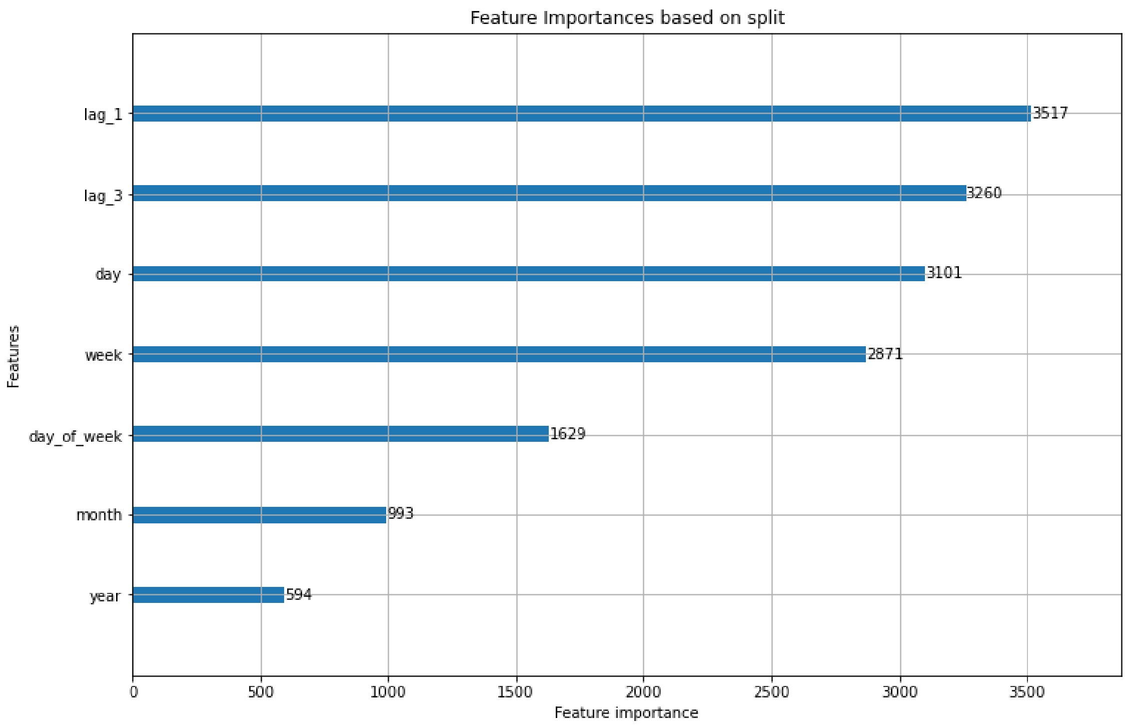


FIGURE 17: Feature Importances Based on Split

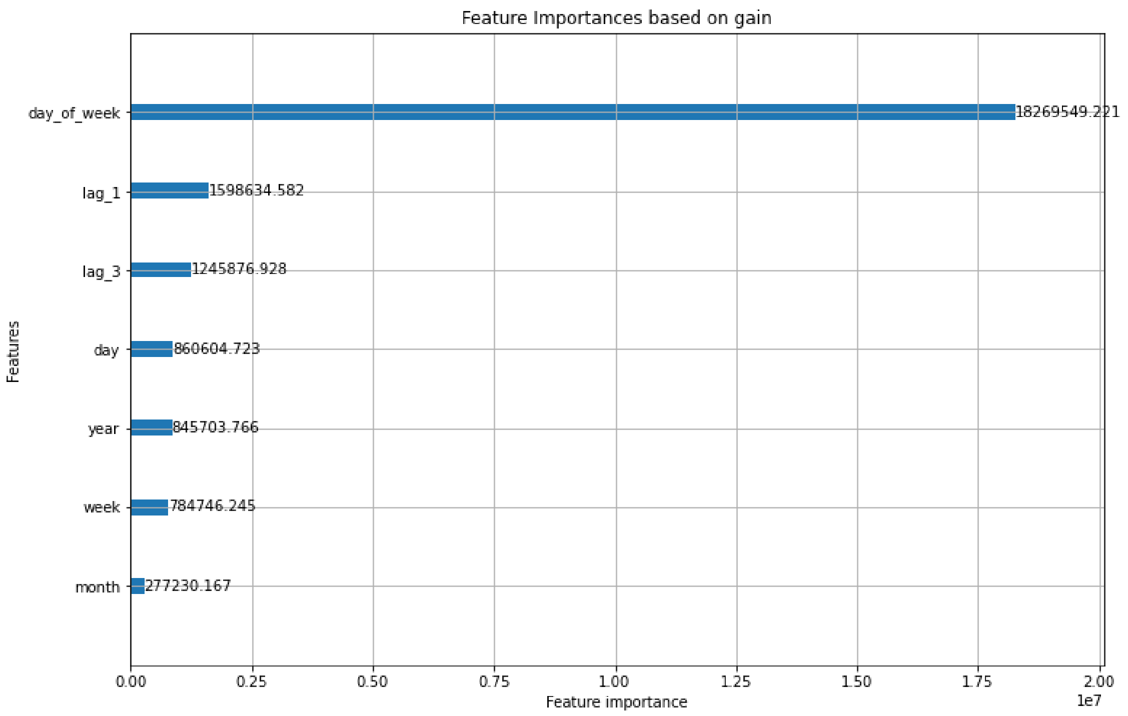


FIGURE 18: Feature Importances Based on Gain

3.6 Summary of Forecasting Service

We noted that each day there were always three workers for picking. So, we could not establish a correlation between daily number of pallets to be picked and how many workers are needed on that day. We proceeded to predicting the daily picking demand measured by number of pallets being picked. Table 15 lists the performances of the algorithms with the test set. MAPE from all algorithms achieved lower than 15%, and their MASE are lower than 0.8. When comparing RMSE, the SMA outperformed the other algorithms. LightGBM did not outperform others in the short-term (1 to 7 days) forecast. Notably, SARIMA was able to capture both the fluctuation and magnitude of data at the last one-third of the test set while other algorithms did not. Therefore, SARIMA was used as the forecast model to predict the daily picking volume and the results are used for visualisation.

Algorithm **Metric**	**MAPE (%)**	**RMSE**	**MASE**
LightGBM	10.46	13.82	0.69
SARIMA	12.51	15.37	0.728
Triple Exponential Smoothing	13.4	16.33	0.7823
SMA_3	11.53	14.81	0.6689
SMA_6	11.45	14.13	0.6658
SMA_10	11.81	14.46	0.6867
SMA_20	11.86	13.95	0.6817

TABLE 15: Forecasting Errors of the Algorithms for the Test Set

Chapter 4

Storage Simulator

This chapter shows the design process of the Warehouse Simulator with the following steps: First, the workflow is explained. Second, the requirements for the Storage Simulator are specified. Third, the conceptual model is illustrated with tables, state machine model, and process diagrams. Fourth, we listed a number of available simulation software and explained why AnyLogic was chosen. Fifth, KPIs about travelling distance, picking time, worker utilisation, and total waiting time are specified. Sixth, we explained the layout, data model, and how the KPIs are measured in AnyLogic.

4.1 Workflow

This section describes the workflow of Storage Simulator, as Figure 19 shows. First the movement history of pallets (as Table 4 shows) are retrieved from the data warehouse (provided by the company) to the Python development software Spyder, which then processed it and classify A, B, and C products by their frequency of being picked. The pick list data (used for experiments) was derived from the movement history. With a pick list spanning two days, the simulation model ran experiments and produced results for storage policy and resource monitor. The results are formatted by Spyder again and manually sent over to the visualisation tool Power BI.

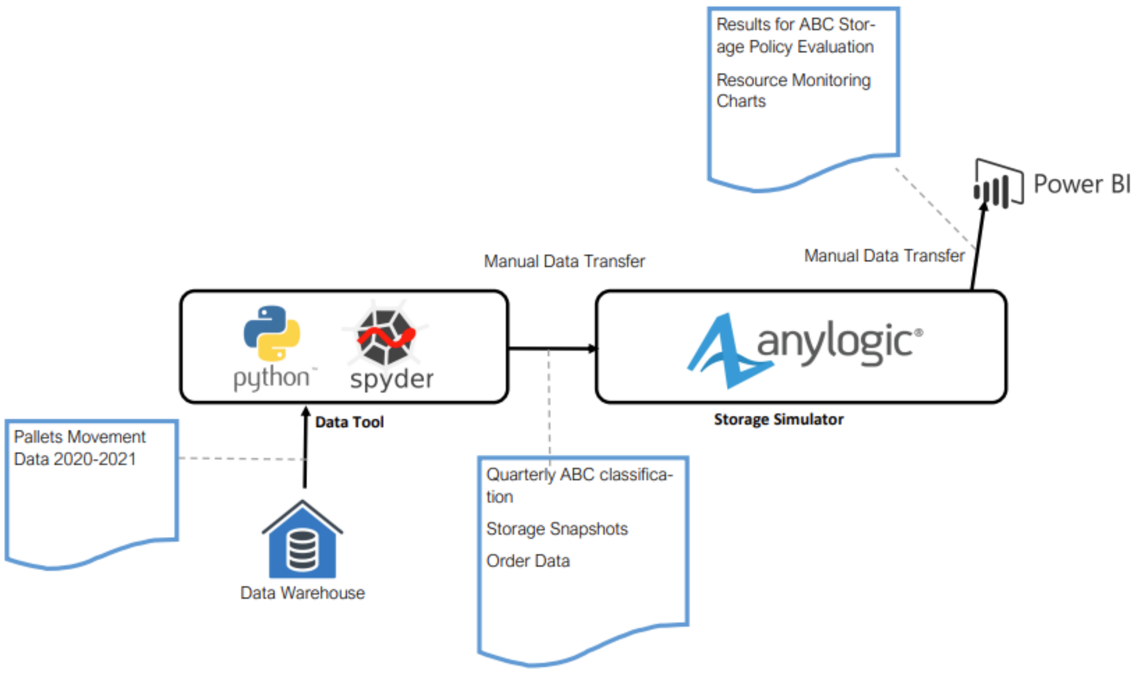


FIGURE 19: Workflow for Storage Simulator

4.2 Requirements

The simulation model is used to fulfil the following requirements to provide decision-support data for warehousing stakeholders:

① Measure travelling distance and time during the picking process.

② Compare the impact of variations of ABC storage policies on the travelling distance and picking time during the picking process.

③ Monitor the utilisation of workers, waiting time and bottlenecks/congestion resulting from changing the number of available workers given the same amount of the task.

4.3 Conceptual Model

The conceptual model consists of two process models (see Figures 21 ??) and a state machine model (see Figure 23) . The framework proposed in [61] was used to explain the conceptual model with aspects shown in Figure 20.

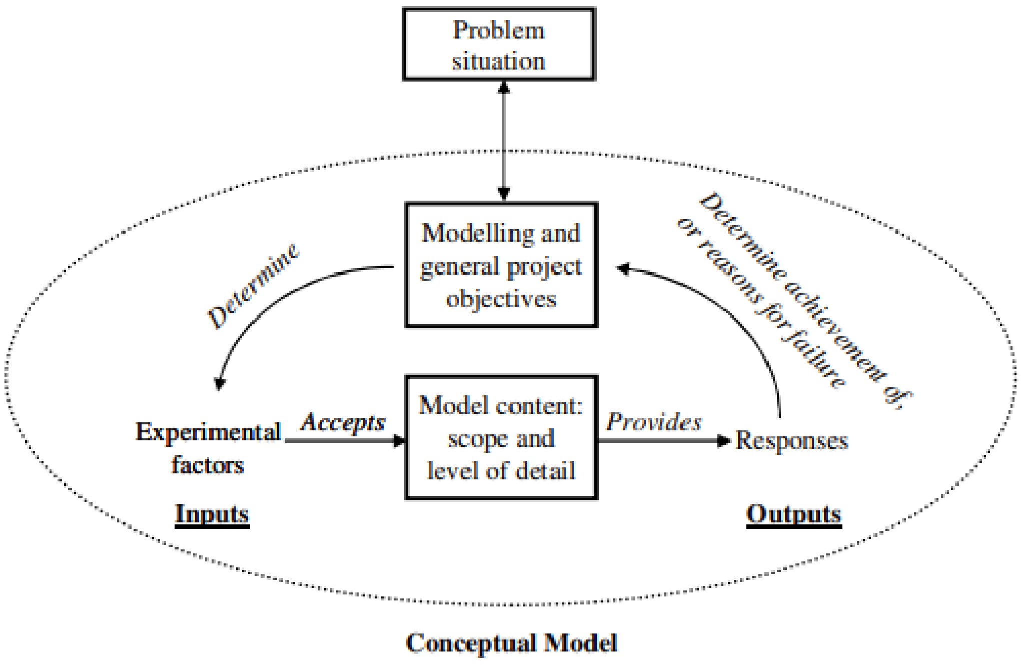

Problem situation
Modelling and general project objectives
Determine
Determine achievement of, or reasons for failure
Experimental factors
Accepts
Model content: scope and level of detail
Provides
Responses
Inputs
Outputs
Conceptual Model

Figure 20: Conceptual Modelling for Simulation[61]

4.3.1 Process Models

Figure 21 shows the picking process model and Figure 22 shows the storing process model. The storing process starts with an arrival event. With a product unit going through a checking process carried out by a worker. After checking is complete, it goes to a storing queue and is then transported by a forklift to its assigned storage slot. The picking process is similar: an order arrival event specifies which product is ordered. Afterwards a forklift is assigned to pick it up and drop it off to the checking point where a worker is assigned to check the product before it is shipped out of the warehouse.

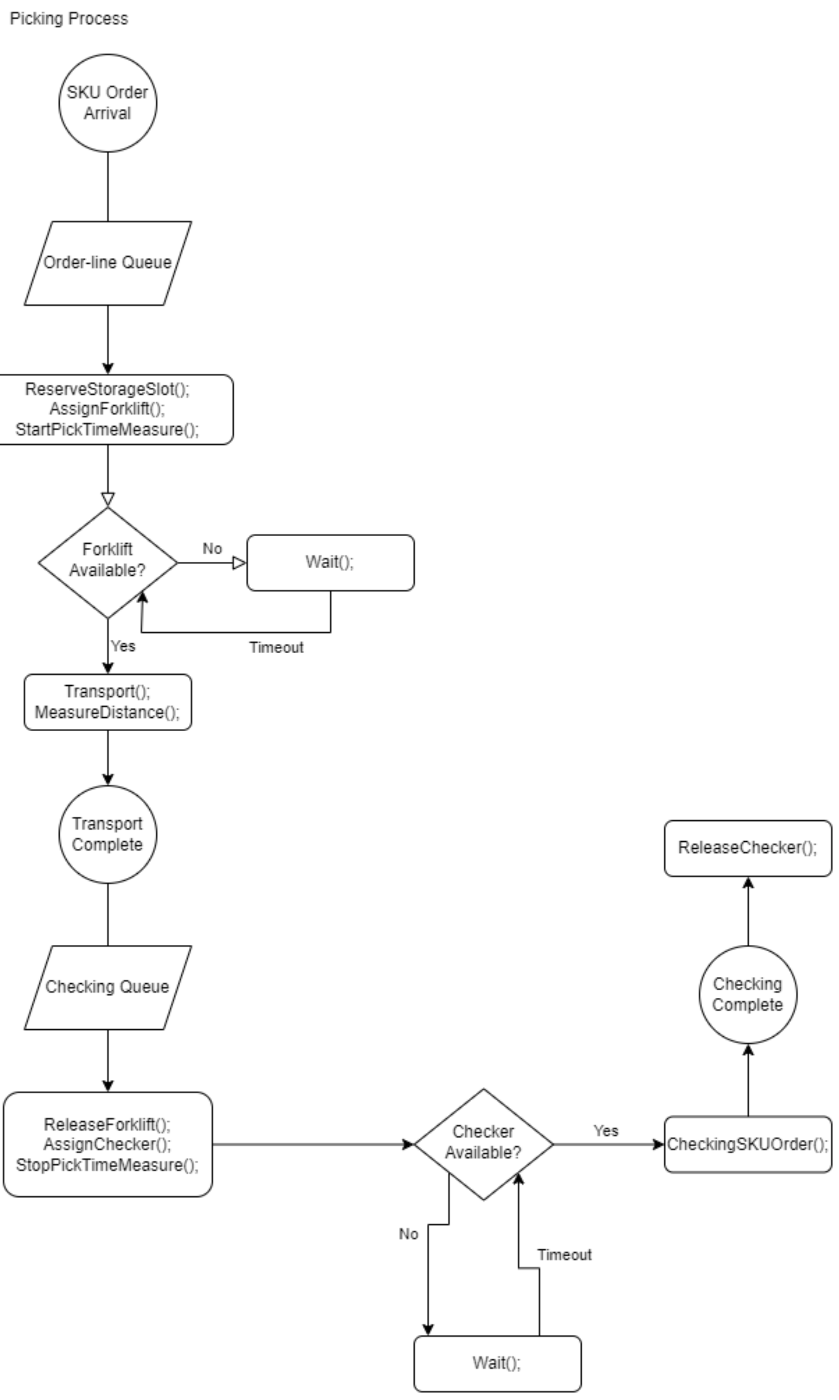


FIGURE 21: Behaviours of the Picking Process

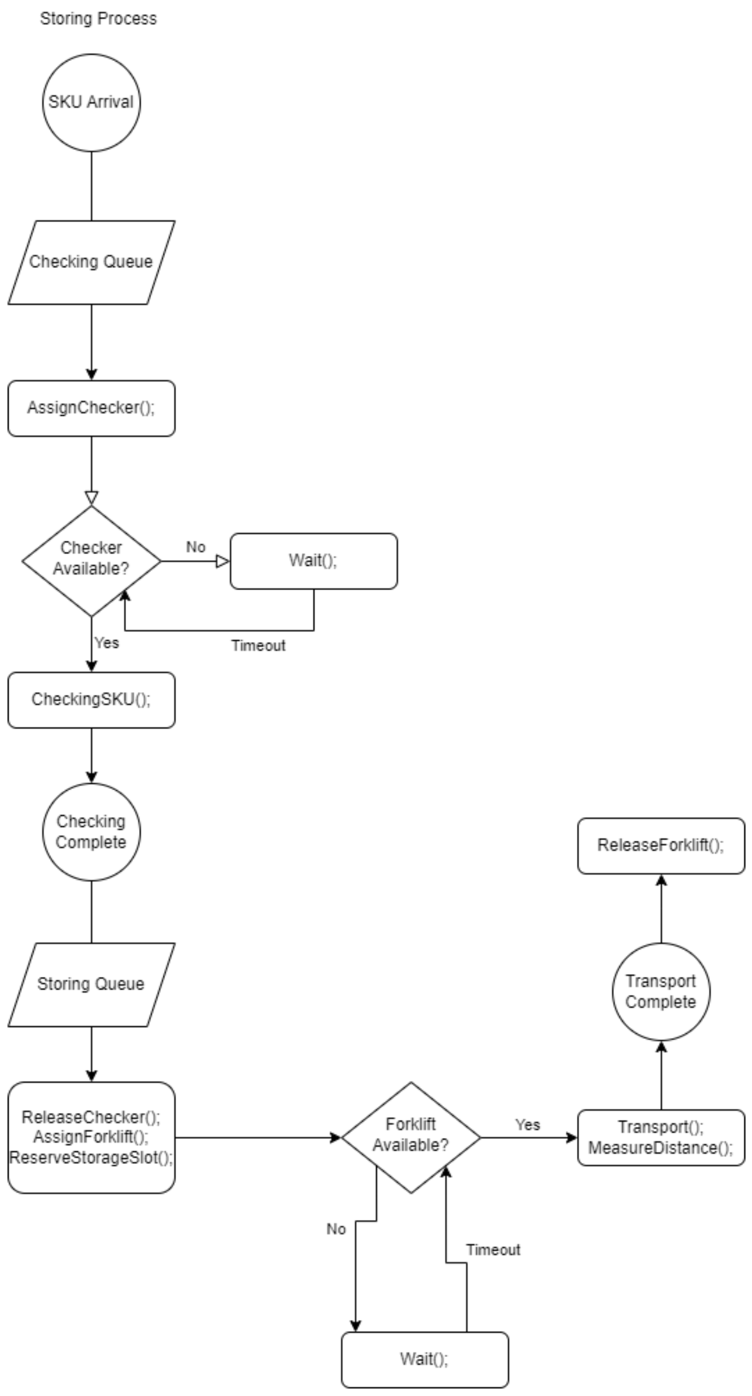


FIGURE 22: Behaviours of the Storing Process

4.3.2 State Machine Model

The state and behaviour of each type of agent are modelled by a state machine: an agent can be inside one of the states that can be transitioned to another state depending on the event it receives. Figure 23 shows the state machine for SKU, storage location, worker, and forklift. The SKU state diagram specifies the states of an SKU from the time it enters the warehouse until it leaves the warehouse. The Storage Location state diagram specifies the states of a slot/cell in the rack storage: the Empty State means that the cell does not contain a SKU; the Occupied State means that a SKU is currently stored in the cell. A forklift can have three states: being idle, transporting products for storage, or transporting products for picking. Similarly, a worker can be idle, or checking arriving products, or checking outgoing products. The benefit of using a state machine for each agent is the ability to track the status of all agents at any given time.

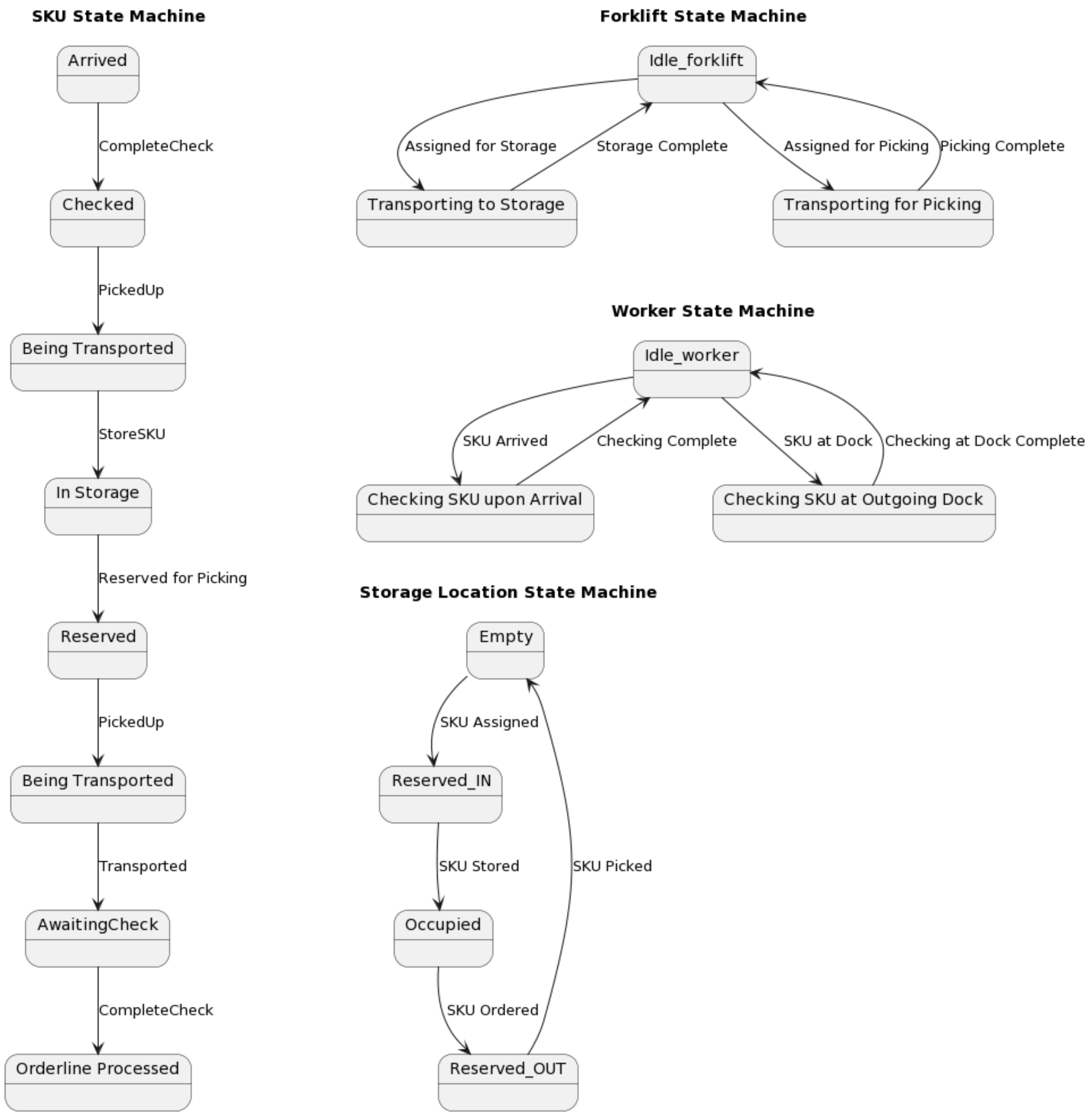


FIGURE 23: State Machine Model for Agents

4.3.3 Aspects of the Conceptual Model

Non-functional Requirements	**Description**
Time-scale	1 day to 1 week
Flexibility	Medium (can be expanded, modified)
Visual display	2D and 3D
Ease of Use	Used mainly by technicians, communicated to stakeholders

Table 16: Non-functional Requirements

Input	**Description**
Arrangement of Class A,B,C Areas	Based on variations of ABC storage policies
Order	Product orders on daily basis
Storage Snapshot	The location of products in the rack storage.
Storing/Replenish Tasks	Products to be stored on daily basis
Number of available workers	Variable number

Table 17: Model Inputs

Output	**Description**
Total travelling distance	Quantitative measurement
Total picking time	Quantitative measurement
Individual SKU picking time	Represented in a chart
Worker Utilisation	Quantitative and chart
Total Waiting Time	Quantitative and chart

Table 18: Model Outputs

4.3.4 Model Details

This section explains the details of queues, agents, assumptions and simplifications.

Checking Queue

It is assumed that the checking area has a max capacity of 10 SKUs. The checking queue follows a First-In-First-Out (FIFO) queue discipline.

Order-line Queue and Storing Queue

We assume that order-line queue and storing queue have a capacity of 30 SKUs each. FIFO is applied here as well.

SKU (Stock Keeping Unit)

A SKU is used as an identification code in warehouses to represent a unique type of product associated with colour, size, brand, etc. For example, a jar of blue and red paint manufactured by the same company with the same packaging will have different SKU IDs.

- ID: The unique identifier for a SKU.
- Product Name.
- ABC Class: Type of the ABC class it belongs to.
- Coordinate (x, y, z): The location of the storage slot storing the SKU.
- Colour: Visual identifier, corresponding to the ABC class.
- Time Stamps: Timestamps to track the processing stages.
- States: Different states the SKU can be in during its lifecycle in the warehouse.

Worker

A worker checks products from the checking queue in the picking or storing process. It often involves scanning, labelling, verifying and packaging products.

- ID: The ID of a worker.
- Processing Time: The distribution of time it takes for a worker to complete the checking activity.
- States: being idle, busy checking incoming or outgoing products.

Forklift

- ID: The ID of a forklift.
- Speed: Movement speed of the forklift.
- Acceleration: Rate at which the forklift changes its speed.

- Elevation Time: Time taken to raise/lower the forks.
- Distance Measurement: Measurement of distance covered during a period.
- States: being idle, busy transporting for storing, or busy transporting for picking.

Storage Location

A storage location is a slot/cell that can host a pallet loaded with products in a rack storage.

- Coordinate (x, y, z): The location coordinates of the storage slot.
- Assigned ABC Class Category: The classification category based on the importance level.
- Assigned SKU ID: The identifier of the SKU assigned to the storage location.
- States: being empty, reserved for storing, occupied, or reserved for picking.

The model assumption is that workers and forklifts are available throughout the day and have no shifts. The model simplifications is that a single SKU occupies one pallet.

4.4 Choice of Simulation Software

We reviewed six major simulation software tools that can be used for modelling a warehouse through their official websites, website articles, and videos of using these tools. We summarise the findings for each simulation tool in Table 19.

Table 19: Comparison of Warehousing Simulation Software

Software	Features	Limitations	Notable Feature
AnyLogic	Has numerous libraries including one for material handling. 2D and 3D objects customisation. Can add, modify, or customise most components using native Java code. Supports Java debugging.	Steep learning curve. Meager number of tutorials online.	Free version comes with most features with few restrictions. Multimethod modeling (combines discrete-event modelling, agent-based modelling, system dynamics).
FlexSim	3D with drag-and-drop. It specializes in logistics, manufacturing, healthcare, and warehousing.	Costly. Challenging scripting language. Free version difficult to acquire.	Interactive 3D models. Modules for conveyor systems, AGVs, etc.
Arena	Tools for simulating various processes. Uses the SIMAN simulation language.	Mainly discrete-event modelling support. Old-fashioned user interface.	Good track record.
SIMUL8	Visual drag and drop. Used for logistics, manufacturing, etc.	Basic visualization. Discrete-event only. Custom scripting language for more complex logic requirements.	Fast simulations.
Tecnomatix	Part of the Siemens PLM software suite. Detailed production system models. 3D visualization.	Paid. Uses the SimTalk programming language	Integration with Siemens software. Works well with other Siemens products.
WITNESS	Discrete and continuous simulation. Visual process-driven design.	Non-intuitive UI, Paid	Discrete and continuous simulation combination.

We evaluated them with two dominant factors in mind: first, the tool should preferably use a general-purpose programming languages; second, it should support DES and ABS modelling. AnyLogic was chosen because of its ability of multi-modelling so that we can employ both discrete-event and agent-based methods in our model. It is also one of the few with free version readily available on the official website. It has no restrictions on the number of components or simulation time in the free version. More importantly, it supports native Java, a general-purpose programming language unlike the proprietary scripting languages required by some of the other simulation software. It offers a Material Handling Library consisting of physical and operational assets, allowing for the representation of a warehouse. 2D and 3D animations are supported, and 3D objects can be imported and customised.

4.5 KPIs

KPIs are defined for travelling distance, picking time, resource utilisation and cost.

4.5.1 Travelling Distance in the Picking Process (single pallet pickup)

In a picking process, a forklift picks up a pallet containing the SKU from a cell in a rack storage and transport it the the dock. Let $S = \{s_1, s_2, \ldots, s_n\}$ be the set of SKUs with each SKU s_i having a coordinate $(x_{s_i}, y_{s_i}, z_{s_i})$. Let (x_f, y_f, z_f) be the home location coordinates of the forklift and (x_d, y_d, z_d) be the coordinates of the docking area.

For each SKU s_i, the distance from the forklift's home location to the SKU is:

$$d_{\text{forklift},s_i} = \sqrt{(x_{s_i} - x_f)^2 + (y_{s_i} - y_f)^2 + (z_{s_i} - z_f)^2}$$

The distance from SKU s_i to the docking area is:

$$d_{s_i,\text{dock}} = \sqrt{(x_{s_i} - x_d)^2 + (y_{s_i} - y_d)^2 + (z_{s_i} - z_d)^2}$$

The objective is to find where the SKUs should be stored in S that minimizes the total distance the forklift travels:

$$\text{Minimize} \quad \sum_{i=1}^{n} (d_{\text{forklift},s_i} + d_{s_i,\text{dock}}) \tag{4.1}$$

4.5.2 Travelling Distance in the Picking Process (multiple SKU pickup in a single trip)

For the picking process in some warehouses such as e-commerce warehouse or spare parts warehouse, a transporter picks up multiple SKUs from different locations and batch them to the destination, which corresponds to a Travelling Sales Man problem [43]. Considering a warehouse with a set of SKUs that a transporter has to pick up, the goal is to determine the shortest possible route for the transporter to pick up all SKUs to minimise the total traveled distance.

Definitions:

- $S = \{s_1, s_2, \ldots, s_n\}$: Set of SKUs.
- $d_{i,j}$: Distance between SKU s_i and SKU s_j.
- The binary decision variable $x_{i,j}$ Represents the travelling of a transporter to move from the location of SKU s_i to that of s_j without other SKUs in the way.
  - $x_{i,j} = 1$: Represents the direct travel from SKU s_i to SKU s_j.
  - $x_{i,j} = 0$: No direct move from SKU s_i to SKU s_j, meaning that the locations of other SKUs to be picked were reached first.

Objective:

Minimize

$$\sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} d_{i,j} x_{i,j} \tag{4.2}$$

It is difficult mathematically to solve it directly [77]. Thus, methods such as Nearest Neighbour Algorithm [64] (a greedy approach), v-opt, k-opt, and Christofides & Serdyukov algorithms are often used as heuristics to solve TSP problems. Since the multiple-sku pickup method is not applied in our simulation model, discussing and applying these heuristics algorithms are outside the scope of this work.

4.5.3 Picking Time

Picking time (T_{pick}) is defined by the duration it takes for a forklift to travel to the SKU's location, collect the SKU, and then transport it to the destination. It consists of the following components:

① **Travel Time** (T_{travel}): Time taken to move to the specific location of the item.

② **Search Time** (T_{search}): Time taken to identify the correct item once at the location.

③ **Elevation Time** ($T_{\text{elevation}}$): Time taken to adjust to the height of the item and retrieve it.

④ **Retrieval Time** ($T_{\text{retrieval}}$): Time taken to attach to the pallet and pick it up.

Considering a linear relationship with height, the elevation time can be calculated as:

$$T_{\text{elevation}} = k \times H$$

Where:

- k is a constant representing time taken per unit height.
- H represents the height from which the pallet is picked.

Combining the components, the overall equation of the picking time becomes:

$$T_{\text{pick}} = T_{\text{travel}} + T_{\text{search}} + k \times H + T_{\text{retrieval}} \tag{4.3}$$

The idea is to use an optimal variation of the ABC storage policy to reduce T_{pick}.

4.5.4 Utilisation & Waiting Time

To maintain efficiency in a warehouse, it is necessary to balance the utilization of resources (such as workers and forklifts) and associated costs. Queuing theory provides a mathematical framework to analyze and understand these dynamics.

The utilization factor ρ in queuing systems is given by:

$$\rho = \frac{\lambda}{\mu s} \tag{4.4}$$

Where:

- λ is the average arrival rate.
- μ is the average service rate (productivity).
- s is the number of resources such as workers or forklifts.

Higher utilization (ρ closer to 1) means resources are used more efficiently. However, it might lead to longer waiting times when there are too many tasks awaiting to be processed at the same time, potentially increasing indirect costs due to delays or congestion. Thus, there is a trade-off between direct costs of adding more resources (reducing ρ) and the indirect costs of imposing a high utilization.

The total waiting time (W) in the SKU checking process is computed as the sum of the waiting times for each individual SKU (w_i) from $i = 1$ to n, where n is the total number of SKUs.

$$W = \sum_{i=1}^{n} w_i \tag{4.5}$$

and

- W is the total waiting time
- w_i is the waiting time for the i-th SKU
- n is the total number of SKUs in the queue

Each w_i is the time that the i-th SKU spends waiting to be checked, which contributes to the total waiting time in the checking process.

4.6 Model Implementation

In this section, the implementation of model in AnyLogic is explained. We describe the physical layout, agent and Java class model, arrival events, picking and routing strategy, and how the KPIs are measured in the software.

4.6.1 Layout

The layout designed for the simulation is depicted by Figure 24. Overall it is simplified to contain the core areas of a warehouse for storing and picking processes. The incoming dock area is where SKUs are generated, and the outgoing dock area is where the ordered SKUs are sent out. The trucks, however, are not active agents in the model because we can then treat the arrival and departure of SKUs simply as a function of time. The checking area is for workers to verify SKUs before they were stored. There is a temporary ground storage space (with no space limit to host products) in both docking area. The incoming dock area is assigned to the top left relative to the rack storage while the outgoing dock area is assigned to the top right. This layout contains the main physical components in a warehouse without considering doors, walls, and other obstacles except for the rack storage.

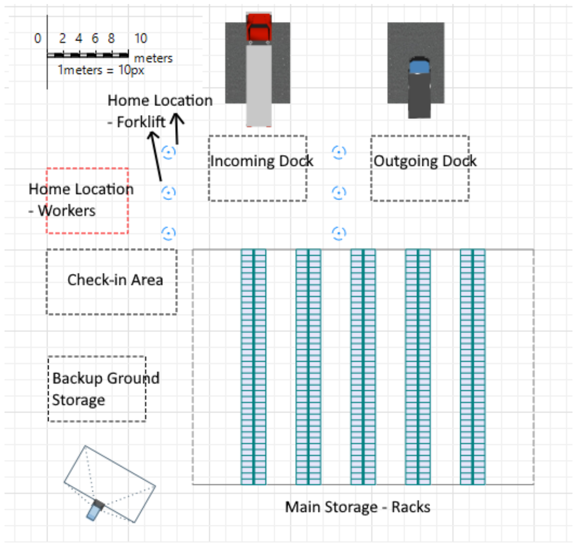


FIGURE 24: Layout Used for Simulation

4.6.2 Agent and Java Class Model

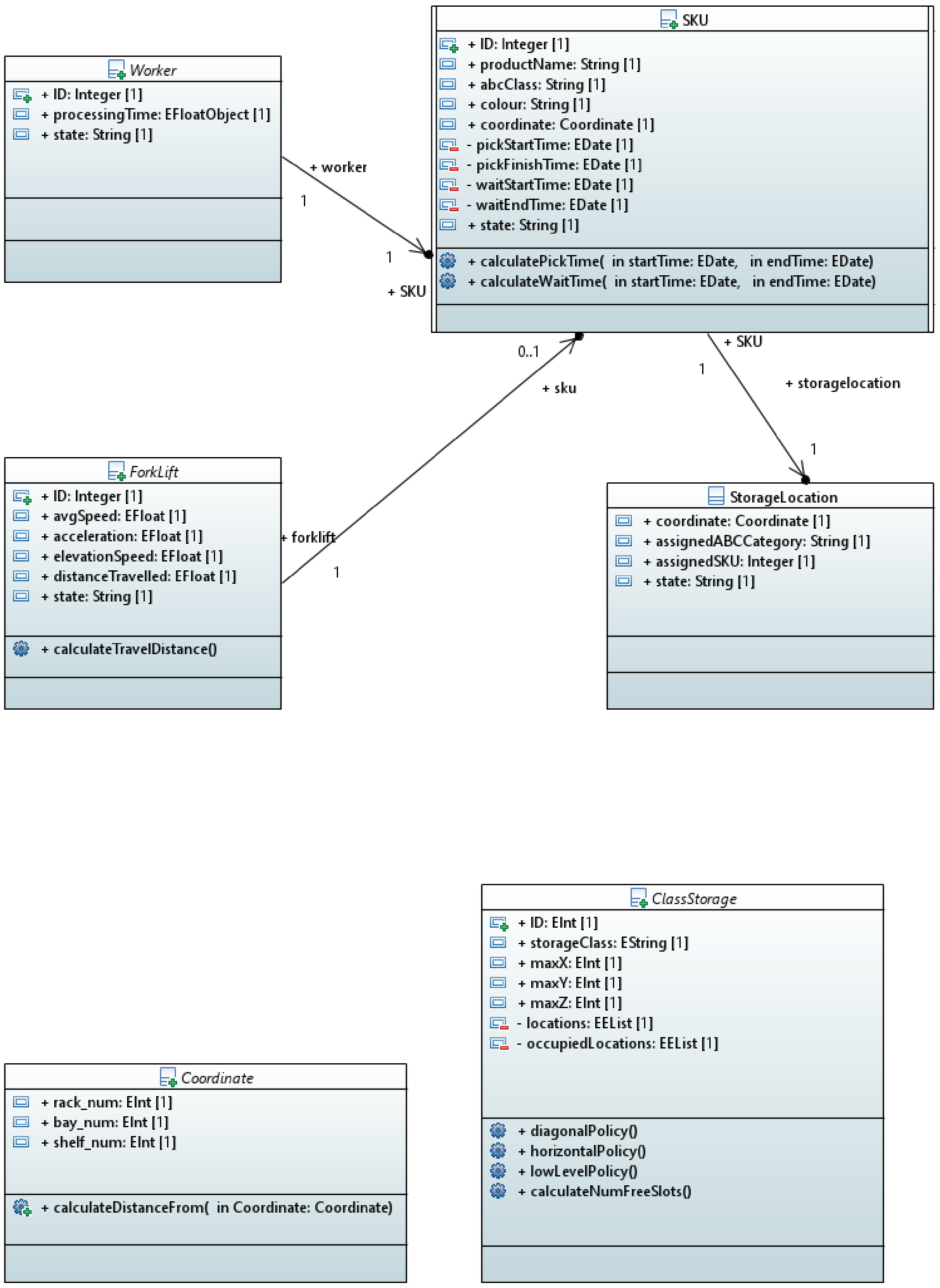
Figure 25 shows a class diagram to represent agents and general classes in the simulation model. Storage Location, Worker, Forlift, and SKU are agents classes; Coordinate and ClassStorage are utility classes.

Rack Storage Location A rack storage with 10 racks, 40 bays, and 5 shelves is modelled in our simulation. Each individual cell/slot is called Storage Location and is represented by a three-dimensional coordinate. A single SKU occupies one cell/slot in this rack storage.

Workers Workers function as checkers who verify SKUs in the storing and picking processes. The time it takes to check any given SKU is modelled with a triangular distribution. No schedule or work shifts are configured in the current implementation.

Forklifts We have six forklifts with an average speed of 1m/s. Each forklift has its own home location. No work shifts are configured.

SKUs Each SKU is assumed to occupy one pallet. So, retrieving a SKU is equivalent to retrieving a pallet from the storage.



Worker
+ ID: Integer [1]
+ processingTime: EFloatObject [1]
+ state: String [1]
SKU
+ ID: Integer [1]
+ productName: String [1]
+ abcClass: String [1]
+ colour: String [1]
+ coordinate: Coordinate [1]
- pickStartTime: EDate [1]
- pickFinishTime: EDate [1]
- waitStartTime: EDate [1]
- waitEndTime: EDate [1]
+ state: String [1]
+ calculatePickTime(in startTime: EDate, in endTime: EDate)
+ calculateWaitTime(in startTime: EDate, in endTime: EDate)
+ worker
1
1
+ SKU
0..1
+ sku
+ SKU
1
+ storagelocation
1
ForkLift
+ ID: Integer [1]
+ avgSpeed: EFloat [1]
+ acceleration: EFloat [1]
+ elevationSpeed: EFloat [1]
+ distanceTravelled: EFloat [1]
+ state: String [1]
+ calculateTravelDistance()
+ forklift
1
StorageLocation
+ coordinate: Coordinate [1]
+ assignedABCCategory: String [1]
+ assignedSKU: Integer [1]
+ state: String [1]
ClassStorage
+ ID: EInt [1]
+ storageClass: EString [1]
+ maxX: EInt [1]
+ maxY: EInt [1]
+ maxZ: EInt [1]
- locations: EEList [1]
- occupiedLocations: EEList [1]
+ diagonalPolicy()
+ horizontalPolicy()
+ lowLevelPolicy()
+ calculateNumFreeSlots()
Coordinate
+ rack_num: EInt [1]
+ bay_num: EInt [1]
+ shelf_num: EInt [1]
+ calculateDistanceFrom(in Coordinate: Coordinate)

FIGURE 25: Agent and General Class Diagram

Below is the description of the general Java classes.

Coordinate A coordinate (x,y,z) corresponds to the storage rack number (equivalent to column), bay number (equivalent to row) and shelf number (equivalent to level), respectively. A coordinate represents a unique cell/slot of the rack storage.

ClassStorage It contains the coordinates, functions, and other utility variables for the designated area of an ABC-based storage. It can be instantiated to represent A, B, or C area. When the simulation starts, the chosen policy is configured using this class.

4.6.3 SKU Arrival Events

There are several options for generating events for SKU order arrival such as rate, rate schedule, inter-arrival time, arrival table from databases, calling custom function. In our experiment, we chose scheduled arrival table to generate incoming SKU order events, which contains the number of SKUs and their arrival time for one day. The table is fixed so that we can use the same table to test different scenarios in our simulation runs.

4.6.4 Picking and Routing Strategy

Single order pick is chosen as the pick strategy. It is the default picking behaviour in AnyLogic, because a forklift can only transport one SKU object at a time. This means that a forklift picks a single SKU from an order and transport it to the destination rather than combining several SKUs in one trip. This means that the distance measurement with Equation 4.1 is used instead of Equation 4.2. The routing strategy is automatically determined by the AnyLogic software with the shortest path chosen for each picking activity.

4.6.5 Methods for KPI Measurement

The KPIs are measured in the simulation runs with the following methods.

Distance Travelled Equation 4.1 reflects the measurement. The distance travelled by individual forklifts can be measured by calling a specific function of Transporter Agent. We sum up the distance travelled by all forklifts every time a forklift completes a picking activity.

Picking Time For each picking activity, we measure it by placing two Time Measurement blocks between the start and the end of the picking activity blocks (namely before and after the retrieve1 block. Individual picking time can be displayed on a chart and the total picking time can be viewed directly by clicking on the block during a simulation run. Another way to measure the picking time is to record and subtract the timestamps of each SKU right before and after the picking activity. These two methods are tested to yield the same results. The measurement is reflected by Equation 4.3.

Utilisation, Waiting Time We measure utilisation of workers (see Equation 5.3) for the picking process flow by calling a calculation function and the results are plotted in a line chart during the simulation. The total waiting time (see Equation 4.5) can be measured by attaching time measurement blocks between the checking queue. It can also be measured by summing the individual waiting time of SKUs recorded in each SKU agent. The waiting time can be visualised in a line chart plotted against time to show how queued up the SKUs are while waiting to be processed in the checking activity.

Chapter 5

Simulation Experiments

Based on the simulation model specified in the last chapter, this chapter describes two experiments. The first is to assess the ABC storage policy variants' impact on travelling distance and picking time in the picking process. The second is to measure the utilisation of workers and the cost measured by the waiting time and queue. The equations for the KPIs measured and the steps of each experiment are specified therein. After showing and discussing the results of the experiments, verification and validation were conducted.

Both experiments used the common configuration shown in Table 20 with the order table shown in Table 30.

Subject	**Parameters**
Forklifts	Speed: 1m/s, Acceleration: 1 m/s^2 Elevation speed: 0.3 m/s, Number of forklifts: 6
SKU Order Arrival Rate	Based on a schedule in the span of 24 hours.
Workers (Checkers)	Number of checkers: 3, Checking time: triangular (0.5, 1, 1.5) minutes.
Rack Storage	Rack configuration 10x40x5 (10 racks, 40 bays, 5 shelves), total slots: 2000, rack depth: 1.5m, shelf height: 1m, storage length: 30m, storage width:29m, access zone width: 6m
Warehouse Indoor Dimension	60m x 60m
Simulation Seed	Seed number: 2
Total Running Time for Each Experiment	48 hours
Total Number of Pallets to Be Picked	228 pallets

Table 20: The Common Configuration

According to movement history from forecasting service, There are always three forklifts assigned each day, and on average 120 pallets were the daily demand for picking. We could not model the correlation between number of forklifts and number of pallets to be picked because we do not have data about it. So, we assume that they follow a linear relationship where if number of the picking demand is doubled, twice as many forklifts are needed. Therefore, we allocated six forklifts to fulfil picking tasks of 228 pallets for our experiments.

5.1 Travel Distance and Picking Time Assessment

This section describes ABC classification, size of ABC area, the location of the products prioritised to be picked, ABC storage policy variants, and simulation steps. The objective is to evaluate the impact of four variations of the ABC storage policy on the total distance travelled by forklifts and the total picking time in the picking process flow. This should help us identify the most efficient storage policy in terms of these two KPIs.

5.1.1 ABC Classification, Area Size, Selection Priority

Based on the movement frequency of products for the first quarter of 2021, class A, B, C products account for 63%, 24%, and 13% of the movements respectively. We extrapolated this information to generate a list of orders (see Table 30) for an 24-hour period, meaning that class A orders comprise 63%, class B 24%, and class C 13% of the orders respectively. The implication is that the order list represents the daily frequency of the products being ordered on average for the quarter.

The occupancy rate of the storage is set to be 85% for each area, because we assume the storage is highly utilised by this warehouse. This means that on average A products are occupying 85% of slots in Area A, and so do B and C products in their respective area. As for the allocated sizes of A, B, and C areas in our experiments, they are configured to be close to each other as shown in Table 21.

Storage Policy	**Area Sizes**
Horizontal	A:800; B:600; C:600; Total:2000
Low-level	A:800; B:400; C:800; Total:2000
Diagonal	A:690; B:780; C:530; Total:2000
Random	Total:2000

TABLE 21: Storage and SKU

When a picking task is generated for a specific product, the product that is closest to the p/d point (in our case the left dock) is going to be picked.

Traveling Distance (D) The traveling distance D for a forklift is calculated based on the summation of distances from the home location to each location of the SKU plus the location of the SKU to the docking area plus the forklift's returning distance from the dock to home location. N is the total number of SKU to be picked. The distance does not include the vertical lifting of the forklift reaching the pallet at a height.

$$D = \sum_{i=1}^{N} (D_{\text{home},i} + D_{\text{i, dock}} + D_{\text{dock,home}}) \tag{5.1}$$

Picking Time (T_{pick}) The total picking time T_{pick} for forklifts to pick up N SKUs comprises the following components:

- $T_{\text{travel},i}$ - Time taken to travel from the forklift's home location to the i-th SKU, then to the dock.
- $k \times H_i$ - Elevation time, which depends on the height H_i from which the SKU is picked.
- $T_{\text{retrieval}}$ - A constant time for a forklift to attach to a pallet and mount it to itself.
- $T_{\text{i},dock}$- Time taken to travel from the i-th SKU to the dock

Thus, T_{pick} can be expressed as:

$$T_{\text{pick}} = \sum_{i=1}^{N} (T_{\text{travel},i} + k \times H_i + T_{\text{retrieval}}) \tag{5.2}$$

5.1.2 Variations of the ABC Storage Policy

We define four variations of the ABC storage policy, derived from [11] [71], as:

$$P \in \{P_1, P_2, P_3, P_4\}$$

where each P_i represents a spatial arrangement of A, B, and C products by these variants: random, horizontal, diagonal, low-level policy variants. SKUs in area A, B, C are represented as blue, green, red boxes respectively (as shown in Figure 29). Area A is the closest to the top left dock, followed by area B and C.

- Random Policy: the assignments of pallets are randomly allocated to all possible storage locations, as shown in Figure26.

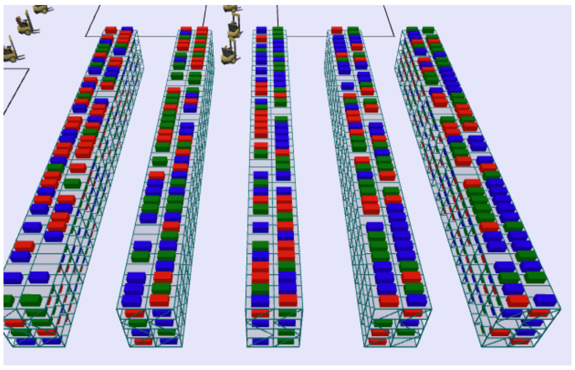


FIGURE 26: Random Policy

- Horizontal Policy: the storage racks are divided into three horizontal sections, with the section A closest to the docks, as shown in Figure 27.

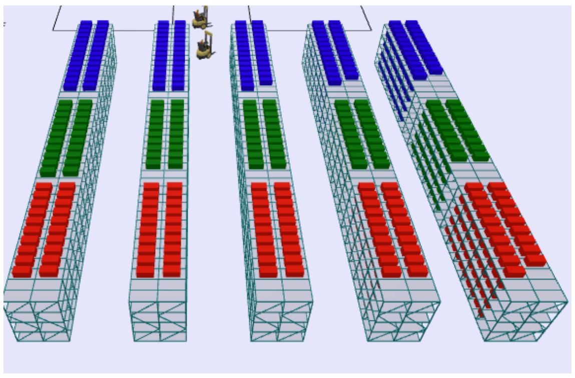


FIGURE 27: Horizontal Policy

- Low-Level Policy: as shown in Figure 28, Class A products are stored on the lower levels and sections closer to the docks. It would take less time to retrieve them compared to products located higher on the rack due to less elevation time taken for forklifts to pick them up.

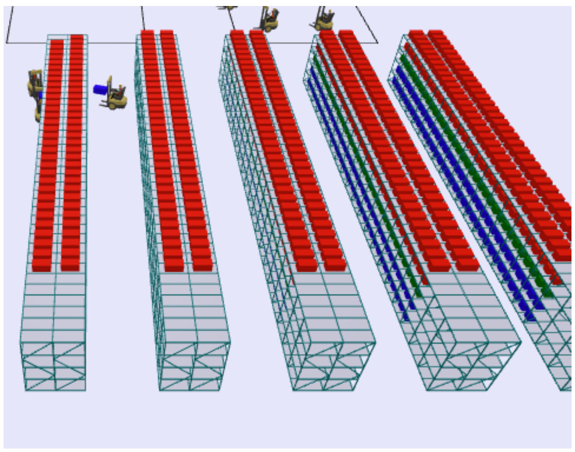


FIGURE 28: Low-level Policy

- Diagonal Policy: sections are divided diagonally and section A is close to forklifts and the docks, as shown in Figure 29.

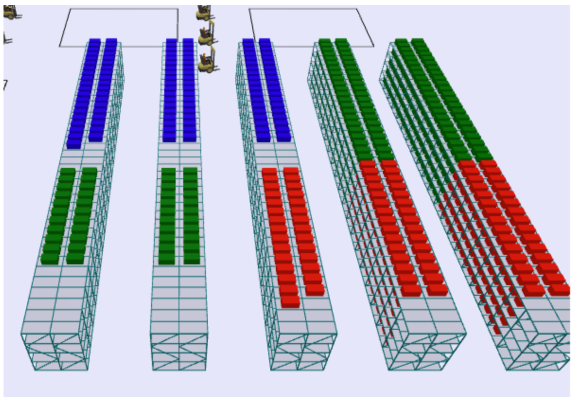


FIGURE 29: Diagonal Policy

5.1.3 Steps

For each storage policy P_i,

- At the start, we initialise the storage with 9 different SKUs from A,B,C class categories with ID ranging from 0 to 8. In total, they occupy 85% of the storage's max capacity. SKUs with higher ID are stored further away and thus have incrementally more total distance as defined by Equation 4.1. Coordinate (0,0,0) is used as the reference point to the p/d point. For example, the first SKU with ID 0 belongs to class A and is assigned the coordinate (0,0,0), followed by the second SKU with ID 0, assigned to (0,0,1), etc.
- The arrival schedule from Table 30 in the Appendix is used to generate picking orders. Each experiment runs for 48 hours. 228 SKUs are ordered in the span of the first 24 hours. For the next 24 hours the same orders are repeated.
- Measure and record D Equation 5.1and T_{pick} Equation 5.2.
- Repeat three times to account for variations of total distance travelled and total picking time.

Table 22 below shows the configuration of storage and SKUs for this experiment.

Subject	**Parameters**
Storage Occupancy/Utilisation	85% for each of the A, B, C area.
Storage Policies	P1: Random; P2: Horizontal; P3: Diagonal; P4: Low-level
SKU ID Allocated in Each Area	Area A: **0 1 2** ;Area B: **3 4 5** ;Area C: **6 7 8**
Number of SKUs in each area	**0**:50% of Area A; **1**:25% of Area A; **2**:25% of Area A; **3**:50% of Area B; **4**:25% of Area B; **5**:25% of Area B; **6**:50% of Area C; **7**:25% of Area C; **8**:25% of Area C;
Number of SKUs ordered for each ID in the 24-hour period	**0**:63 ;**1**:48 ;**2**:33 **3**:20 ;**4**:20 ;**5**:15 ;**6**:11 ;**7**:9 ;**8**:9
Total ordered SKUs for the 24-hour period	228

Table 22: Storage and SKU

5.2 Utilisation and Cost Assessment

This experiment measures the impact of having different number of workers in the picking process flow. The experiment considers scenarios with 1, 3, and 5 workers.

$$N_w \in \{1, 3, 5\}$$

We measure the workers' utilisation as the proportion of time they are busy in a certain period T. We approximate the indirect cost by the number of SKUs waiting to be checked in a certain period T.

Worker Utilization (ρ) is represented by ρ, measuring the efficiency of a worker in terms of how much of the available working time is actively spent on tasks. It is defined as:

$$\rho = \frac{T_{\text{busy}}}{T} \tag{5.3}$$

Where:

- T_{busy} is the amount of time the worker is busy.
- T is the total available working time (in our case the entire simulation time).

Total waiting time for all SKUs is defined in Equation4.5.

Steps

For each scenario (the allocated number of workers in N_w) :

- The configuration is the same as Section 5.1 and it uses horizontal policy where the picking activities are run for two days.
- We measure the utilisation of workers with Equation 5.3and total waiting time of SKUs with Equation 4.5. Their results are displayed as charts during the simulation runs.
- Each scenario is repeated three times to observe whether there are random effects during simulation.

5.3 Results

We present the KPI measurements and observations for the two experiments.

5.3.1 Travel Distance and Picking Time Assessment

Table 23 shows the result for the experiment: Travel Distance and Picking Time. An average value is taken from three runs per scenario. These three runs showed consistent results with almost no variations for the KPIs measured.

Scenario: Storage Policy Name	**Total travelling distance (D)**	**Total Picking Time ($T_{\textbf{pick}}$)**
Horizontal Policy	4.37km	9.16 hours
Low-level Policy	4.03 km	7.20 hours
Diagonal Policy	3.93 km	7.87 hours
Random Policy	5.33km	11.57 hours

Table 23: Results for Travel Distance and Picking Time Assessment

We sorted and allocated 9 different SKUs based on their order frequencies: the highest are the closest to the forklifts and the docks; then we assign them to the corresponding class-based areas, varied by the variants of ABC storage policy. Orders were generated in a way the the most frequently ordered SKUs are picked up the most during a 24-hour period, with Class A SKUs constituting 63% of the total order volume. The result shows that the diagonal policy incurs the least travelling distance but the low-level policy triumph others in time saving, which could be attributed to the fact that there were less elevation time spent during pickups. This experiment suggests that we can choose the best policy based on the performance of the KPIs. If picking time is vital to stakeholders, the low-level policy should be considered. It could also be used as a testing ground to help us devise a more specific policy with higher performance.

5.3.2 Worker Utilisation and Waiting Time Assessment

Table 26 shows the results for the experiment: Worker Utilisation and Waiting Time Assessment.

Scenario: Number of Workers	**Worker Utilisation (ρ)**	**Total Waiting Time ($W_{\mathbf{sum}}$)**
1	18%	45.96 hours
3	6%	13.92 hours
5	4%	8.10 hours

Table 24: Experiment Results for Worker Utilisation and Waiting Time

With only one worker, we have utilisation of 18% with a long waiting time of 45.96 hours for SKUs. The total waiting time can be reduced by 69% by adding two extra workers. However, whether the two metrics could really impact the operational effectiveness is better estimated through chart inspection, as Figure 30 shows. For utilisation in this time window, the worker is busy for about 15 minutes every hour, which is likely not going to cause stress for the person. When viewing the activity from 6 am, we had about 12 SKUs waiting to be checked and it lasted for about 15 minutes. Congestion issues may occur if there was not enough capacity in the checking area, and it could also cause higher waiting time for trucks if these SKUs cannot be checked and loaded into outbound trucks fast enough. In comparison, having three workers can significantly reduce the duration of the waiting, as shown in Figure 31.

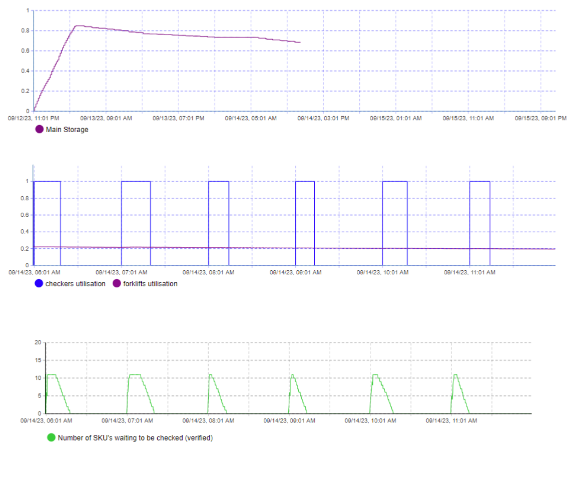


Figure 30: Scenario: One Worker

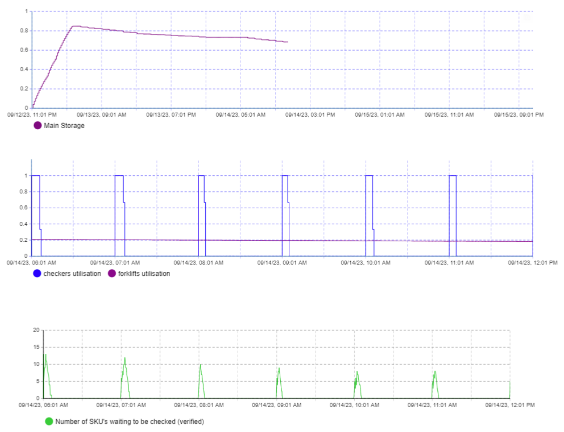


FIGURE 31: Scenario: Three Workers

The experiment illustrates how the operational performance can be estimated for scenarios containing different number of available workers in a single checking activity. The chart inspection showed us a possible congestion when we have only one worker. In a real warehouse, we may have the same pool of workers performing multiple activities in multiple process flows. This is when chart inspection can aid in detecting inefficient parts of the process.

5.4 Verification

Verification includes Exception Handling, which are the functions to handle errors that could occur during a simulation run when unexpected or unwanted situations take place. Debugging, Animation & Graphical Inspection are used to test syntactical, compilation, and run-time errors. Run-time testing is used to check whether the simulation runs normally continuously given fixed inputs.

5.4.1 Exception Handling

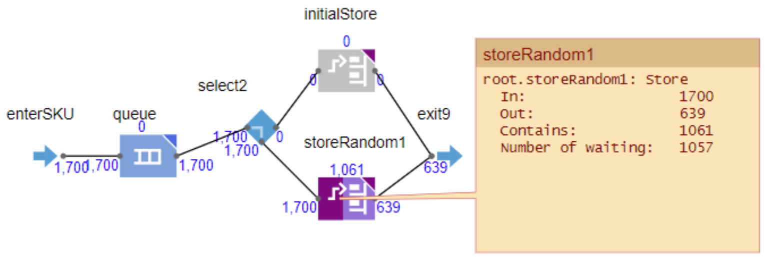
Some exceptions were taken into account including:

- **Condition**: receiving an order where there is no matching SKU in storage. **Solution**: reject the order and record it in the database.
- **Condition**: there is no storage space for an incoming SKU in the corresponding class-based area. **Solution**: destroy the SKU object and record the destroyed object in the database. Alternative Solution (not implemented): store the SKU in the closest available area.

5.4.2 Debugging, Animation and Graphic Inspection

During development, syntactic and compilation errors occurred time to time and they are identified by reading error messages and using the debugging terminal. Having these errors outstanding would cause the model not to start at all. Thus, these errors were prioritised to be solved during the development.

Run-time errors may occur when the simulation is running. It can be identified by graphic inspection of the process blocks. Figure 32 shows a process block (in red). For example, if there is an error when storing a product into a cell of the rack storage because that slot is reserved or already occupied, a red dot in the process block would be visualised with additional information in the debugging terminal. In the 3D graph (as shown in Figure 33, a red rectangular shape would also be highlighted at the cell slot where the problem occurred. These errors are more difficult to solve than syntactic or compilation error because it implies that part of a running processes caused the program flow to err. It is usually identified and solved by doing run-time testing.



```
storeRandom1
root.storeRandom1: Store
  In:                    1700
  Out:                   639
  Contains:              1061
  Number of waiting:     1057
```

FIGURE 32: Graphical Inspection with the Process Flow Blocks

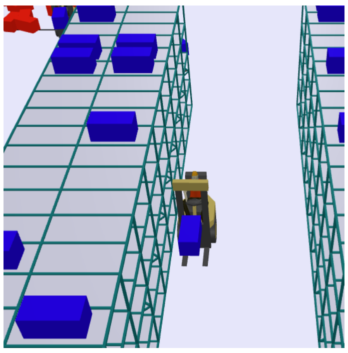


FIGURE 33: Graphical Inspection with the 3D Graph

5.4.3 Run-time Testing

The storing and picking process are tested together and ran for seven days (in simulation software time) with automatically generated incoming and outgoing orders using rate of arrival. The simulation did not encounter any run-time issue. This suggested that all the incoming and outgoing products were handled correctly by the process flows.

5.5 Validation

Data, process and theory validation ensure that the simulation model of the warehouse resembles the real-world counterpart. Face validation ensure that the model can be effectively used by stakeholders. Since our design is based on a hypothetical warehouse, guidelines are provided for real warehouses for data, process, and face validation.

5.5.1 Data validation

Data validation can be carried out as Table 25 illustrates.

Data	**Parameters**
Forklifts	**Speed**, **acceleration**, **elevation speed**: measured by comparing them against the real-world counterpart. They can be estimated by doing field experiments. **Shifts**: follow the real schedule of forklift operators.
SKU	**Types of SKU**, **ID numbers**, **demand patterns**, **sizes**, **Special Conditions**: whether they require special storage area. **The number of SKUs that can be stored into a single pallet.**
Workers	**Shifts**: check the real schedule of workers against the table used by the simulation model. **Productivity**: may vary based on skill level; can be acquired through field observation or by using historic data. The processing time for a task might follow a distribution pattern.
Rack Storage	**Total Slots, Rack Depth, Shelf Height, Storage Length, Storage Width, Access zone Width.**
Warehouse Indoor Environment	**Indoor dimensions**, **Building Components** such as walls, obstacles, and elevators can be validated by comparing the real blueprint and the 2D model in the simulation tool.

Table 25: Data Validation Table

5.5.2 Process Validation

The process validation can be done by:

- Verifying process flows with the stakeholders of the processes.
- Expert review and adjustment based on their feedback.
- Conducting experiments with storing and picking processes using real-world historic data. For instance, use a week of data to check whether **average storing and picking time** matches with the reality, whether **processing time of each activity** closely follow the ones measured in real world.

5.5.3 Face Validation

Face validation is to check whether the behaviours of the agents and processes match with those in the real-world system. One of the validation tasks to be considered is the routing of forklifts: it may not resemble the real choices of their operators because human operators may occasionally take any route they feel the best or make detours or delay the task because of various reasons such as distractions. As a result, the model could underestimate the travelling distance and picking time. Another validation task is whether the workers store or pick up SKUs according to the computer's instructions; in the real world, sometimes workers do not follow these instructions; they might store, for example, a SKU somewhere close to the slot originally assigned by the computer. The impact resulting from these issues should be quantified and adjusted to make the measured KPIs closer to reality.

5.5.4 Theory Validation

The KPIs measured in the two experiments should reflect the theories behind them. Table 26 shows the theory validation by comparing theory and what is measured in the simulation software.

KPI	**Comparison**	**Validation**
Total travelling distance (D)	Equation 4.1 & Equation 5.1	The distance covered by the forklifts were calculated for the picking process and the components match with the equation from the theory.
Total Picking Time (T_{pick})	Equation 4.3 & Equation 5.2	The total picking time started the moment a picking task is generated for a forklift and ended when the picked product is delivered to the outgoing dock. It accounts for travelling time, search & retrieval time, and elevation time and thus reflects the picking time from the theory.
Worker Utilisation (ρ)	Equation 4.4 & Equation 5.3	The duration of every worker when busy is measured by the simulation software; the total time workers are working is the entire duration of the simulation run-time (48 hours). Thus, the measurement reflects the theory.
Total Waiting Time	Equation (W) 4.5	Each SKU's waiting time is calculated and summed up and thus it reflects the theory.
Cost	N/A	The cost is approximately by how queued up in the checking queue by chart inspection.

Table 26: Theory Validation

Chapter 6

Evaluation

This is chapter describes how the artefact can be used by developers, warehousing supervisors and managers. Use cases are designed to illustrate the usage of the artefact. Specific details and data of the use cases have been omitted to maintain confidentiality. We also evaluated dashboard pages in Power BI visualisation tool with experts surveys.

6.1 Implementation Model

Figure 34 shows the implementation model of the two solutions. Spyder with Python capability serves as the Data Tool; the Warehouse Simulator is run by AnyLogic simulation software, and visualisation tool is chosen to be Power BI.

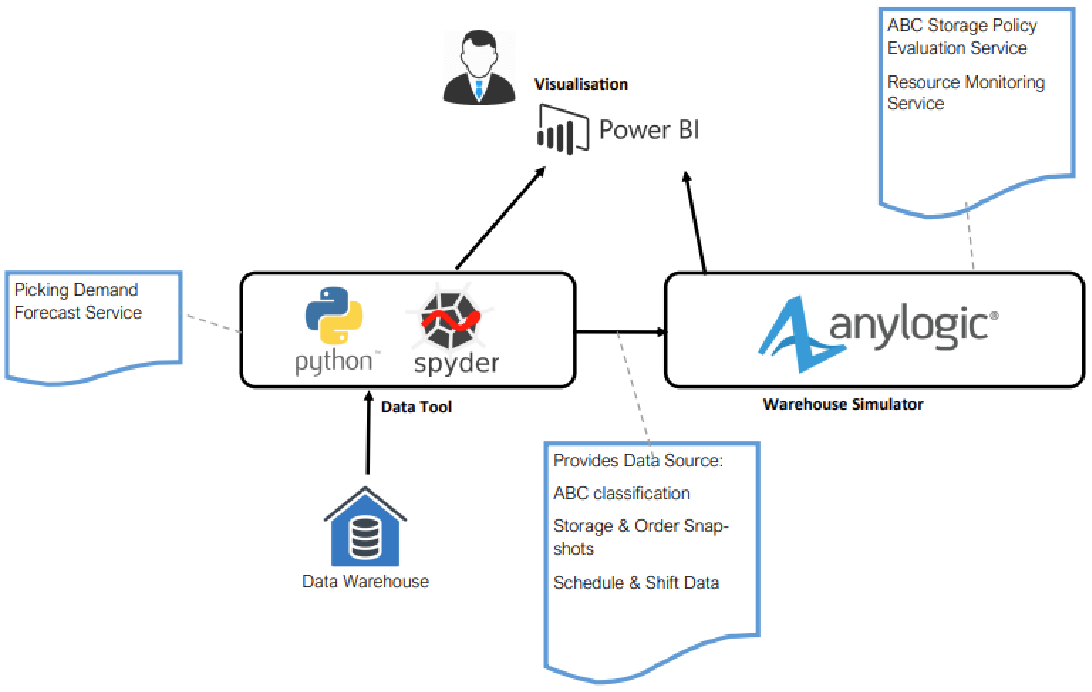
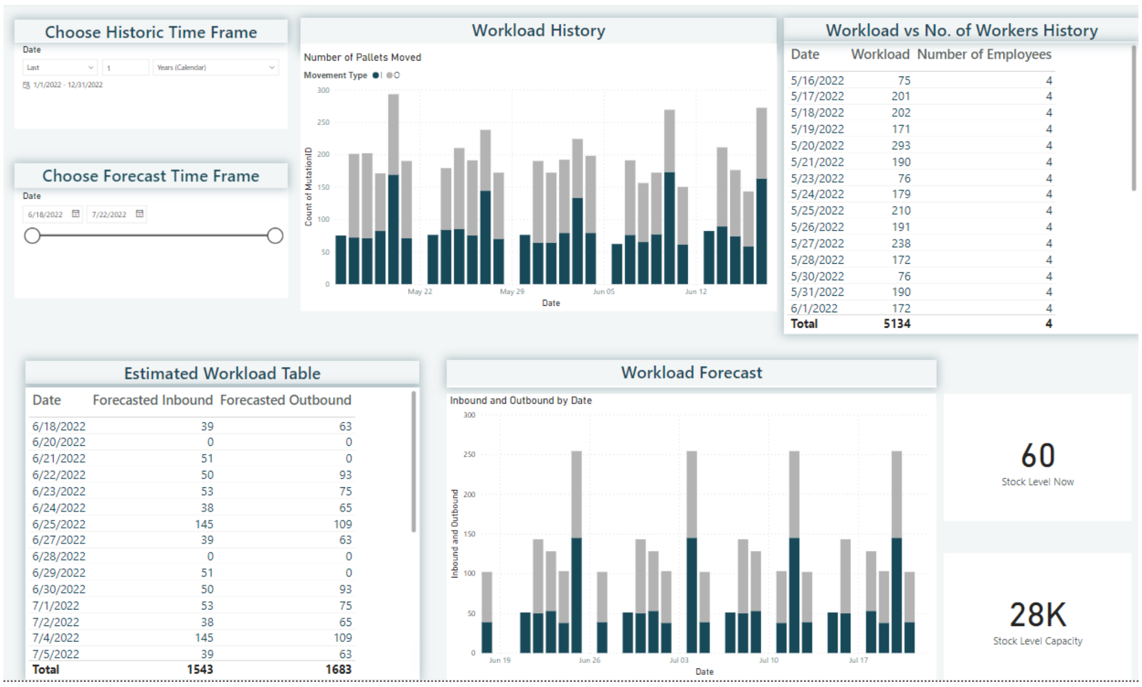


FIGURE 34: Implementation Model

6.2 Use Case - Forecasting Service

Description
Algorithms are used to forecast short-term and medium-term workload demands, helping decisions in resource allocation and capacity planning.
User Interactions
Developers utilize data tools and visualization software for algorithm optimization and dashboard design. Managers and Supervisors access these dashboards for operational insights and decision-making.
Output Data
The output includes forecasts of picking workload in various forms of graph such as line charts, pie charts, and tables.

Figure 35 shows a sample page of Power BI Dashboard.



Workload vs No. of Workers History

Date	Workload	Number of Employees
5/16/2022	75	4
5/17/2022	201	4
5/18/2022	202	4
5/19/2022	171	4
5/20/2022	293	4
5/21/2022	190	4
5/23/2022	76	4
5/24/2022	179	4
5/25/2022	210	4
5/26/2022	191	4
5/27/2022	238	4
5/28/2022	172	4
5/30/2022	76	4
5/31/2022	190	4
6/1/2022	172	4
Total	**5134**	**4**

Estimated Workload Table

Date	Forecasted Inbound	Forecasted Outbound
6/18/2022	39	63
6/20/2022	0	0
6/21/2022	51	0
6/22/2022	50	93
6/23/2022	53	75
6/24/2022	38	65
6/25/2022	145	109
6/27/2022	39	63
6/28/2022	0	0
6/29/2022	51	0
6/30/2022	50	93
7/1/2022	53	75
7/2/2022	38	65
7/4/2022	145	109
7/5/2022	39	63
Total	**1543**	**1683**

FIGURE 35: Power BI Dashboard Sample for Forecasting

6.3 Use Case - Generalised ABC Storage Analysis

Description
The use case optimises the warehouse storage space and picking efficiencies based on product classification and ABC storage policy.
User Interactions
Developers utilize the Data Tool to perform product classification and design interactive graphs in the visualisation tool to show which products should be stored in which section. They then use the simulation model to test different policy variants and look for the best options. Managers and Supervisors access the visualisation tool to view the configurations. Integration could be done to configure the Warehouse Management System (WMS) with the desired parameters.
Output
Recommendations for product classification and their allocation in storage areas. In case there is an integration with the WMS, the configurations can be manually or automatically realised.

6.4 Use Case - Snapshot Analysis

Description
A snapshot analysis provides a view of how the storage and operations warehouse

respond to given inputs. We can simulate a short-term scenario where one can analyse for example, various KPIs given a list of orders and replenishment tasks for the next day. Parameters such as number of workers and their schedule can be varied to construct different scenarios.
User Interactions
Developers uses the Data Tool to obtain the input data from the data warehouse. The parameters including workers' count and schedule, storage snapshot, incoming orders, and replenishment tasks are fed into simulation model to preview short-term KPIs. Managers and Supervisors get a summary of KPIs from the visualisation tool and recommended actions.
Output
Various KPIs produced by the simulation model under different scenarios along with recommendations.

6.5 Expert Survey

Two experts in warehousing business and Power BI were consulted. The maximum score for a criteria is five. Questions asked during interviews are as follows:

1. It is straightforward to understand what is being forecasted.

2. As a warehouse supervisor, I find it useful to view the trend chart and table for daily workload forecasts of incoming and outgoing pallets.

3. As a warehouse supervisor, I find it useful to see the current storage utilisation (how much percent the storage is occupied) and estimate whether the storage can accept new pallets in the short and medium term with the net incoming pallet table.

4. As a warehouse manager, I find it useful to view and identify quarterly Class A, B, C products with the classification table.

5. It is useful to see the best ABC policy variation in terms of picking time and the travel distance.

6. The best ABC policy recommended can be applied in practise.

7. As a supervisor I can understand the impact on worker utilisation, waiting time, and when assigning 3 workers to the checking activity compared to assigning only 1 worker.

8. Which parts of the visualisation can be improved?

6.5.1 Survey Feedback

No.	Criteria	surveyor1's Feedback	surveyor2's Feedback	Average Score
1	Clarity of forecasts in dashboard pages	4	3.5	3.75
2	Daily workload forecasts usefulness	4.5	4.5	4.25
3	Usefulness of storage utilization view	4	4	4
4	ABC Classification Usefulness	4	3.5	3.75
5	Differentiating storage policies	3.5	4	3.75
6	Real-life applicability	4	4.5	4.25
7	Understanding of the impact of worker allocation and assignments on Utilisation and Congestion	3	3.5	3.25

Table 27: Summary of Expert Survey Feedback

As for the improvements, surveyor 1 mentioned that users could be more customised forecasts with which stakeholders can input a percentage number to alter the forecasts produced by algorithms. If a manager with experience knows an upcoming period of surging demand, the forecasts should also be adjusted in case it underestimated the demand. Moreover, the user interface could be improved with navigation bars and simple descriptions. The resource monitoring part is rather static because it is not interactive. If a supervisor wants to inspect the charts hour by hour in detail, he/she must use the simulation software. Surveyor 2 said that the simulation part could be improved by incorporating a 3D embedded element to show interactive 3D storage inside Power BI. It allows users to view and locate individual ABC products in the 3D model; this gives them a detailed view of the size of ABC sections and cell/slot location for each SKU based on the optimal storage policy. The readability of dashboard can be improved. For instance, it can be improved by changing the names of the columns and tooltips. Furthermore, more textual explanation is needed for the each page of the dashboard for a warehouse manager to understand it.

Overall, the visualisation tool provided warehousing stakeholders with information to help several operational and strategic decision making cases in general. Some improvements are suggested for future works. In addition, when the artefact is to be applied for a specific warehouse, some dedicated pages in Power BI may be needed to suit the type of the warehouse. For example, when applying it on a spare parts warehouse, a new page for the supply and demand of individual products should be built to demonstrate their dynamics and suggestions produced by forecasting service. Another example would be an e-commerce warehouse, where demand is influenced by many societal and environmental factors. A new page could be built to allow users to view how these factors could affect product demand.

Chapter 7

Conclusion

This thesis aimed to answer the main question:

How can a warehouse improve decision making by using data-driven services?

This is answered by RQ1, RQ2, and RQ3, which we discuss with the following three sections respectively.

7.1 Demand Forecast of Picking Workload

This section answers RQ1:

Which forecasting method is the most suitable to predict daily picking demand?

SARIMA was chosen as the best algorithm for our data set, shown in Chapter 3. A data-driven demand forecast is usually done through a statistical or Machine Learning approach. A statistical approach is to take trend and seasonality into consideration. Exponential Smoothing and ARIMA/SARIMA methods are statistical methods with mathematical equations containing trend and seasonal components. LightGBM, in comparison, is a Machine Learning method that bases its predictions on features. It take can take time-related features (such as day of the week) and exogenous features (such as number of open orders) into consideration. We experimented the implementation of the Exponential Smoothing, SARIMA, and LightGBM algorithms in Python using a synthetic data set, and they all showed acceptable error metrics with MAPE lower than 15%, and MASE lower than 0.8. We found that SARIMA to be able to capture both the magnitude and fluctuation of the data while the forecast produced by exponential smoothing and LightGBM did not. We therefore concluded that SARIMA was the most suitable model for daily picking demand forecast. When comparing our findings to some other similar studies, there are some disparities. For example, paper [7] did a case study in case-picking volume demand and showed that LightGBM (with more than 40 features) performed better than ARIMA in short-term forecast. However, the medium term performances (more than 7 days) were similar to ARIMA. Another study [21] (which used statistical methods and no machine learning methods) demonstrated that a composite forecast using exponential smoothing and SARIMA/ARIMA performed the best and showed MAPE as low as 5%. It was because the model captured the demand pattern well with level and trend for both ARIMA and exponential models.

A limitation to this part of research is the fact that the data was synthetic, which means there is no information about business context to help us choose suitable features and algorithms. There is also no exogenous factors such as open orders available in the data set. It is the reason why only time-related features were used in training the LightGBM algorithm. Normally LightGBM has issue dealing with trend and seasonality in a time-series data, but it can be mitigated by using the output of LightGBM as weight to adjust the forecast by trend forecasting models such as ARIMA and Prophet [85]. A future research direction we suggest is to use statistical and Machine Learning methods with a hybrid approach [21, 85] to improve forecast accuracy.

7.2 Storage Simulator

To answer RQ2:

Which factors affect the effectiveness of an ABC storage policy?

Five factors affect the effectiveness of an ABC storage policy: the policy variant, classification/division method, layout, zone size, and real-time assignment methods. A policy variant is how A, B,and C sections are arranged based on the specific indoor environment of a warehouse. Variants may perform differently due to different pick/delivery points. For layout, we should choose suitable storage types: a drive-in storage does not work well with ABC storage policy while rack storage does. For ABC class division methods, frequency is a common parameter used to classify the products, and more advanced methods such as EIQ and COI can also be used depending on the type of warehouse. We also found from studies that ABC storage zone size allocation should be optimised to prevent overstocking, causing products to be stored in another section. The optimal size can be determined by heuristics or a method proposed by [67] with Machine Learning and simulation methods. Last but not least, using real-time assignment methods such as linear programming and affinity methods can further improve picking efficiency.

The next part answers RQ3:

How to model an environment with storage and other areas of a ware- house and measure KPIs including the travel distance and picking time in the picking process?

We applied a combined DES and ABS method to design the simulation model in Chapter 4. The model may be classified based on the framework by paper [8] with the following subjects, as Table 28 shows.

Subject	**Description**	**Justification**
Type of Study	A mix of Type A and B: for specific application and template that can be utilised by other modellers.	We showed an application of measuring several KPIs in a hypothetical warehouse. The model is specified conceptually and the implementation is also described.
Type of Simulation	DES plus partial ABS	The DES part consists of process flows, queue handling, event and schedule generation, etc. The ABS part consists of properties, states and variables of individual agents.
Hybrid Simulation Integration Method	Interaction	AnyLogic uses interaction method [8] to close the gap between DES and ABS.
Data Input Sources	Illustrative	Synthetic Data.
Level of Implementation	Potential for real-life implementation	Other modellers can expand and modify the model and implementation to suit different purposes and level of detail.

Table 28: Classification of our Simulation Model

The simulation model we proposed is suitable for warehouses that deals with bulk SKUs, meaning that all SKUs are palletised and picking a SKU implies picking up a pallet. It is dissimilar to multiple SKU picking where a picker travels to the storage, collects a SKU from a storage bin, and proceeds to the next picking location until all order-lines are collected. Multiple picks are applicable to, for example, an e-commerce which deals with cellphones, medicines, supplements, etc. or to a spare parts warehouse where small spare parts of cars are stored. Single picking is the default picking behaviour in the AnyLogic Material Handling Library. Multiple picks can be custom designed and implemented with Java, but it would require additional programming effort.

In the AnyLogic simulation model, we employed some aspects of ABS method by utilising properties and variables of individual products so that we can assign them to the correct class-based areas in the storage, and it also enables us to measure, for instance, time stamps and states of individual products. A full ABS would suggest that agents have autonomous behaviour and can interact with or react to other agents. These features are feasible to be implemented in AnyLogic, as shown by an example [33]. A full ABS might be suitable for warehouses with AGVs or drones, who are agents with their own logic of searching for a route and avoiding traffic congestion. In our case, full ABS is unnecessary because we do not have any AGVs, and we assume that every forklift follows a pre-defined routing logic with automatic avoidance of obstacles; this allows us to isolate these factors and focus on our KPI objectives.

Our model is similar to study [71] which examines order picking time using multiple SKU picking method with pre-defined routing methods. For future research, our KPIs about worker utilisation and waiting time could be useful if incoming and outgoing SKUs are known upfront in the short term. It allows the user of the simulation tool to detect bottleneck. This is similar to study [20], in which it attempts to allocate workers to maximise inbound and outbound service level. The assignment and allocation of workers found using the method from this study can be tested in the AnyLogic model to simulate the real operations in warehouse. Another feature that could be implemented in our model is schedule generator and control introduced by paper [55] to manage storing and picking tasks.

There are some limitations to the model. For example, the simplification of the designed layout. In a real warehouse, obstacles such as walls and doors should be modelled so that the forklifts will act based on these environmental factors while travelling. In the assessment of the total travel distance and picking time, we did not perform storing process in parallel with picking process because different warehouses may have different replenishment or storing mechanisms and we want to isolate picking from storing. In a real warehouse, they might affect each other and have impact on the picking time. Another limitation to the experiments is that the orders generated are extrapolated from the historic order frequency for a 24-hour period. The KPIs obtained are therefore representative of an average day.

7.3 Use Cases of the Data-driven Services

This section answers RQ4:

How do warehousing stakeholders use the forecast and storage simulation services for decision making?

From the literature review, we observed that few study had touched on how to effectively apply the solutions in the real decision making processes in warehouses particularly in simulation studies. For example, many simulation studies demonstrated the results derived from simulation models but it was unclear whether the solutions offered by the models were actually applied in the real warehouses or how the solutions could be used to benefit stakeholders on a regular basis. To tackle this issue, we designed use cases so that developers and stakeholders can work together to benefit from the services to make better decisions on worker assignment and storage policy. The forecasting and storage simulator services have the following implemented architecture, as Figure 34 shows. A limitation of this architecture is that the data transfers between AnyLogic and Power BI are mostly manual.

7.4 Summary

This thesis addressed two main challenges of warehousing business: picking demand forecast and evaluation of ABC storage policy variants. They are deemed as two data-driven services because first: data is acquired from a data warehouse, which can be refreshed on daily basis to be up-to-date; second: decision-making factors are also based on data generated by forecasting algorithms in forecasting service, and simulation model in the Storage Simulator. For forecasting service, we used SARIMA for our picking demand prediction, and discussed that accuracy could be further improved in the future by combining LightGBM and trend-based models. For Storage Simulator, we used a combined DES and ABS approach to model a warehouse in AnyLogic, allowing us to accurately define and measure the KPIs in the picking process, which then enables us to evaluate the goodness of the policy variants. We also reviewed factors influencing effectiveness of ABC storage policy, which provides directions for future research on optimisation of ABC storage policy. The connection between the two services is that the forecasting service can estimate the number of workers needed to carry out picking tasks in a given time frame. It is especially true for a large warehouse. As shown by [21], forecast was done for different zones in a large warehouse that handles tens of thousands of product units per day. Knowing how many pickers are needed for each zone helps us allocate the right number of pickers in the simulation model to have the simulation environment closer to the reality.

Our recommendation for future research is to expand the forecasting service to serve as a generic data service so that it supports the Storage Simulator with more data as input such as shift and schedule. The Storage Simulator can also be expanded to include features such as layout change analysis and shift optimiser for workers. An improvement could be made for the integration, which should be more automated rather having to transfer data manually to the visualisation tool.

Bibliography

[1] Mohammed Abdelghany and Amr Eltawil. "A Discrete-Event and Agent-Based Hybrid Simulation Approach for Healthcare Systems Modeling and Analysis". In: Mar. 2016.

[2] AndresHG. *Time Series Analysis: A Complete Guide*. Kaggle. 2022. URL: https://www.kaggle.com/code/andreshg/timeseries-analysis-a-complete-guide.

[3] AnyLogic. "Optimizing E-commerceWarehouse Operations". In: (2023). URL: https://www.anylogic.com/resources/case-studies/optimizing-e-commerce-warehouse-operations/.

[4] Unknown Author. *Chapter 4 Exponential Smoothing*. See section 4.4.2 Multiplicative Holt-Winters Method. Unknown Year. URL: https://bookdown.org/fjcc/timeseries/chapter-4-exponential-smoothing.html.

[5] MJ Bahmani. *Understanding LightGBM Parameters (and How to Tune Them)*. 2023. URL: https://neptune.ai/blog/lightgbm-parameters-guide.

[6] Prashant Banerjee. *ARIMA Model for Time Series Forecasting*. Accessed: 2023-09-29. 2021. URL: https://www.kaggle.com/code/prashant111/arima-model-for-time-series-forecasting.

[7] M. L. J. Boer. "A Forecasting Model for Daily Case Picking Volumes in Warehouses". In: (2022). URL: https://research.tue.nl/en/studentTheses/a-forecasting-model-for-daily-case-picking-volumes-in-warehouses.

[8] Sally C. Brailsford et al. "Hybrid simulation modelling in operational research: A state-of-the-art review". In: *European Journal of Operational Research* 278.3 (2019), pp. 721–737. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2018.10.025. URL: https://www.sciencedirect.com/science/article/pii/S0377221718308786.

[9] Jason Brownlee. *Introduction to Time Series Forecasting with Python: How to Prepare Data and Develop Models to Predict the Future*.

[10] Aurelija Burinskiene. "Order picking process at warehouses". In: *International Journal of Logistics Systems and Management - Int J Logist Syst Manag* 6 (Jan. 2010). DOI: 10.1504/IJLSM.2010.030958.

[11] Felix T.S. Chan and H.K. Chan. “Improving the productivity of order picking of a manual-pick and multi-level rack distribution warehouse through the implementation of class-based storage”. In: *Expert Systems with Applications* 38.3 (2011), pp. 2686–2700. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2010.08.058. URL: https://www.sciencedirect.com/science/article/pii/S0957417410008547.

[12] Felix T.S. Chan and H.K. Chan. “Improving the productivity of order picking of a manual-pick and multi-level rack distribution warehouse through the implementation of class-based storage”. In: *Expert Systems with Applications* 38.3 (2011), pp. 2686–2700. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2010.08.058. URL: https://www.sciencedirect.com/science/article/pii/S0957417410008547.

[13] Cheng chen et al. “High season demand forecasting method in logistics warehouse inventory management”. In: *IEICE* F-025 (2020). URL: https://www.ieice.org/publications/conferences/summary.php?id=FIT0000014278&expandable=2&ConfCd=F&session_num=6d&lecture_number=F-025&year=2020&conf_type=F.

[14] Tianqi Chen and Carlos Guestrin. “XGBoost: A Scalable Tree Boosting System”. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2016.

[15] Kelsey Clements et al. “Evaluation of warehouse bulk storage lane depth and ABC space allocation using simulation”. In: *2016 Winter Simulation Conference (WSC)*. 2016, pp. 2239–2249. DOI: 10.1109/WSC.2016.7822265.

[16] Kelly Derickx. “A comparative study of different storage policies in warehouse management”. In: (2012).

[17] Negri E. and Fumagalli L.and Macchi M. “A Review of the Roles of Digital Twin in CPS-based Production Systems”. In: *Procedia Manufacturing* 11 (2017), pp. 939–948. DOI: https://doi.org/10.1016/j.promfg.2017.07.198.

[18] Michael P. Foley. *Chapter 4 Exponential Smoothing*. See sections on Additive and Multiplicative Holt-Winters Method for information on the trend component. 2023. URL: https://bookdown.org/mpfoley1973/time-series/exponential.html.

[19] Aidan Fuller et al. “Digital Twin: Enabling Technologies, Challenges and Open Research”. In: *IEEE Access* 8 (2020), pp. 108952–108971. DOI: 10.1109/ACCESS.2020.2998358.

[20] Odkhishig Ganbold et al. “A Simulation-Based Optimization Method for Warehouse Worker Assignment”. In: *Algorithms* 13.12 (Dec. 2020), p. 326. ISSN: 1999-4893. DOI: 10.3390/a13120326. URL: http://dx.doi.org/10.3390/a13120326.

[21] Teun van Gils et al. “The use of time series forecasting in zone order picking systems to predict order pickers’ workload”. In: *International Journal of Production Research* 55.21 (2017), pp. 6380–6393. DOI: 10.1080/00207543.2016.1216659. URL: https://doi.org/10.1080/00207543.2016.1216659.

[22] M. Grieves and J. Vickers. “Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In Transdisciplinary Perspectives on Complex Systems”. In: *Springer* (2017), pp. 85–113. DOI: https://doi.org/10.1007/978-3-319-38756-7_4.

[23] M. Grieves and J. Vickers. “Origins of the digital twin concept”. In: 2016, pp. 2798–2810.

[24] Jinxiang Gu, Marc Goetschalckx, and Leon F. McGinnis. “Research on warehouse operation: A comprehensive review”. In: *European Journal of Operational Research* 177.1 (2007), pp. 1–21. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2006.02.025. URL: https://www.sciencedirect.com/science/article/pii/S0377221706001056.

[25] Zhang He and Sun Yu. “Application of LightGBM and LSTM combined model in vegetable sales forecast”. In: *Journal of Physics: Conference Series* 1693.1 (Dec. 2020), p. 012110. DOI: 10.1088/1742-6596/1693/1/012110. URL: https://dx.doi.org/10.1088/1742-6596/1693/1/012110.

[26] Charles C. Holt. “Forecasting seasonals and trends by exponentially weighted moving averages”. In: *International Journal of Forecasting* 20.1 (2004), pp. 5–10. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2003.09.015. URL: https://www.sciencedirect.com/science/article/pii/S0169207003001134.

[27] N.J. Huitink. “Workforce Prediction of Order Picking Personnel”. In: (Aug. 2020). URL: http://essay.utwente.nl/83027/.

[28] Rob J Hyndman and George Athanasopoulos. *Holt-Winters’ seasonal method.* In Forecasting: Principles and Practice (2nd ed). 2018. URL: https://otexts.com/fpp2/holt-winters.html.

[29] W. Inmon. “The data warehouse and data mining”. In: *Communications of the ACM* 39 (11 1996), pp. 49–50. DOI: 10.1145/240455.240470.

[30] Milan Jemelka et al. “ABC analyses with recursive method for warehouse”. In: *2017 4th International Conference on Control, Decision and Information Technologies (CoDIT)*. 2017, pp. 0960–0963. DOI: 10.1109/CoDIT.2017.8102722.

[31] Abby Jenkins. “55 Expert Warehouse Order Picking Tips and Best Practices”. In: (2021). URL: https://www.netsuite.com/portal/resource/articles/ecommerce/warehouse-order-picking-tips.shtml.

[32] Nick Cox Jeremias K. *Determine best ARIMA model with AICC and RMSE.* 2016. URL: https://stats.stackexchange.com/questions/219605/determine-best-arima-model-with-aicc-and-rmse.

[33] Edward Junprung and Sahar Esmaeilzadeh. “Automated Guided Vehicle (AGV) Powered by AI”. In: (2023). URL: https://cloud.anylogic.com/model/bc88f3be-b5a6-4962-abc1-a26def59994d?mode=SETTINGS.

[34] Jan Karasek. “An Overview of Warehouse Optimization”. In: *International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems* 2.3 (2013), pp. 111–117. ISSN: 1805-5443. DOI: 10.11601/ijates.v2i3.61. URL: http://www.ijates.org/index.php/ijates/article/view/61.

[35] Dirk Kauke, Stefan Galka, and Johannes Fottner. “Digital Twins in Order Picking Systems for Operational Decision Support”. In: Jan. 2021. DOI: 10.24251/HICSS.2021.200.

[36] Vivek Khanzode and Bhavin Shah. “A comprehensive review of warehouse operational issues”. In: *International Journal of Logistics Systems and Management* 26 (Jan. 2017), p. 346. DOI: 10.1504/IJLSM.2017.10002597.

[37] Thai Young Kim, Rommert Dekker, and Christiaan Heij. “Improving warehouse labour efficiency by intentional forecast bias”. In: *International Journal of Physical Distribution and Logistics Management* 48.1 (2018). Cited by: 12; All Open Access, Green Open Access, pp. 93–110. DOI: 10.1108/IJPDLM-10-2017-0313. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85040256873&doi=10.1108%2fIJPDLM-10-2017-0313&partnerID=40&md5=fd57b798e61b4dc8a9490436e1150772.

[38] S. Kolassa. “Combining exponential smoothing forecasts using Akaike weights”. In: *International Journal of Forecasting* 27.2 (2011), pp. 238–251.

[39] Benjamin Korth, Christian Schwede, and Markus Zajac. “Simulation-ready digital twin for realtime management of logistics systems”. In: *2018 IEEE International Conference on Big Data (Big Data)*. 2018, pp. 4194–4201. DOI: 10.1109/BigData.2018.8622160.

[40] R. d. Koster, T. Le-Duc, and K. J. Roodbergen. “Design and control of warehouse order picking: a literature review”. In: *European Journal of Operational Research* 182 (2 2007), pp. 481–501. DOI: 10.1016/j.ejor.2006.07.009.

[41] Nikolaos Kourentzes, Juan R. Trapero, and Devon K. Barrow. “Optimising forecasting models for inventory planning”. In: *International Journal of Production Economics* 225 (2020), p. 107597. ISSN: 0925-5273. DOI: https://doi.org/10.1016/j.ijpe.2019.107597. URL: https://www.sciencedirect.com/science/article/pii/S0925527319304323.

[42] Michael E. Kuhl et al. “Warehouse Digital Twin: Simulation Modeling and Analysis Techniques”. In: *2022 Winter Simulation Conference (WSC)* (2022), pp. 2947–2956. URL: https://api.semanticscholar.org/CorpusID:256220070.

[43] Gilbert Laporte. “The traveling salesman problem: An overview of exact and approximate algorithms”. In: *European Journal of Operational Research* 59.2 (1992), pp. 231–247. ISSN: 0377-2217. DOI: https://doi.org/10.1016/0377-2217(92)90138-Y. URL: https://www.sciencedirect.com/science/article/pii/037722179290138Y.

[44] Jiewu Leng et al. “Digital twin-driven joint optimisation of packing and storage assignment in large-scale automated high-rise warehouse product-service system”. In: *International Journal of Computer Integrated Manufacturing* 34.7-8 (2021), pp. 783–800. DOI: 10.1080/0951192X.2019.1667032. eprint: https://doi.org/10.1080/0951192X.2019.1667032. URL: https://doi.org/10.1080/0951192X.2019.1667032.

[45] Jiaxi Li and Mohsen Moghaddam. “Dynamic storage assignment with product affinity and ABC classification—a case study”. In: vol. 84. 2016, pp. 2179–2194. URL: https://link.springer.com/article/10.1007/s00170-015-7806-7#citeas.

[46] Augustyn Lorenc and Tone Lerher. “Effectiveness of Product Storage Policy According to Classification Criteria and Warehouse Size”. In: (2018). URL: http://scindeks-clanci.ceon.rs/data/pdf/1451-2092/2019/1451-20921901142L.pdf.

[47] Shan Lu, Hongyue Li, and Xiaojun Yu. “The Application of EIQ Analysis to The Order Picking of Book Industry”. In: *Proceedings of the International Conference on Logistics, Engineering, Management and Computer Science*. Atlantis Press, 2015/07, pp. 93–97. ISBN: 978-94-6252-102-5. DOI: 10.2991/lemcs-15.2015.18. URL: https://doi.org/10.2991/lemcs-15.2015.18.

[48] Scott M Lundberg and Su-In Lee. “A Unified Approach to Interpreting Model Predictions”. In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc. 2017.

[49] Yuri Merkuryev, Aurelija Burinskiene, and Galina Merkuryeva. “Warehouse Order Picking Process”. In: Jan. 2009, pp. 147–165. ISBN: 978-1-84882-186-6. DOI: 10.1007/978-1-84882-187-3_9.

[50] Bhoomica Nataraja. “Simulation of storage strategies for a forward-reserve warehouse”. In: (2019). URL: https://research.tue.nl/en/studentTheses/simulation-of-storage-strategies-for-a-forward-reserve-warehouse.

[51] N.C. Nielsen. *AR Racking - Drive In compact racking system*. 2023. URL: https://www.nc-nielsen.com/ar-racking-drive-in-racking-system.

[52] Kannan Nilakantan. “Replenishment policies for warehouse systems under cyclic demand”. In: *International Journal of Business Performance and Supply Chain Modelling* 5 (Jan. 2013). DOI: 10.1504/IJBPSCM.2013.053491.

[53] Michael J. North and Charles M. Macal. *Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation*. Oxford University Press, Apr. 2007. ISBN: 9780195172119. DOI: 10.1093/acprof:oso/9780195172119.001.0001. URL: https://doi.org/10.1093/acprof:oso/9780195172119.001.0001.

[54] Fariborz Y Partovi and Murugan Anandarajan. “Classifying inventory using an artificial neural network approach”. In: *Computers Industrial Engineering* 41.4 (2002), pp. 389–404. ISSN: 0360-8352. DOI: https://doi.org/10.1016/S0360-8352(01)00064-X. URL: https://www.sciencedirect.com/science/article/pii/S036083520100064X.

[55] Pawel Pawlewski. “DES/ABS Approach to Simulate Warehouse Operations”. In: *Highlights of Practical Applications of Agents, Multi-Agent Systems, and Sustainability - The PAAMS Collection*. Ed. by Javier Bajo et al. Cham: Springer International Publishing, 2015, pp. 115–125. ISBN: 978-3-319-19033-4.

[56] C. G. Petersen. “The impact of routing and storage policies on warehouse efficiency”. In: *International Journal of Operations Amp; Production Management* 19 (10 1999), pp. 1053–1064. DOI: 10.1108/01443579910287073.

[57] Benjamin Kai Pierre et al. “DYNAMIC ABC STORAGE POLICY IN ERRATIC DEMAND ENVIRONMENTS”. In: 2004. URL: https://api.semanticscholar.org/CorpusID:55370916.

[58] Anindita Putri and Bayu Wahyudi. “Design of Performance Indicators in Warehouse Management”. In: *Indikator: Jurnal Ilmiah Manajemen dan Bisnis* 7 (Jan. 2023), p. 73. DOI: 10.22441/indikator.v7i1.17843.

[59] F. Abd. Razak et al. “Load Forecasting Using Time Series Models”. In: *Jurnal Kejuruteraan* 21.1 (2009), pp. 53–62.

[60] Gwynne Richards. “Warehouse processes: pick prepration”. In: *Warehouse Management: A complete guide to improving efficiency and minimizing costs in the modern warehouse*. KoganPage, 2017.

[61] Stewart Robinson. “Conceptual modelling for simulation Part II: A framework for conceptual modelling”. In: *Journal of the Operational Research Society* 59 (Mar. 2008), pp. 291–304. DOI: 10.1057/palgrave.jors.2602369.

[62] Kees Jan Roodbergen. “Storage Assignment for Order Picking in Multiple-Block Warehouses”. In: (2012). Ed. by Riccardo Manzini, pp. 139–155. DOI: 10.1007/978-1-4471-2274-6_7. URL: https://doi.org/10.1007/978-1-4471-2274-6_7.

[63] Kees Jan Roodbergen and René De Koster. “Routing order pickers in a warehouse with a middle aisle”. In: *European Journal of Operational Research* 133 (Aug. 2001), pp. 32–43. DOI: 10.1016/S0377-2217(00)00177-6.

[64] Daniel J Rosenkrantz, Richard E Stearns, and Philip M Lewis. “An analysis of several heuristics for the traveling salesman problem”. In: *SIAM journal on computing* 6.3 (1977), pp. 563–581.

[65] A. Al-Saadi, W. Al-Saadi, and J. M. Guerrero. “Microgrid Digital Twins: Concepts, Applications, and Future Trends. IEEE Transactions on Industry Applications”. In: *Springer* (2021). DOI: https://doi.org/10.1109/TIA.2021.3107980.

[66] Sumit Saha. *XGBoost vs LightGBM: Which Algorithm Wins in a Fight?* Accessed: yyyy-mm-dd. 2023. URL: https://neptune.ai/blog/xgboost-vs-lightgbm.

[67] Allyson Silva et al. “Estimating optimal ABC zone sizes in manual warehouses”. In: *International Journal of Production Economics* 252 (2022), p. 108579. ISSN: 0925-5273. DOI: https://doi.org/10.1016/j.ijpe.2022.108579. URL: https://www.sciencedirect.com/science/article/pii/S0925527322001682.

[68] Taylor G. Smith et al. *pmdarima: ARIMA estimators for Python.* 2017–. URL: http://www.alkaline-ml.com/pmdarima.

[69] *Stata Time-Series Reference Manual.* See section on Seasonal Holt-Winters methods. URL: https://www.stata.com/manuals/tstssmoothshwinters.pdf.

[70] Ashik Talupula. "Demand Forecasting Of Outbound Logistics Using Machine learning". In: (2018). URL: http://www.diva-portal.org/smash/get/diva2:1367098/FULLTEXT02.pdf.

[71] Grzegorz Tarczyński. "Warehouse Real-Time Simulator – How to Optimize Order Picking Time". In: (2013). DOI: https://dx.doi.org/10.2139/ssrn.2354827. URL: https://ssrn.com/abstract=2354827.

[72] Tran Thanh and Le Van Dai. "Grid search of exponential smoothing method: a case study of Ho Chi Minh City load demand". In: *Indonesian Journal of Electrical Engineering and Computer Science* 19 (May 2020). DOI: 10.11591/ijeecs.v19.i3.pp1121-1130.

[73] Korrakot Yaibuathet Tippayawong, Apichat Sopadang, and Patchanee Patitad. "Improving warehouse layout design of a chicken slaughterhouse using combined ABC class based and optimized allocation techniques". In: 2013. URL: https://api.semanticscholar.org/CorpusID:16222836.

[74] Korrakot Yaibuathet Tippayawong, Apichat Sopadang, and Patchanee Patitad. "Improving warehouse layout design of a chicken slaughterhouse using combined ABC class based and optimized allocation techniques". In: 2013. URL: https://api.semanticscholar.org/CorpusID:16222836.

[75] Ngoc Cuong Truong, Truong Giang Dang, and Duy Anh Nguyen. "Building Management Algorithms in Automated Warehouse Using Continuous Cluster Analysis Method". In: *AETA 2017 - Recent Advances in Electrical Engineering and Related Sciences: Theory and Application.* Ed. by Vo Hoang Duy et al. Cham: Springer International Publishing, 2018, pp. 1068–1077. ISBN: 978-3-319-69814-4.

[76] Metasebiya Tsige. "Improving order-picking efficiency via storage assignments strategies". In: 2013. URL: https://api.semanticscholar.org/CorpusID:17291058.

[77] René van Bevern and Viktoriia A. Slugina. "A historical note on the 3/2-approximation algorithm for the metric traveling salesman problem". In: *Historia Mathematica* 53 (2020), pp. 118–127. ISSN: 0315-0860. DOI: https://doi.org/10.1016/j.hm.2020.04.003. URL: https://www.sciencedirect.com/science/article/pii/S0315086020300240.

[78] Nicolas Vandeput. *Data Science for Supply Chain Forecasting.* Walter de Gruyter, 2021.

[79] Jesús Vázquez-Serrano, Rodrigo Peimbert-García, and Leopoldo Cárdenas-Barrón. "Discrete-Event Simulation Modeling in Healthcare: A Comprehensive Review". In: *International Journal of Environmental Research and Public Health* 18 (Nov. 2021), p. 12262. DOI: 10.3390/ijerph182212262.

[80] Sven Winkelhaus et al. “Hybrid order picking: A simulation model of a joint manual and autonomous order picking system”. In: *Computers Industrial Engineering* 167 (2022), p. 107981. ISSN: 0360-8352. DOI: https://doi.org/10.1016/j.cie.2022.107981. URL: https://www.sciencedirect.com/science/article/pii/S0360835222000511.

[81] P. R. Winters. “Forecasting sales by exponentially weighted moving averages”. In: *Management Science* 6 (3 1960), pp. 324–342. DOI: 10.1287/mnsc.6.3.324.

[82] Yugang Yu Xiaolong Guo and René B.M. De Koster. “Impact of required storage space on storage policy performance in a unit-load warehouse”. In: *International Journal of Production Research* 54.8 (2016), pp. 2405–2418. DOI: 10.1080/00207543.2015.1083624. eprint: 3.2015.1083624. URL: https://doi.org/10.1080/00207543.2015.1083624.

[83] Charles Zaiontz. *Holt-Winters Additive Method*. 2021. URL: https://www.real-statistics.com/time-series-analysis/forecasting-smoothing/holt-winters-additive-method/.

[84] Tong Zhou. “Improved Sales Forecasting using Trend and Seasonality Decomposition with LightGBM”. In: (May 2023).

[85] Tong Zhou. “Improved Sales Forecasting using Trend and Seasonality Decomposition with LightGBM”. In: *2023 6th International Conference on Artificial Intelligence and Big Data (ICAIBD)*. 2023, pp. 656–661. DOI: 10.1109/ICAIBD57115.2023.10206380.

[86] Margareta Živičnjak, Kristijan Rogić, and Ivona Bajor. “Case-study analysis of warehouse process optimization”. In: *Transportation Research Procedia* 64 (2022). International Scientific Conference “The Science and Development of Transport - Znanost i razvitak prometa”, pp. 215–223. ISSN: 2352-1465. DOI: https://doi.org/10.1016/j.trpro.2022.09.026. URL: https://www.sciencedirect.com/science/article/pii/S2352146522006408.

Appendix A

A.1 LightGBM Parameters

Below is the most important parameters in LightGBM according to[5].

Parameter	Description
objective	Specifies the optimization objective during the training process, e.g., binary, regression.
metric	Defines the evaluation metric to be used for model evaluation, e.g., auc, binary_error, binary_logloss, etc.
boosting	Specifies the boosting type to be used, e.g., gbdt (traditional Gradient Boosting Decision Tree), dart (Dropouts meet Multiple Additive Regression Trees), goss (Gradient-based One-Side Sampling), etc.
lambda_l1	Controls L1 regularization, helping to avoid overfitting by penalizing complex models.
bagging_fraction	Specifies the fraction of data to be used for each boosting iteration, also known as subsampling.
bagging_freq	Defines the frequency of bagging, i.e., performing bagging every k iterations.
num_leaves	Sets the maximum number of leaves for each weak learner (tree).
feature_fraction	Specifies the fraction of features to be used for each boosting iteration, also known as column subsampling.
max_depth	Controls the maximum depth of each trained tree.
max_bin	Specifies the maximum number of bins that feature values will be bucketed into, which is crucial for the histogram-based training method.
num_iterations	Specifies the number of boosting iterations, i.e., the number of trees to build.
learning_rate	Sets the step size shrinkage to prevent overfitting, commonly set between 0.01 and 0.3.
early_stopping_rounds	Stops training if the validation metric does not improve for a specified number of rounds.
categorical_feature	Specifies the categorical features in the dataset.
verbosity	Controls the level of detail in the output generated during model training, e.g., 0 (silent), 1 (warning), 2 (info), etc.
min_data_in_leaf	Sets the minimum number of data points required in a leaf to continue splitting, helping to prevent overfitting.

Table 29: Descriptions of LightGBM Parameters

A.2 Simulation Model

This section contains topics related to the AnyLogic Simulation Model.

A.2.1 Order Arrival Schedule

Below is the arrival schedule of orders for the two experiments.

Date & Time	SKU Class	SKU ID	Number of SKU Ordered
7/24/2023 9:00	A	0	6
7/24/2023 9:00	A	1	5
7/24/2023 9:01	A	0	3
7/24/2023 10:00	A	2	5
7/24/2023 10:01	A	1	5
7/24/2023 10:02	A	0	6
7/24/2023 11:00	B	4	5
7/24/2023 11:01	A	1	5
7/24/2023 11:02	B	4	5
7/24/2023 12:00	A	2	5
7/24/2023 12:01	B	5	15
7/24/2023 12:02	C	6	5
7/24/2023 13:00	B	3	4
7/24/2023 13:01	C	6	2
7/24/2023 13:02	A	1	5
7/24/2023 14:00	A	0	7
7/24/2023 14:01	A	2	5
7/24/2023 14:02	B	4	5
7/24/2023 14:03	A	0	8
7/24/2023 15:00	C	6	4
7/24/2023 16:00	A	2	5
7/24/2023 16:01	A	1	3
7/24/2023 17:00	B	3	4
7/24/2023 18:00	B	4	5
7/24/2023 18:01	A	1	5
7/24/2023 19:00	A	0	9
7/24/2023 20:00	B	3	3
7/24/2023 20:01	A	2	5
7/24/2023 21:01	B	3	4
7/24/2023 22:00	A	1	3
7/24/2023 22:01	B	3	4
7/24/2023 23:00	A	0	11
7/25/2023 0:00	A	2	3
7/25/2023 0:01	A	1	5

Table 30: Scheduled SKU Order Arrival