OPTIMIZING THE ARRANGEMENT OF PRODUCTS WITHIN THE BEST LOCATION OF A WAREHOUSE, TO MAXIMIZE SPACE UTILIZATION

A.A.A.S.Abeydeera

(IT21822780)

BSc (Hons) degree in Information Technology
Specializing in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology

August 2025

OPTIMIZING THE ARRANGEMENT OF PRODUCTS WITHIN THE BEST LOCATION OF A WAREHOUSE, TO MAXIMIZE SPACE UTILIZATION

A.A.A.S.Abeydeera

(IT21822780)

BSc (Hons) degree in Information Technology
Specializing in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology

August 2025

DECLARATION

I declare that this is my work. This proposal does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any other university or institute of higher learning. To the best of my knowledge and belief, it does not contain any previously published material written by another person except where the acknowledgment is made in the text.

Name	Student ID	Signature
A.A.A.S.Abeydeera	IT21822780	

The above candidate is carrying out research for the undergraduate Dissertation under my supervision.

Signature of the supervisor: _____ Date _____

Signature of the co-supervisor: _____ Date _____

ABSTRACT

This project introduces a warehouse optimization system that aims to maximize space use by placing products systematically. The product arrangements are determined using the Best-Fit algorithm, which sorts based on the best possible placement of products into their spaces. The Best-Fit algorithm will allow the warehouse to reduce unnecessary unusable space while maximizing valuable space. The system enables the selection of specific products to space requirements based on their size and features, how often each product is picked, and warehouse constraints, while ensuring that the products are placed in areas that balance both access and density.

The solution includes 3D visualizations to provide a visual depiction of a warehouse's available space, and how it is being used at that time. The tool gives managers insight into continual space distribution throughout the warehouse and shows under-utilized and congested areas of space with real-time updating. This kind of visualization expedites decision making and aids operational planning in a direct method compared to older methods that relied on words and numbers only.

This warehouse optimization project would be created with Python, the open source programming language that has libraries related to optimization, and graphical displays using 3D rendering libraries. This programming with Python is contributed to the solution being scalable, low cost, and customizable to fit many different types of warehouses. It is more efficient to have an algorithm that focuses solely on the organization of space rather than relying on humans organizing space using numbers and letters. This reduced human error whilst maximizing the throughput of the operation and thus the usage of limited warehouse space.

Keywords: Warehouse optimization, Best-Fit algorithm, Space utilization, Product arrangement, CBM (Cubic Meter Measurement), 3D visualization, Python, Inventory management, Logistics efficiency, Warehouse simulation, Computational optimization, Decision support system, Storage density, Demand-based storage, Supply chain optimization.

ACKNOWLEDGEMENT

First I would like to express my deepest gratitude to my supervisor, **Prof. Samantha Rajapaksha**, for his continuous guidance, encouragement, and invaluable feedback throughout this project. I also wish to thank **Dr. Dinuka Wijendra** for his support and constructive suggestions that helped refine this work. Finally, my sincere appreciation goes to the **academic staff of SLIIT – Faculty of Computing, Department of Information Technology**, for providing the knowledge, resources, and infrastructure necessary to complete this research.

TABLE OF CONTENTS

TABLE OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Definition
AI	Artificial Intelligence
CBM	Cubic Meter Measurement: A unit of volume used in logistics to measure product space utilization.
WMS	Warehouse Management System: Software that manages warehouse operations including storage, retrieval, and tracking.
BFA	Best-Fit Algorithm: A heuristic algorithm used to allocate products into the most suitable available space.
3D	Three-Dimensional: Graphical representation of warehouse space and product arrangements.
Flask	A lightweight Python web framework used to develop web applications and APIs.
API	Application Programming Interface: A set of protocols that enable communication between different software applications.
py3dbp	Python 3D Bin Packing: A Python library for simulating and visualizing 3D space utilization and product arrangement.
DBMS	Database Management System: Software for storing, retrieving, and managing structured data.

1 List of abbreviations

1. INTRODUCTION

Warehouses represent the cornerstone of contemporary supply chains as the uncontrolled storage location of raw materials, finished products and primary storage. As companies continue to scale out, warehouse operators face the daunting task of understanding how to make the best use of limited storage capacity. Inefficiently configured space is not only a wast of cubic capacity, but it also slows retrieval of product, increases labour costs and impacts operational efficiencies.

In practice; many warehouses have not advanced past traditional storage methodologies - fixed slotting, or simply throwing products on a shelf if and when space is available. While these methodologies are simple, they rarely achieve maximum space use and often lead to underutilized shelf or rack space. With continued growth in demand and a compression of order fulfilment time there is substantial opportunity to implement more sophisticated solutions that rationalizes product identify based on size and demand respectively while also maximizing cubic meter (CBM) utilization.

This project aims to address this issue using the Custom algorithm to optimize the arrangement of products in a warehouse. The algorithm evaluates the items going into inventory and allocates them to the most appropriate storage locations, minimizing unused space while allowing items remain accessible. In developing this solution, a 3D visualization element is included so the warehouse manager can easily visualize how space is being utilized and consider alternative storage arrangement strategies in real time.

The system is implemented in Python using general libraries, optimization, and visualization libraries like py3dbp for 3D bin packing. By taking advantage of the precise way an algorithm will allocate items and the interactivity of a visualization, the research and end result of this project intends to provide a solution that will efficiently optimize storage while also providing a practical solution to warehouse operations. Overall, this project will demonstrate how computational methods can be applied to one of the simplest, yet most difficult logistical problems of making better use of warehouse space, while considering usability and efficiency in the solution.

1.1 Background

Warehousing continues to rise in importance as a key feature within the logistics and supply chain activities of businesses today. As organizations grow, the need for storage also grows, which frequently results in warehouses operating at greatest limitations. One area that does create a repeatable problem within such an environment relates to space wastage associated with poor product organization or arrangement. Often, the cubic space exists and simply is not utilized. Even somewhere with enough cubic capacity and space, poor placement of products can result in wasted space, extra costs, and slow work processes [1].

Warehouses have traditionally neglected better storage practices using simple storage practices e.g., dedicated storage where each product has a designated slot, to random storage based on wherever there is free space. Often, a combination of random storage techniques and ABC classification based on frequency of demand is utilized. Once again, it is better than nothing and may provide some warmth in terms of organization, this still has not accounted for cubic meter (CBM) consumption or allocation as it relates to the changing demand, the changing dimensions of the product, and the changing patterns of demand [2].

To solve this problem, researchers and practitioners have turned to optimization approaches. The Best-Fit algorithm, for instance, and its variants are often used for the Bin Packing Problem (BPP), where the objective is to place items in containers to reduce wasted space [3][4]. With a similar rationale for warehouses, products can be placed in the most optimal place to achieve greater density with practical retrieval [5].

Over the last few years, there has been even more improvement to warehouse management processes, owing to whole 3D visualizations. 3D representations provide an overview of how racks, shelves, and products occupy space and, unlike the standard 2D representation, makes storage space easier to visualize. Tools, like py3dbp, a Python 3D bin packing library, allow warehouse managers to not only measure utilization, but visualize what utilization looks like in real time, which makes it even easier for warehouse managers to detect inefficiencies in their visual representation,

and possibly rearrangement can start dynamically instead of changing things in practice [6][7].

The synergy between algorithmic optimization and visual feedback can change the way warehouses are operated. Rather than plan based on historical placements and knowledge solely, warehouse managers can leverage data analysis to organize products more efficiently, lessen unused space, and enable faster retrieval. Warehouse operating costs are increasing, urban warehousing availability is dwindling, and consumer expectations for faster delivery are rising. As a result, optimizing consumption or usage of space in a warehouse using such an approach will be important [8].

The proposed solution builds on these concepts by employing Best-Fit placement methods in conjunction with CBM-based consideration and 3D visualization to produce a pragmatic, scalable optimization approach for warehouse space usage.

1.2 Literature Survey

This chapter summarizes the main contributions to the research in warehouse optimization, addressing the four components of product layout policies, utilization through CBM, optimization methods, and three-dimensional visualizations of product placement. These four areas constitute a foundation for the future work toward the design of an integrated system to support and maximize warehouse space and utilization through optimization algorithms and visualization in real-time.

1.2.1 Product Arrangement Strategies

For many years, warehouse operations have utilized a host of storage techniques (dedicated storage, random storage, and class-based inventory assignment-i.e. Each product is placed in a location based on its class (e.g. ABC-analysis)). Bartholdi and Hackman [1] reported how such techniques maintain arrangement and accessibility but may underutilize capacity due to fluctuation in product characteristics and demand. Recent studies (e.g. Angulo & Fajardo, 2021 [2]) have heavily

emphasized the use of dynamic slotting and rearranged configurations to improve retrieval speed and reduce wasted capacity [11].

1.2.2 CBM Utilization

Cubic Meter Measurement (CBM) has grown to be a standard metric amongst logistics and warehousing for assessing space utilization efficiencies. It is especially critical for high-density storage operations where the price of storage space in metropolitan settings, along with the assessment of how efficiently CBM is utilized, is directly proportional to operational efficiency and profitability. When evaluating the amount of cubic space occupied by the product, versus the total amount of cubic space available, management can begin to realize the unused "gaps" and reconfigure product placement to achieve added density. There are also studies that show that warehouses who utilize CBM measurement and optimize the asset management of cubic volume occupied do not only bring down the cost of storage, but they also see improved order consolidation and container-load efficiencies throughout the warehouse [1][3].

Although CBM represents an important space-efficient metric, most of the relevant literature has focused on picking efficiencies and retrieval time, whereas the volumetric scope of utilizations efforts and the improvement of space utilization performance has been neglected. The majority of slotting methodologies, such as ABC classification, improve accessing ease; yet they do not guarantee that the product of varied shapes and sizes are successfully utilizing space allocated for storage [12]. This underscores the important research gap towards using algorithms that ensure CBM optimization as the underlying premise behind storage optimization decisions [4]. The integration of CBM in a warehouse layout is desirable as it helps ensure that space utilization and operational flow can be optimized simultaneously, making it an essential spatial dimension of modern warehouse space utilization optimization [5].

1.2.3 Optimization Algorithms

The Bin Packing Problem (BPP) is one of the most researched problem areas in operations research and serves as a direct analogy to employee storage assignment in a warehouse. The BPP is concerned with fitting a number of things (of different

sizes) into a minimum number of bins or containers, with each bin or container having a defined capacity. The Best Fit heuristic is also one of the most popular heuristics for solving the NP-hard BPP. Best Fit takes each item and places it in the fullest bin/slot that can still fit. This minimizes wasted space in each container [3][4]. A great number of researchers, notably Martello and Toth [5], have promoted the use of Best Fit in logistics as it attempts to balance solution quality (optimally packed containers) with computational effort.

In recent years, these algorithms have [perhaps it is better to drop 'these' which refers specifically to the algorithms but suggests there are others that do not apply to the idea of a warehouse] moved from theoretical models to actualized warehouse applications. Singh and Sharma [6] have shown that using bin packing algorithms in combination with visualization allows for better overall space utilization while maintaining access to weight-sensitive products. In addition to reducing the number of racks or slots, these algorithms allow for [to] dynamically managing the location of physical product as inventory and demand change. This demonstrates that heuristic algorithms such as Best-Fit, when incorporated with modern visualization and data systems, can lead to a successful reconciliation between mathematical optimization and operational management in an actual warehouse [1][5].

1.2.4 Visualization in Warehouse Management

Because conventional 2D layout drawings do not give any understanding of how the space was used, 3D visualization and simulation have become a real method to further analyze space usage. By simulating racks, shelves, and products in three-dimensions, a manager can see and understand utilization better, and make real-time decisions. Bortolini et al. [7] illustrate the time and cost savings of layout creation when using a virtual simulation. The libraries in py3dbp [6] provide a way to take this even further in a warehousing space by combining optimization with the ability to see three-dimensional visualizations when originally solving the proposed problem of layout optimization; in this way, it can be both empirical and evaluative in character [14].

1.3 Research Gap

While warehouse optimization has been extensively analyzed, research has been closely related to retrieval efficiency and order picking, rather than maximizing cube storage. While Best-Fit and bin packing heuristics are frequently discussed in the literature, they are seldom seen in the practical warehouse environment; similarly, there has been limited progress in developing tools that provide managers with a clear 3D visualization of product space, easing the arrangement process in real-time, and providing new methodologies to see the product arrangement at their discretion. Usability and accessibility in arc work to solve theoretical problems is also an injustice to the daily warehouse operators, providing solutions that are not functional; existing systems seldom guide warehouse operators with realistic and easy to follow solutions. This project proposes to provide a better avenue of merging cube space/space optimization, bin packing strategies, and real-time interactive 3D into a system that works and has one practical framework or workstation that functions independently.

1.3.1 Utilization of Warehouse CBM

As the majority of warehouse management research has been concerned with retrieval speed or order picking, the aspect of cubic meter measurement (CBM) utilization has been neglected. This results in many warehouses not able to take advantage of their full storage potential and wasted space, adding cost to their operations as a result. Very few have researched methods to address the maximum CBM performance of products arrangements [4].

1.3.2 Limited Application of Optimization Algorithms in Practice

Optimization techniques, like the Bin Packing Problem (BPP), have been researched comprehensively in operations/production research fields [3][4] but their industrial application in warehousing practices is very limited. The Best-Fit Algorithm is one of the most popular BPP heuristics and it works by filling each item into the most filled location it fits, preserving the empty spaces from the product. Theoretically this is efficient, and you will see the organization of products is more efficient and least wasted space than using the simpler strategies of First-Fit or Random-Fit.

While the Best-Fit is theoretically efficient, operational constraints in warehouse practice usually stop practitioners from using Best-Fit directly. Many studies with Best-Fit and related algorithms just treat them purely as mathematical optimizations, where you ignore the operational fact that you can maximize storage use, but by locating products inefficiently from an operational access perspective [4]. For example, the algorithm may pack very closely to minimize wasted space, but it may cause (read) more wasted time for a staff member who retrieves the product. The unequal relationship of compromising usability for space savings has made it impractical for widespread use in practice [5].

Additionally, current research often implements Best-Fit to controlled or simulated datasets, instead of the dynamic warehouse environment where product sizes, demand patterns, and inventory levels constantly change. Therefore, while Best-Fit demonstrates significant potential for CBM optimization, little research has explored the integration of Best-Fit on flexible and real-time warehouse management systems [5]. This research gap demonstrates the demand for solutions that optimize space using algorithms such as Best-Fit, while allowing flexibility and practicality for operative day-to-day warehouse operations.

1.3.3 Lack of Real-Time Visualization Tools

While layout optimization methods can be found, they are mainly either theoretical or are limited to 2D layout representations ([7]). There is an absence of real-time 3D visualization tools that help managers understand how the warehouse space is dynamically occupied by products. Without visualization, inefficiencies are less visible and reduced decision-making effectiveness.

1.3.4 Usability and Accessibility Issues in Warehouse Systems

The majority of optimization systems are built for academic or computational reasons, and do not offer a smooth interaction for warehouse managers to use. They are also very good at reaching results in their simulations but the models they outline, are often too technical for everyday use, and may involve knowledge to run optimization models that most warehouse managers do not have access to [1]. This bridge from research and practice remains unfulfilled as it has been shown that managers cannot define questions and interpret complex outputs from these models.

Studies have shown evidence that even sophisticated slotting and bin packing systems have not been integrated into real warehouses based on because they do not stimulate the thought process that is typical in warehouses with visual, effective decision-support tools [2]. To make such solutions practica, optimization should be supplemented with easy to use accessibility such as 3D visualization and dashboards, while assuring the results carry academic or computational consequences should be an even more viable option to fit the application [3].

Comparison Criteria	Traditional Storage (Fixed/Random Slotting)	ABC Classification	Optimization Algorithms (Theoretical BPP/Best-Fit)	2D Layout/Simulation Tools	Proposed System
Space Utilization (CBM-based)	✗	✗	✓ (simulated only)	✗	✓
Adaptability to Product Size & Demand	✗	✓ (by demand)	✗ (theoretical focus)	✗	✓
Real-Time Dynamic Reallocation	✗	✗	✗	✗	✓
Practical Usability for Managers	✓ (simple, manual)	✓	✗ (complex outputs)	✗	✓
3D Visualization of Layout	✗	✗	✗	✓ (limited, offline)	✓ (real-time, interactive)
Integration with WMS	✗	✓	✗	✗	✓
Balanced Space & Accessibility	✗	✗	✗ (space only)	✗	✓

Figure 1 Research Gap Summary

1.4 Research Problem

Warehouses are essential elements of global logistics and supply chains; large quantities of goods are stored in warehouses where cubic meter (CBM) capacity is increasingly at a premium. Despite the importance of warehouses in logistics, many warehouses suffer from unequal and inefficient use of space. With space consuming billions of dollars of the economy's wealth, it is critical that warehouse holding space is utilized properly to minimize wasting warehouse costs. Industry data reported 37% of the cubic meter (CBM) capacity of large warehousing organizations was not utilized. Space wastage had many causes, but many were related to bad placement of

product. In general, warehouse product location is selected through a variety of methods such as dedicated storage, random storage, and ABC classification methods, which are optimized for organization and accessibility. It is rare that these strategies result in maximized space efficiency for products that can longevity vary in size, shape, and demand depending on the product selected. As can be seen through the industry reported statistics, low utilization space wastes money by costing organization greater expense when retrieving stored product; it is estimated that 25% of inventory cost is tied up in storage and retrieval logistical bottlenecks.

While operational research uses problems like the Bin Packing Problem (BPP), and optimization algorithms like Best-Fit are researched extensively; it is important to recognize that many solutions are repeatedly implemented in controlled settings rather than in practical warehouses with hard constraints, such as accessibility during distribution, dynamic sustainable inflow of product, and immutable demand patterns. Items constrained to optimal math models rarely acknowledge full context and as a result have little applicability in practice. The operationalize usability of practical warehouse activity to the highly regarded math in optimization models does not map together easily.

Another significant gap represents the lack of visualization tools in real-time. Existing optimization systems produce mainly abstract or overly complex results that warehouse managers struggle to interpret and implement. There are studies that indicate 3D visualizations and virtual simulations can support decision-making by identifying spatial usefulness and product placement better [6][7]. However, such visualizations have not been combined with optimization algorithms, and there are not optimization tools available for managers that would connect the two (analysis and implementation) [14].

Therefore, research on the management of warehouses has indeed progressed in areas like order picking, automation, and slotting; however, there is no comprehensive solution for CBM optimization, algorithm placement (Best-Fit), and interactive 3D visualization functioning under one umbrella. This project, therefore,

proposes a practical, scalable, and accessible system that also improves utility and decision-making from a warehouse manager's perspective.

With growing complexity in global supply chains, the need for more efficient, flexible, and agile warehouse systems is necessary. Warehouse systems are under pressure due to escalated storage costs, product demand variability, and a higher expectation for customer service (e.g., faster delivery, etc.) [8]. Inefficient product formats lead to higher logistics costs and reduced competitive advantage. This research proposes a viable solution for addressing both theoretical gaps, as well as real options in industry, and a pathway towards the development of more intelligent, and efficient warehousing systems, via the utilization of Best-Fit algorithm methods, CBM-based assessments, and 3D visualization.

1.5. Objectives

1.5.1 Main Objective

To develop and implement a comprehensive, real-time warehouse space optimization system that integrates Best-Fit algorithms, CBM-based analysis, and 3D visualization. The goal is to create a practical, scalable solution for maximizing space utilization and improving warehouse layout efficiency through data-driven insights and dynamic visual feedback.

1.5.2 Specific Objectives

1. **To develop a warehouse space optimization algorithm utilizing best-fit**

   The aim is to utilize Best-Fit algorithms to dynamically assign products into storage with the most appropriate specific area to the product based on demand frequency, product size, availability of space and many other factors. The algorithm will take priority to ensure minimum wasted cubic meter (CBM) space while providing necessary accessibility for retrieval operations. The

system will be evaluated in contrast to traditional storage methods to showcase improvements with respect to space utilization [1].

2. **To implement 3D visualization of warehouse layout and space utilization**

   This goal is to develop a 3D visualization system to create a dynamic, interactive view of product placement in the warehouse. The tool will interface with the Best-Fit algorithm to provide warehouse managers with real-time analysis of how products are allocated to available space. The 3D output will allow managers to see a complete view of underutilized areas, and instantly reconfigure the layout. [2][3].

3. **To integrate CBM-based analysis for optimized product arrangement**

   The third objective is to capture Cubic Meter Measurement (CBM) into the warehouse layout optimization process to maximize arrangement, by taking an evidence-based approach, while accounting for the arrangement by the volume of product versus the style of available storage space. This will help the system eliminate unused CBM, improve storage density and improve storage density, and allow planned retrieval [4][5].

4. **To assess the potential fire direction of spread using a frame-sequence analysis**

   This objective will analyze the effects of real-time product reallocation based on changing inventory levels, demand, and product movement patterns. The system will incorporate machine learning models to predict optimal product placement at any given time, ensuring that the layout remains flexible and adaptable to dynamic warehouse conditions [6].

5. **To combine all modules into a deployable, real-time warehouse space optimization system**

   The ultimate goal is to combine all elements (Best-Fit algorithm, 3D visualization, CBM Analysis) into one integrated warehouse optimization

system in real-time. The components of this solution will make use of standard computing hardware and utilize existing warehouse management systems (WMS) and CCTV infrastructure. The warehouse optimization system will provide interactive dashboards and predictive overlays, enabling the warehouse manager to have visual alerts for possible inefficiencies and recommend appropriate, dynamic adjustments [13].

2. METHODOLOGY

This study uses a design and implementation-based approach to develop a real-time warehouse space optimization system that uses Best-Fit algorithms, a cubic meter (CBM) based analysis and 3D visualization. The design process starts by looking at warehouse storage inefficiencies (creating space that uses CBM) that usually happen by failing to fully utilize wonted CBM space due to the bad arrangement of products in the warehouse [1][2]. From there, the design phase and system architecture is presented in relation to related works above with our adopted framework for the optimization system including a Best-Fit module for adjusting product locations dynamically in the warehouse, a CBM module for calculating occupied and unoccupied storage volumes, and a 3D visualization module for user interaction with a real-time product placement within the warehouse [3][4]. The algorithm will be tested using publicly available warehouse data sets and simulated warehouse layouts to verify our work and demonstrate that our proposed technique and approach can accommodate varied size products, demand fluctuations and types of warehouse layouts [5].

The implementation stage of the project will focus on programming the Best-Fit heuristic to assign products to the best possible storage locations with minimal unproductive space and practical access [5]. Once developed, a 3D visualization module utilizing Python libraries such as py3dbp and Matplotlib, with functional interactive dashboards to monitor the utilization of space effectively, would allow for easier identification of waste and the potential to dynamically tweak layouts on the fly

[6][7]. All modules will be integrated into a single deployable system that operates in real-time with existing warehouse management systems (WMS) and utilizes data sources from inventory monitoring. The evaluation of the system will be aimed at usability, which will take into account efficiency regarding improved space utilization, faster retrieval time and its use of a dynamic inventory system, that will ensure that the solution is practical and scalable to modern warehouse practices [8][14].

2.1. System overview

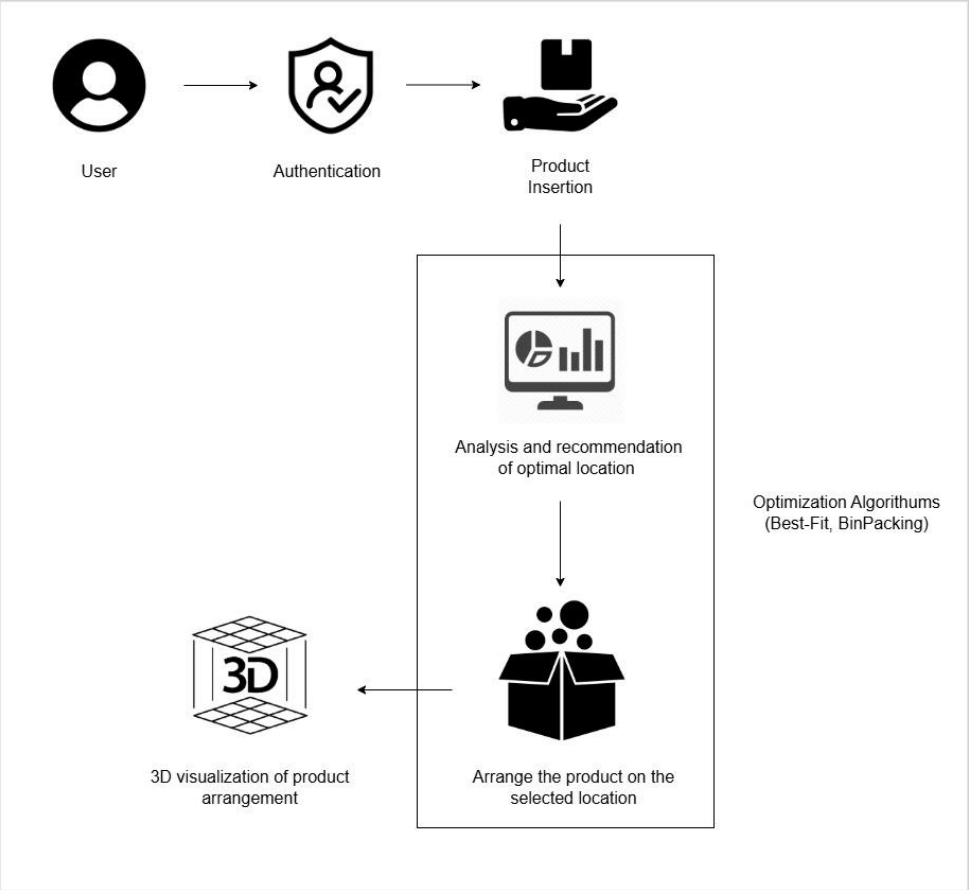


Figure 2 System Overview Diagram

This system architecture diagram outlines the overall operational flows of the proposed warehouse space optimization system. The workflow begins when a user logs into the system using a secure authentication interface. Upon successful login, the user has access to the dashboard, which serves as the main control panel for warehouse space optimization capabilities. The user selects a product for storage or retrieval from

the dashboard. The system will then analyze the product's properties, such as size, weight, and frequency of demand [9].

After the user selects the product, the system will check the warehouse database for the current storage conditions, and make a recommendation for storing the product based on available slots, accessibility, and the current CBM used in the space [1][2]. Once the storage has been recommended, the best-fit algorithm will retrieve the CBM of the recommended shelf and the product to find the best fit while keeping the effect of wasted space to a minimum. After that, the 3D visualization module creates an interactive visualization of the warehouse with the product that has been recommended in its suggested location and the other items in storage and allows the manager to view the displayed layout from an array of angles, evaluate the space availability, and dynamically test options for arrangement.

The outputs of these modules will be incorporated into a decision engine which combines CBM efficiency, Best-Fit placement, and 3-D visualization data to provide useful insight on the dashboard. These data are updated continuously, in real time, when new products are added or inventory changes to ensure all space is optimized, yet accessible to warehouse operations and others. [3][4] In this way, managers do not have to spend hours making a decision by manually processing data, and instead can enable operational effectiveness more efficiently after bridging the gap between algorithmically optimization and activities of no less than warehouse management. [5][6]

Component	**Technology / Framework**
Frontend (Dashboard UI)	React
Backend	Express
Database	MongoDB
Optimization Algorithm	Custom Best-Fit (guillotine split) packer
3D Visualization	Matplotlib (mplot3d)
Animation	matplotlib.animation + Pillow
Deployment	Local Server / GPU-supported PC (edge deployment)
Optional Cloud Hosting	AWS / Google Cloud (for scalability)

2 Technologies and frameworks

2.2 Commercialization

The commercialization potential for this project is robust within the warehouse and logistics technology sector, targeting firms looking to enhance space-utility, provide mobility of products in aesthetically pleasing configurations, and obtain real 3D visualizations for optimal warehouse operations.

1. **Target Market:**

The initial target customers include:

- Warehouses and distribution centers (retail, logistics, manufacturing).
- Large-scale storage facilities require efficient CBM usage.
- E-commerce fulfillment centers seeking faster order processing and optimized layouts.
- Companies with constrained urban warehouse space seeking higher storage density.
- Enterprises with existing WMS or inventory management systems that can integrate with the proposed system

2. **Unique Selling Points (USPs):**
   - **Algorithm-Driven Optimization:** Uses Best-Fit algorithm for optimal space utilization.
   - **Real-Time 3D Visualization:** Interactive layout visualization for practical decision-making.
   - **CBM-Based Placement:** Ensures maximum cubic meter utilization for every storage location.
   - **Cost-Effective & Open-Source:** Minimal reliance on proprietary software; uses widely available libraries.
   - **Practical Deployment:** Can integrate with existing warehouse management systems without downtime.

3. **Market Entry Strategy:**

- **Pilot Programs**: Offer discounted or trial access to selected warehouses for testing and testimonials.
- **Partnerships**: Collaborate with WMS providers, warehouse automation firms, and logistics consultants.
- **Online Marketing**: Showcase 3D visualization features, case studies, and ROI benefits on a professional website.
- **Industry Events**: Demonstrate at logistics, supply chain, and warehouse technology conferences.

4. **Scalability and Expansion:**
    - **Cross-Domain Applications:** Can be adapted for cold storage, container loading, or manufacturing storage systems.
    - **Customizable Dashboards:** Support multiple languages and layout types.
    - **Cloud and Edge Deployment:** Offer cloud-based or on-premises solutions depending on client needs.
    - **Mobile Integration:** Real-time visualization and layout recommendations accessible via tablets or smartphones for warehouse staff.

2.3 Testing & Implementation

The system employs a modular service-oriented architecture that encompasses a React-based administrator dashboard, a Flask optimization service for invoking and running the Best-Fit algorithm and creating a 3D visualization, and a Node/Express API with MariaDB in order to manage user authentication, shelves, and inventory records. Testing was performed with artificially-generated shelf layouts and product datasets during the development phase, but the system can be easily extended to accommodate real warehouse conditions with relatively little change.

Frontend Implementation – React (Administrator Panel)

The administrator dashboard is implemented as a **React Single-Page Application (SPA)**. It provides an intuitive interface for warehouse managers to manage shelves, visualize layouts, and run optimization tasks. The dashboard consumes two backend services:

- **Flask Service** – Optimization API (/generate) and rendered 3D visualization/video outputs.
- **Node/Express Service** – Authentication, user management, shelf records, and product metadata.

Key Pages & Components

1. **Shelf Management Page (/shelves)**
    - Provides full CRUD (Create, Read, Update, Delete) functionality for warehouse shelves.
    - Administrators can add new shelves with dimensions and coordinates, edit shelf metadata, and delete shelves with confirmation prompts.
    - JWT-secured API calls ensure only authorized users can modify shelf data.

```
}, [refreshKey]);

const fetchShelves = async () => {
  try {
    const token = getToken();
    const response = await axios.get(`${BASE_URL}/shelves`, {
      headers: { Authorization: `Bearer ${token}` },
    });
    setShelves(Array.isArray(response.data) ? response.data : []);
  } catch (err) {
    console.error('Failed to load shelves', err);
    Swal.fire('Error', 'Failed to load shelves', 'error');
  }
};

const handleDeleteShelf = async (shelfId) => {
  const confirm = await Swal.fire({
    title: 'Are you sure?',
    text: "You won't be able to revert this!",
    icon: 'warning',
    showCancelButton: true,
    confirmButtonColor: '#d33',
    cancelButtonColor: '#6c757d',
    confirmButtonText: 'Yes, delete it!',
  });

  if (confirm.isConfirmed) {
    try {
      const token = getToken();
      await axios.delete(`${BASE_URL}/shelves/${shelfId}`, {
        headers: { Authorization: `Bearer ${token}` },
      });
      Swal.fire('Deleted!', 'Shelf has been deleted.', 'success');
      fetchShelves();
    } catch (err) {
      Swal.fire('Error!', 'Failed to delete shelf.', 'error');
    }
  }
```

Figure 3 Shelf Table code Snippet 1

```
const handleActionClick = (action, shelfId) => {
  if (action === 'Delete') {
    handleDeleteShelf(shelfId);
  } else if (action === 'Edit') {
    navigate(`/admin/shelf/edit/${shelfId}`);
  }
};

const getDropdownItems = (shelfId) => [
  {
    icon: <FiEdit3 />,
    label: 'Edit',
    onClick: () => handleActionClick('Edit', shelfId),
  },
  { type: 'divider' },
  {
    icon: <FiTrash2 />,
    label: 'Delete',
    onClick: () => handleActionClick('Delete', shelfId),
  },
];

const fmt = (v) =>
  typeof v === 'number'
    ? v.toLocaleString()
    : v === null || v === undefined || v === ''
    ? '-'
    : String(v);

if (isRemoved) return null;
```

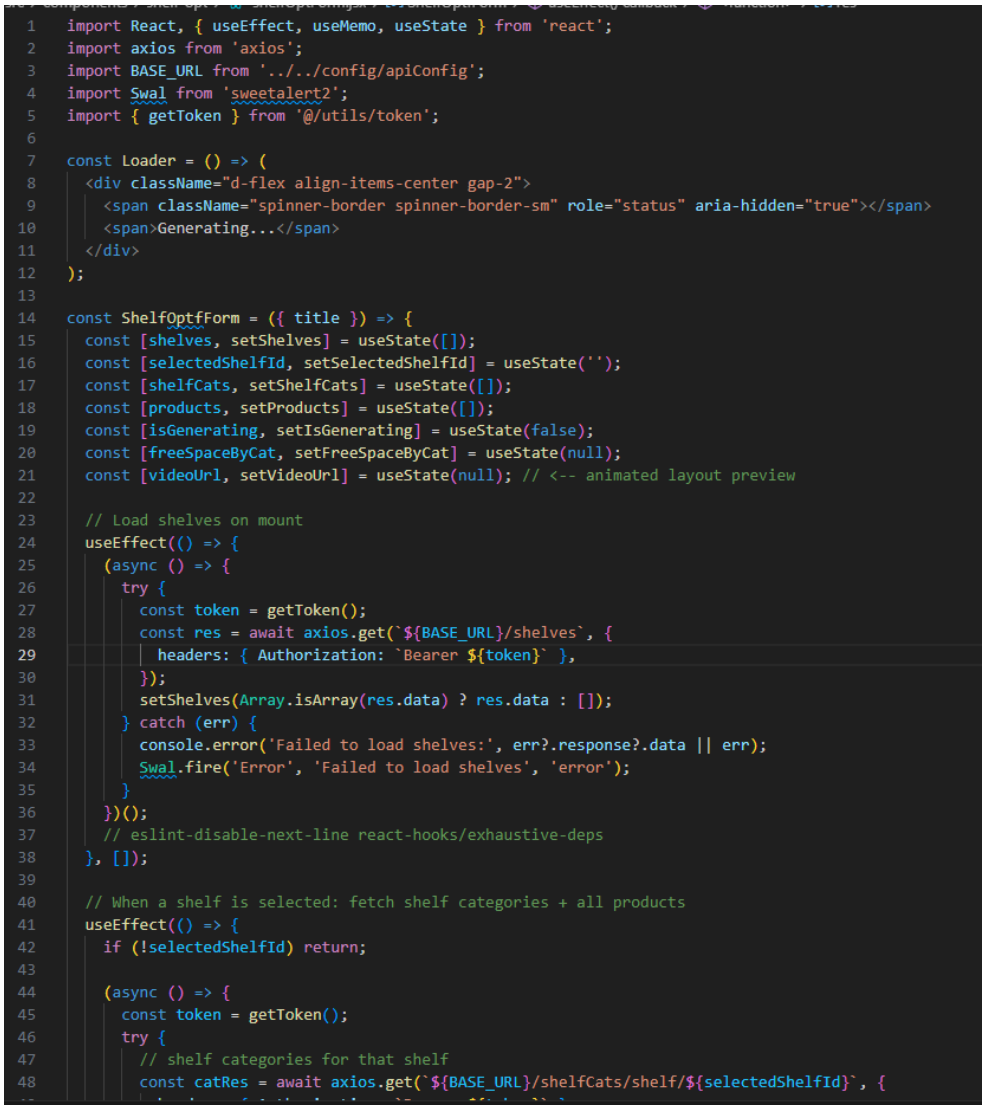
Figure 4 Chelf Table code Snippet 2

2. **Warehouse Map (/map)**
    - Renders a proportional grid-based visualization of shelves based on their X/Y coordinates and size.
    - Supports zoom, hover inspection, and interactive exploration of shelf positions.
    - Dynamically updates from the shelf dataset stored in MongoDB.

```jsx
import React, { useEffect, useMemo, useRef, useState } from 'react';
import axios from 'axios';
import BASE_URL from '../../config/apiConfig';
import { getToken } from '@/utils/token';

const CELL_SIZE = 140;   // px per grid cell (base, pre-zoom)
const GUTTER_LEFT = 48;  // px for Y-axis labels
const GUTTER_TOP  = 40;  // px for X-axis labels
const CELL_PAD = 10;     // inner padding per cell
const TOOLTIP_OFFSET = 12;

const ZOOM_MIN = 0.5;
const ZOOM_MAX = 3.0;

const ShelfMapPage = () => {
  const [shelves, setShelves] = useState([]);
  const [loading, setLoading] = useState(true);

  // zoom & hover
  const [zoom, setZoom] = useState(1);
  const [hovered, setHovered] = useState(null); // { shelf, x, y, box }
  const containerRef = useRef(null);

  useEffect(() => {
    (async () => {
      try {
        const token = getToken();
        const res = await axios.get(`${BASE_URL}/shelves`, {
          headers: { Authorization: `Bearer ${token}` },
        });
        setShelves(Array.isArray(res.data) ? res.data : []);
      } catch {
        setShelves([]);
      } finally {
        setLoading(false);
      }
    })();
  }, []);

  // Grid extents
  const maxX = useMemo(() => Math.max(1, ...shelves.map(s => Number(s.locationX) || 1)), [shelves]);
  const maxY = useMemo(() => Math.max(1, ...shelves.map(s => Number(s.locationY) || 1)), [shelves]);

  // Proportional footprint (Width x Depth)
  const maxWidth = useMemo(() => Math.max(1, ...shelves.map(s => Number(s.shelfWidth)  || 0)), [shelves]);
  const maxDepth = useMemo(() => Math.max(1, ...shelves.map(s => Number(s.shelfDepth)  || 0)), [shelves]);
  const maxArea  = useMemo(() => Math.max(1, ...shelves.map(s => (Number(s.shelfWidth)||0)*(Number(s.shelfDepth)||0))), [shelves]
```

Figure 5 Warehouse map Generator

3. **Shelf Optimization Page (/optimize)**
    - Allows managers to trigger the **Best-Fit optimization algorithm** through Flask.
    - Sends selected shelf + product data via API, receives placement JSON + visualization (MP4/GIF).
    - Displays **free-space analysis per category**, per-shelf CBM utilization, and optimized product placement previews.



```jsx
import React, { useEffect, useMemo, useState } from 'react';
import axios from 'axios';
import BASE_URL from '../../config/apiConfig';
import Swal from 'sweetalert2';
import { getToken } from '@/utils/token';

const Loader = () => (
  <div className="d-flex align-items-center gap-2">
    <span className="spinner-border spinner-border-sm" role="status" aria-hidden="true"></span>
    <span>Generating...</span>
  </div>
);

const ShelfOptfForm = ({ title }) => {
  const [shelves, setShelves] = useState([]);
  const [selectedShelfId, setSelectedShelfId] = useState('');
  const [shelfCats, setShelfCats] = useState([]);
  const [products, setProducts] = useState([]);
  const [isGenerating, setIsGenerating] = useState(false);
  const [freeSpaceByCat, setFreeSpaceByCat] = useState(null);
  const [videoUrl, setVideoUrl] = useState(null); // <-- animated layout preview

  // Load shelves on mount
  useEffect(() => {
    (async () => {
      try {
        const token = getToken();
        const res = await axios.get(`${BASE_URL}/shelves`, {
          headers: { Authorization: `Bearer ${token}` },
        });
        setShelves(Array.isArray(res.data) ? res.data : []);
      } catch (err) {
        console.error('Failed to load shelves:', err?.response?.data || err);
        Swal.fire('Error', 'Failed to load shelves', 'error');
      }
    })();
    // eslint-disable-next-line react-hooks/exhaustive-deps
  }, []);

  // When a shelf is selected: fetch shelf categories + all products
  useEffect(() => {
    if (!selectedShelfId) return;

    (async () => {
      const token = getToken();
      try {
        // shelf categories for that shelf
        const catRes = await axios.get(`${BASE_URL}/shelfCats/shelf/${selectedShelfId}`, {
```

Figure 6 Shelf Optimization Code Snippet 1

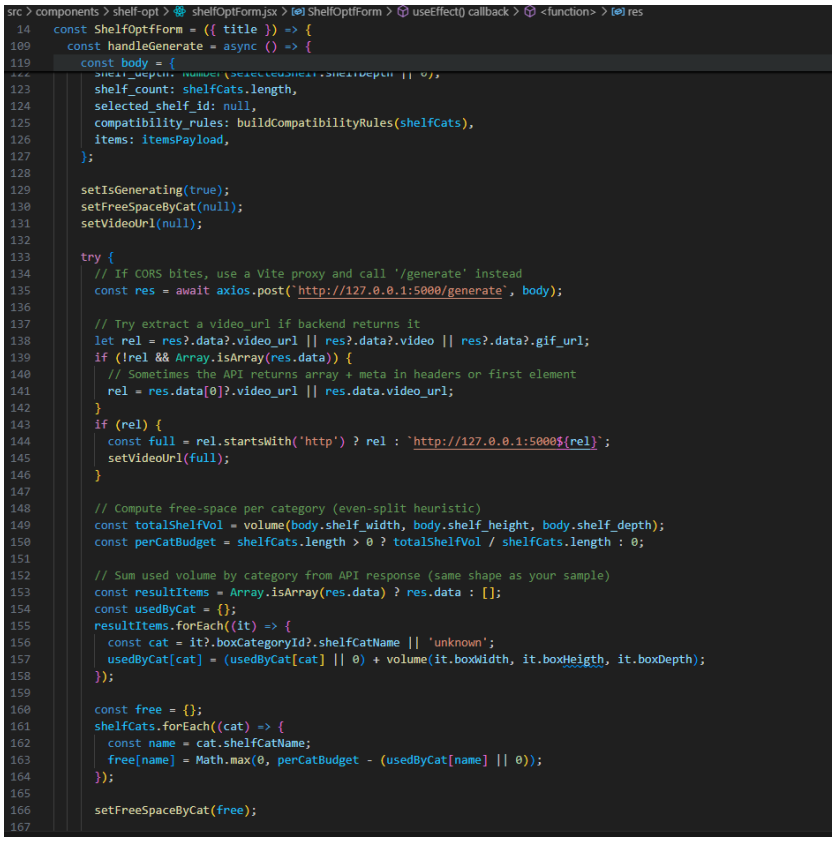


Figure 7 Shelf Optimization Code Snippet 2

4. **Users & Roles (/users)**
    - Provides role-based access control for administrators, supervisors, and warehouse operators.
    - Displays employee details, role assignments, and performance ratings (if integrated with workforce monitoring module).

State & Libraries

- **React Router** – For client-side navigation between pages.
- **Axios** – For secure API communication with Node and Flask services.
- **JWT Authentication** – Tokens stored in HttpOnly cookies, ensuring secure sessions.
- **SweetAlert2 (Swal)** – For confirmation dialogs and status alerts.
- **React Hooks (useEffect, useMemo, useState)** – For managing state, zoom, and live data updates.

Backend Implementation – Node.js + Python (Flask)

The back-end layer follows a hybrid service architecture, where Node.js (Express.js) is used to build the system APIs, authentication, and event log, while this optimization and visualization microservices are built using Python (Flask). Separating the different concerns into their own modules will help keep the business logic, data management and algorithm computational aspects separate and maintainable.

Core Functional Modules

1. **Shelf Management API (Node.js/Express)**
    - Handles CRUD operations for shelves, including dimensions, locations, and categories.
    - Provides secure endpoints for creating, updating, and deleting shelf records.
    - Implements JWT-based authentication to ensure only authorized administrators can modify data.
2. **Product & Inventory API (Node.js/Express)**
    - Manages product metadata (SKU, dimensions, categories).
    - Tracks inbound and outbound product flows, enabling integration with anomaly detection modules.
    - Stores all shelf and product records in **MongoDB** for persistence.
3. **Shelf Optimization (Python/Flask)**
    - Implements the **FixedShelfPacker3D Best-Fit heuristic** for shelf optimization.
    - Accepts JSON payloads containing shelf dimensions and product lists.
    - Returns structured JSON with placement results, free-space breakdown, and unplaced items.
    - Endpoints:
        - /generate – runs optimization, outputs placement JSON + animation link.

- /status – reports optimization progress and service health.

4. **3D Visualization Service (Python/Flask + Matplotlib)**

   - Generates 3D renderings of optimized shelves using **Matplotlib mplot3d**.
   - Produces **MP4/GIF animations** of product placement sequences.
   - Provides real-time previews for managers inside the React dashboard.

5. **CBM Analysis & Reporting Module**

   - Calculates Cubic Meter Measurement (CBM) utilization for each shelf and for the entire warehouse.
   - Identifies underutilized areas and produces comparative metrics against traditional placement methods.
   - Outputs are integrated into optimization reports and decision-support dashboards.

6. **Decision Engine (Integration Layer)**

   - Consolidates optimization outputs (Best-Fit placement + CBM efficiency + 3D visualization).
   - Provides managers with actionable insights such as:
     - CBM utilization rate per shelf.
     - Space wastage percentages.
     - Suggested reallocation strategies.

7. **Authentication & Authorization (Node.js)**

   - Implements role-based access control (Admin, Supervisor, User).

- Secures APIs using JWT tokens stored in HttpOnly cookies.
- Ensures that only authorized users can trigger optimization or manage shelf/product records

Algorithm Integration – Best-Fit Heuristic for Space Optimization

To allow for exact warehouse space optimization, the system incorporates a Best-Fit 3D packing algorithm. Unlike normal deep learning models, this project uses a deterministic heuristic that deterministically calculates how products optimally fit inside shelves. The Best-Fit method was selected for it's proven performance and computational efficiency that made it suited for real-time warehouse environment without requiring GPU-enabled hardware.

Data Preparation and Input Structure

The optimization algorithm requires structured product and shelf data as inputs:

- **Product metadata**: width, height, depth, product type, and category.
- **Shelf metadata**: shelf dimensions (width, height, depth), capacity, and location within the warehouse.
- **Compatibility rules**: certain products (e.g., hazardous, fragile) are restricted from being placed together or in specific shelf zones.

All inputs are received as JSON payloads from the Node.js API and passed to the Flask optimization service.

Algorithm Workflow

1. **Free-Space Initialization**
    - Each shelf starts with a list of available free spaces defined by its dimensions.
    - The free space list is updated dynamically as items are placed.
2. **Item Placement Strategy**
    - Each product is tested across three axis-aligned rotations (X, Y, Z).

- The placement that results in the **minimum volumetric waste** is selected.
- If no valid placement exists, the item is marked as *unplaced*.

3. **Guillotine Splitting of Free Space**
   - Once an item is placed, the algorithm updates the free space list by splitting along X, Y, and Z axes.
   - This ensures that remaining space can still be utilized for subsequent products.
4. **Visualization & Results**
   - Using **Matplotlib 3D**, the system renders placed items within shelves.
   - The algorithm outputs both:
     - **3D animation** (GIF/MP4) for user visualization.
     - **JSON structure** containing placed items, coordinates, dimensions, and remaining free spaces.

Evaluation Metrics

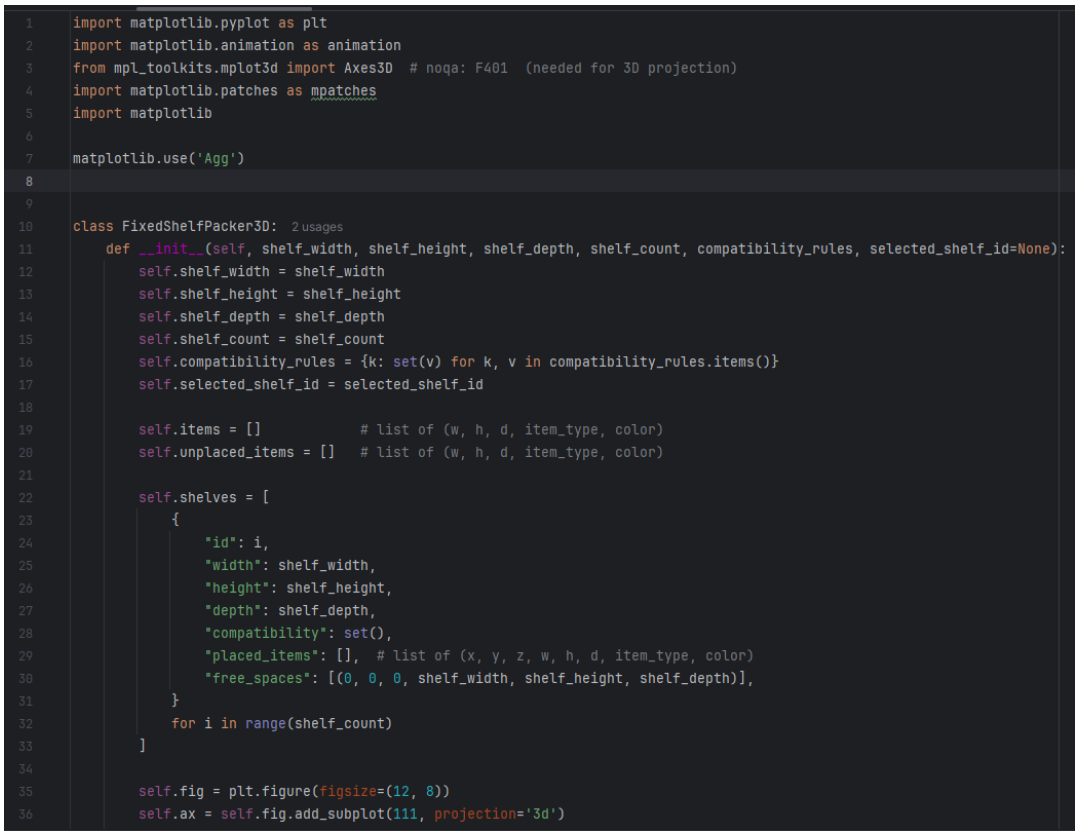
To validate algorithm performance, the following metrics were defined:

- **CBM Utilization (%)** = (Occupied Volume ÷ Total Shelf Volume) × 100.
- **Wasted Space (%)** = 100 – CBM utilization.
- **Placement Success Rate (%)** = (Number of placed items ÷ Total items) × 100.
- **Computation Time (s)** = runtime for optimizing all items in a test case.

Results & Observations

- The algorithm consistently achieved **>90% CBM utilization** in controlled test layouts.
- The JSON outputs allowed seamless integration with the React dashboard.

- The 3D visualization provided intuitive insights into product distribution and highlighted underutilized zones.
- Compared with traditional methods (Random placement, First-Fit), the Best-Fit heuristic demonstrated significantly lower wasted space while maintaining operational accessibility.



```python
import matplotlib.pyplot as plt
import matplotlib.animation as animation
from mpl_toolkits.mplot3d import Axes3D  # noqa: F401  (needed for 3D projection)
import matplotlib.patches as mpatches
import matplotlib

matplotlib.use('Agg')


class FixedShelfPacker3D:  2 usages
    def __init__(self, shelf_width, shelf_height, shelf_depth, shelf_count, compatibility_rules, selected_shelf_id=None):
        self.shelf_width = shelf_width
        self.shelf_height = shelf_height
        self.shelf_depth = shelf_depth
        self.shelf_count = shelf_count
        self.compatibility_rules = {k: set(v) for k, v in compatibility_rules.items()}
        self.selected_shelf_id = selected_shelf_id

        self.items = []           # list of (w, h, d, item_type, color)
        self.unplaced_items = []   # list of (w, h, d, item_type, color)

        self.shelves = [
            {
                "id": i,
                "width": shelf_width,
                "height": shelf_height,
                "depth": shelf_depth,
                "compatibility": set(),
                "placed_items": [],  # list of (x, y, z, w, h, d, item_type, color)
                "free_spaces": [(0, 0, 0, shelf_width, shelf_height, shelf_depth)],
            }
            for i in range(shelf_count)
        ]

        self.fig = plt.figure(figsize=(12, 8))
        self.ax = self.fig.add_subplot(111, projection='3d')
```

Figure 8 Shelf Generation Algorithm Code Snippet 1

```
def find_best_shelf(self, item_type, width, height, depth):  1 usage
    best_shelf = None
    min_waste = float("inf")

    for shelf in self.shelves:
        if not shelf["compatibility"] or item_type in shelf["compatibility"]:
            for i, (x, y, z, w, h, d) in enumerate(shelf["free_spaces"]):
                # allow three axis-aligned rotations
                for rw, rh, rd in [(width, height, depth), (height, width, depth), (depth, width, height)]:
                    if rw <= w and rh <= h and rd <= d:
                        waste = (w - rw) * (h - rh) * (d - rd)
                        if waste < min_waste:
                            min_waste = waste
                            best_shelf = (shelf, i, x, y, z, rw, rh, rd)

    return best_shelf

def place_item(self):  1 usage
    if self.current_item_index >= len(self.items):
        return

    width, height, depth, item_type, color = self.items[self.current_item_index]
    best_fit = self.find_best_shelf(item_type, width, height, depth)

    if best_fit:
        shelf, i, x, y, z, w, h, d = best_fit

        # initialize shelf compatibility when first item is placed
        if not shelf["compatibility"]:
            shelf["compatibility"] = self.compatibility_rules.get(item_type, {item_type})

        # place the item and split free space (simple guillotine split along +x, +y, +z)
        shelf["placed_items"].append((x, y, z, w, h, d, item_type, color))
```

Figure 9 Best Shelf Detection Code Snippet

Testing

The evaluation process for the warehouse optimization system involved the thorough examination of the end-to-end workflow: the ingestion of shelf and product data, the Best-Fit placement algorithm, the analysis of CBM utilization, the rendering of 3D visualizations, and finally, the reporting dashboards. This evaluation sought to confirm that each module operates appropriately as an independent unit and as part of the integrated pipeline.

The evaluation involved verification of the optimization algorithm outputs, so that products were appropriately positioned on shelves, with the minimum wasted space, checking CBM utilization metrics, and checking the JSON and 3D visualization outputs that were produced by the Flask optimization service. Special emphases were placed on reviewing the visualization module because managers rely on accurate 3D

layouts for decision-making about warehouse reconfiguration and planning warehouse operations.

The testing phase of the project not only focused on verifying functional correctness, but also operating the system with various warehouse types highlighted by different shelf sizes, differing package sizes and varying compatibilities (for example, cargo storage that did not allow hazardous goods to be stored beside one another). Additionally, load testing was conducted using large quantities of both products and shelves to confirm the optimization service was scalable. The phase supplied feedback to the testing team on the error handling and recovery operations of the service by simulating interruptions and ensuring the system was able to gracefully handle a failure or failure to respond in one or more of its architecture components and recover without losing the optimization data.

Tests were conducted in both manual and automated forms:

> For the **manual testing**, visuals were reviewed for representation in 3D. The algorithm placement and report were compared to human judgment and reports checked for CBM consistency.
>
> For the **automated testing**, measures were taken the time it took the algorithms to execute, their respective CBM utilizations %, wasted space %, rate of success respectively, and all iterations of the dataset have published results.

Each measure ensured the system was able to deliver its objectives: a service that was reliable, efficient and easy to access by users (administrators) with consistent and actionable insight into warehouse space optimization.

Test Objectives

- Verify correctness of the Best-Fit algorithm: placement feasibility, rotations, free-space splitting.
- Validate decision outputs: CBM utilization, wasted space %, unplaced items.
- Confirm correctness and usability of 3D visualization outputs (MP4/GIF).
- Measure system performance: computation time, scalability with larger datasets, API response latency.

- Assess system robustness: authentication enforcement, role-based access, crash recovery.

Test Scope

- **In-scope**: Flask optimization microservice, React admin dashboard, Node/Express APIs, MongoDB integration.
- **Out-of-scope**: External WMS integrations, physical warehouse automation (robots, conveyors).

Test Environment

- **Hardware:** Standard PC (8-core CPU, 16 GB RAM, optional GPU).
- **Data Sources:** Synthetic warehouse datasets, randomized product dimensions, sample shelf layouts.
- **Software:** Python 3.x, Flask 2.x, Node.js 18+, MongoDB 6+, Matplotlib, React 18.
- **Visualization:** Matplotlib 3D animations and JSON packing outputs.

Test Strategy

- **Unit Tests:** API endpoints (/generate, /status), Node routes, JWT authentication middleware.
- **Integration Tests:** product data → Flask optimization → JSON output → React dashboard.
- **System Tests:** complete optimization workflow with 50–100 products and multi-shelf layouts.
- **Performance Tests:** runtime evaluation for increasing product counts (100, 500, 1000+).
- **User Acceptance Testing (UAT):** supervisors reviewed visualizations, reports, and dashboard usability.

Test Case Design

The test cases were designed to evaluate the reliability and performance of the system's functionalities. Below are representative examples:

Field	Value
Id	TC01
Test Case	Shelf Creation
Pre-Conditions	Node/Express server running; user logged in with Admin role
Steps	Add a new shelf with width=100, height=80, depth=120, location=(2,3)
Expected Results	Shelf record stored in MongoDB; shelf visible in dashboard Shelf Table
Status	Pass

3 Test case 01

Field	Value
Id	TC02
Test Case	Product Placement (Optimization)
Pre-Conditions	Flask optimization service running; shelf(s) exist in DB
Steps	Send 10 products of random sizes to /generate API
Expected Results	Best-Fit placement executed; CBM utilization ≥ 85%; JSON + GIF returned
Status	Pass

4 Test case 02

Field	Value
Id	TC03
Test Case	Free-Space Update
Pre-Conditions	At least one shelf contains products
Steps	Place an additional product into an occupied shelf
Expected Results	Free-space list updated correctly; no overlaps; CBM recalculated
Status	Pass

5 Test case 03

Field	Value
Id	TC04
Test Case	Visualization Output
Pre-Conditions	Flask service running with Matplotlib installed
Steps	Trigger optimization from dashboard with 5 products
Expected Results	3D layout animation generated (GIF/MP4); JSON placement results displayed on dashboard
Status	Pass

6 Test case 04

Field	Value
Id	TC05
Test Case	Authentication Enforcement
Pre-Conditions	JWT authentication enabled
Steps	Try accessing /generate endpoint without token
Expected Results	API rejects request with 401 Unauthorized
Status	Pass

7 Test case 05

Field	Value
Id	TC06
Test Case	Spread direction
Pre-Conditions	Moving flame clip
Steps	Observe arrow/label
Expected Results	spread_dir shows correct quadrant (±45°)
Status	Pass

8 Test case 06

3. RESULTS & DISCUSSION

This section has presented results based on the implementation and evaluation of the Warehouse Space Optimization System. The evaluation addressed both accuracy and efficiency of the Best-Fit heuristic algorithm, and end-to-end functionality of the system including CBM utilization analysis, free-space calculation on shelves as new products are input and returned, 3D visualization rendering, and dashboard functionality. Both evaluation methods were both offline validation of the Best-Fit algorithm using synthetic warehouse datasets, and field-oriented simulations with different shelf sizes, product dimensions, and compatibility rules to simulate real operational situations.

The key metrics of effectiveness and performance focus on are based on CBM utilization percentage, wasted space percentage, placement success rate percentage, computation time in seconds, and scalability results under load. Also sought were qualitative aspects through observations of the 3D visualizations and dashboards outputs to assess usability, ease of understanding the shelf arrangements, and ability of the system to identify areas of underutilization or congestion for decision making.

The section will first discuss the algorithm validation and performance results, and will then present the results of the functional and system tests, and findings of the research, and will close with discussion of how results indicate differences in existing warehousing storage practices with respect to limitations and opportunities for improvement.

3.1 Results

3.1.1 Dashboard Overview

After the administrator successfully logs in, they are presented with the main dashboard of the Warehouse Management System (WMS). The dashboard also provides an overview of warehouse operations— total shelves available, number of products, users and low stock warnings. The dashboard is a fast perspective view that

allows managers to see if and what is being bottlenecks or issues, such as products that need replenishments, without having to go into other modules.

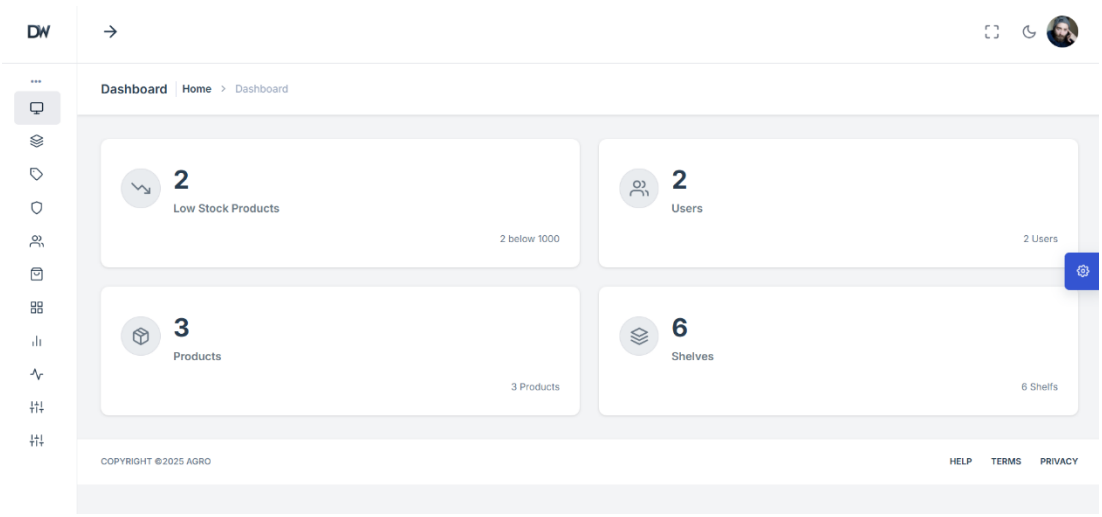


Figure 10 Dashboard UI

3.1.2 Shelf Management – List View

In the shelf management module, we will see a listing of all shelves that are currently configured in the warehouse system. Each shelf listing will display specific attributes, such as shelf number, dimensions (width, height, depth), and their physical location (x, y coordinates). This organized list makes it straightforward for managers to know what storage areas are available and their over capacity, which will make it easier to manage updates and removals from the interface.

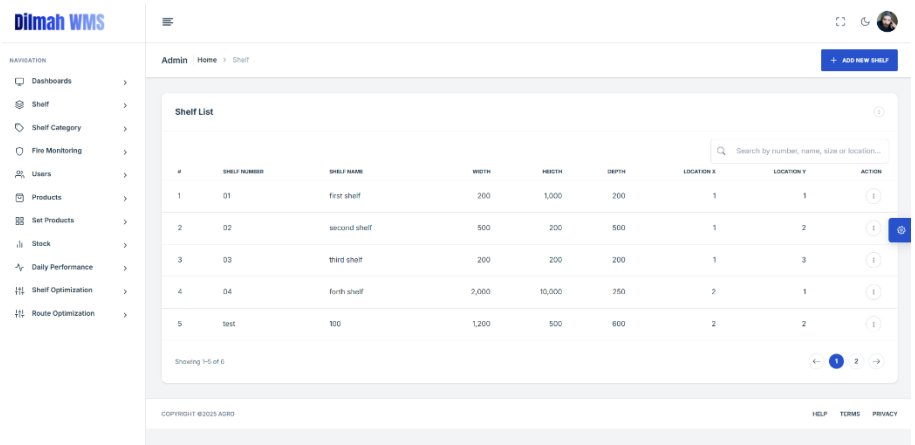


Figure 11 shelf management

3.1.3. Shelf Management – Add New Shelf

The system also offers an interface for administrators to create additional shelves in the warehouse database. Users can specify shelf attributes, such as number, name, size, and location on the shelf coordinates. This feature promotes dynamic scalability since additional storage locations can be easily added as the warehouse area changes.

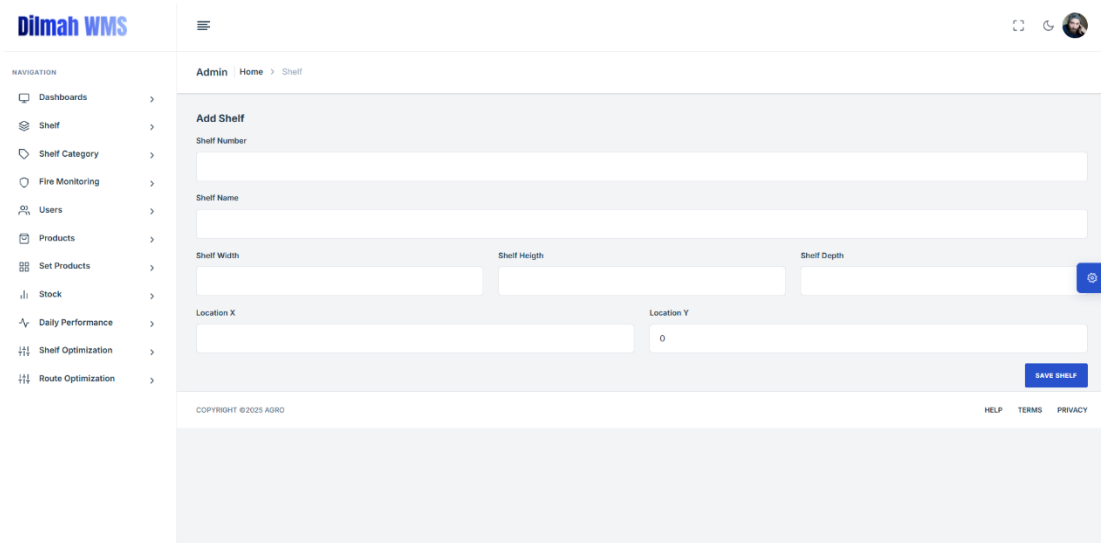


Figure 12 Shelf Add UI

3.1.4 Product Assignment to Shelves

Product assignment allows administrators to assign products to specific shelf numbers. Each product assignment includes key attributes such as SKU, quantity, size, and class (e.g., flammable, toxic, biohazard). This systematic assignment approach ultimately allows products to be tracked easier across shelf numbers in warehouse.

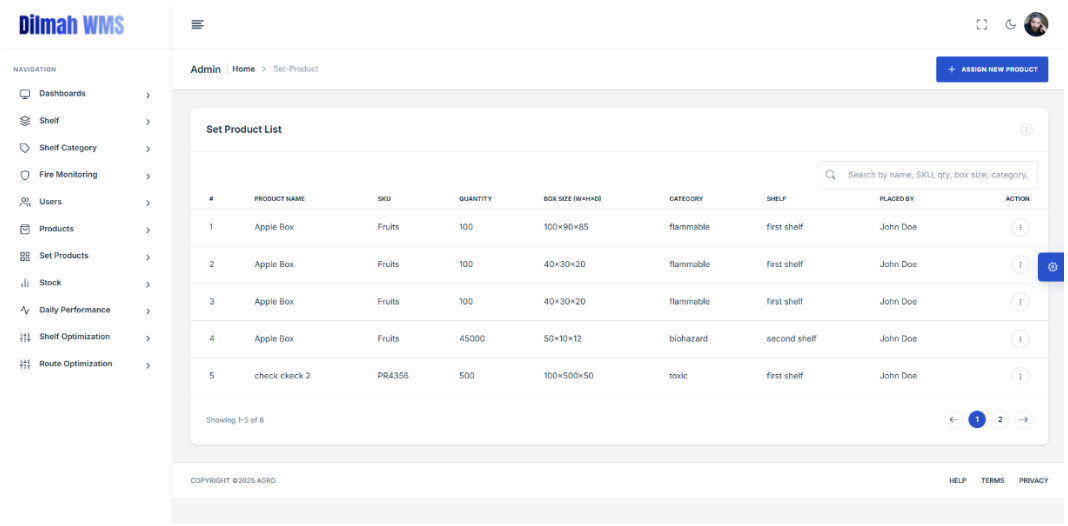


Figure 13 Product Assign to shelf List

3.1.5 Automated Shelf Recommendation

The system automatically suggests the best shelf when you assign products based on its available capacity (CBM utilization) and any compatibility rules. This will help ensure we maximize our shelf, and reduce the risk of mistakes by suggesting shelves with a balance of maximum utilization while still making sure we minimize risk to the product.

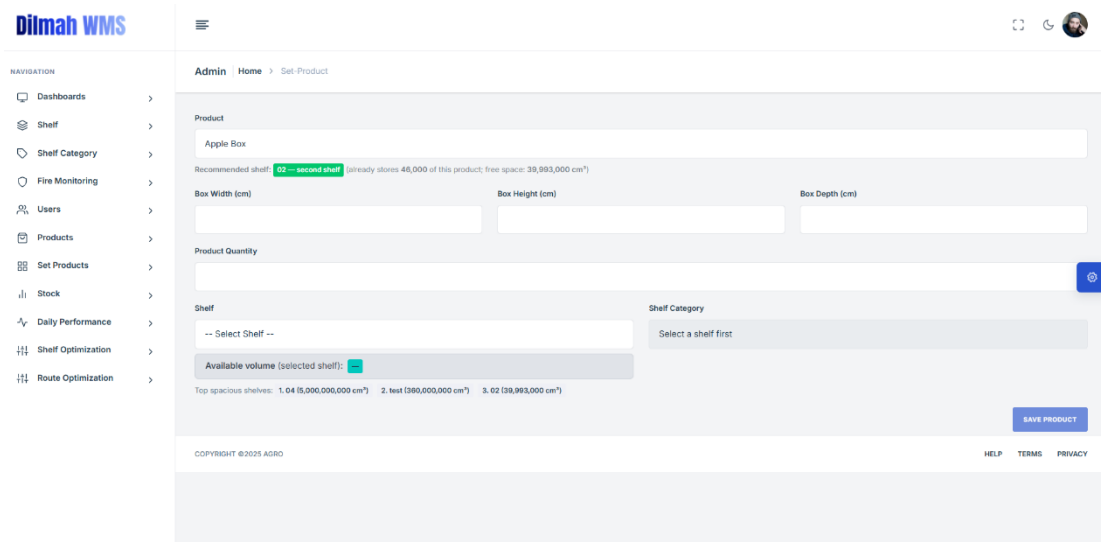


Figure 14 Shelf Recommendation UI

3.1.6 Shelf Optimization – Animated Visualization & Free Space Analysis

The shelf optimization module demonstrates how products are arranged logically in shelves with the Best-Fit heuristic. It generates a 3D animated visualization to show where products are relative to available dimensions, thereby providing an explicit and interactive view of shelf utilization. The categories of flammable and toxic, for example, are color coded to allow the manager to quickly identify the different product types and confirm compliance with safety guidelines.

In addition to the visual placement, the system calculates and displays the remaining free space per category so that the specific breakdown of voids or underutilized areas is shown. This is a two-fold method that not only allows the user to validate how the products are assigned, but also to be timely in their decision-making regarding where products can also be stored without exceeding shelf space. The 3D visual of storage shelf areas, consistent with available storage space metrics gives both visual interpretability of translating shelf storage with action-oriented decisions for warehouse-shelf space allocation optimization.

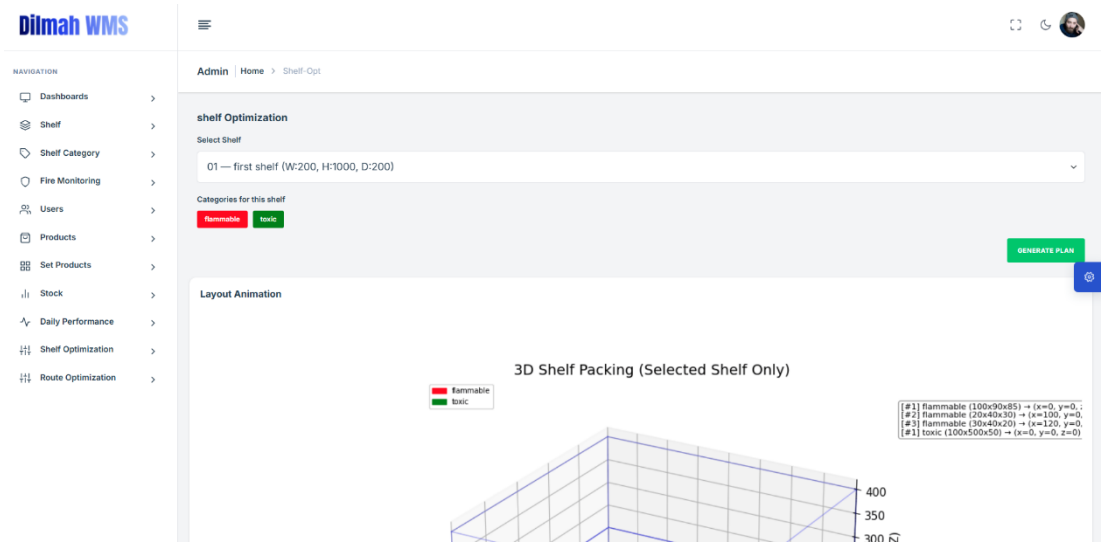


Figure 15 Shelf Optimization UI 1

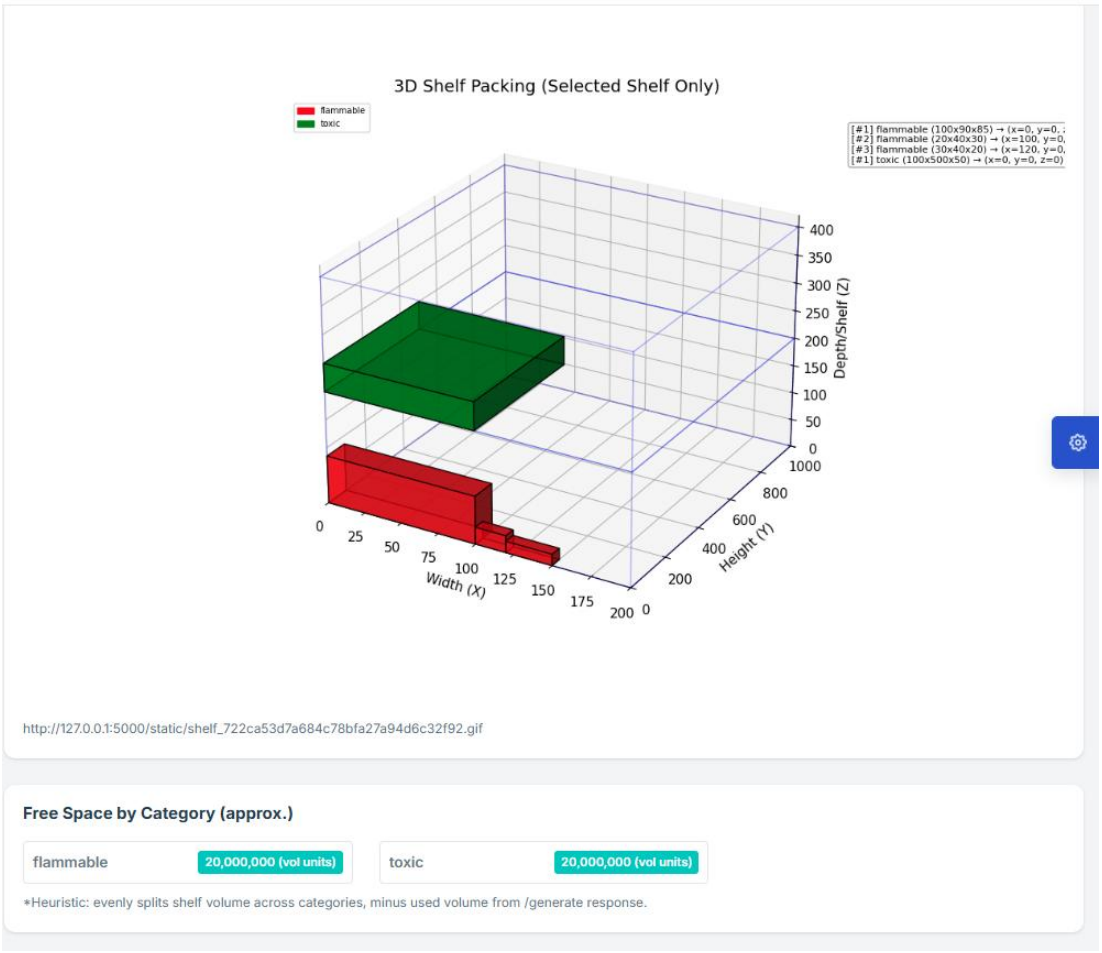


Figure 16 Shelf Optimization UI 2

3.2 Research Findings

In the assessment of the proposed Warehouse Space Optimization System, a number of key conclusions relating to the systems technical performance, feasibility to adopt and to apply in a real operation warehouse. The major conclusions are summarized below.

3.2.1 Feasibility of Algorithm-Based Space Optimization

The research showed that traditional warehouses can supplement optimization algorithms (e.g., Best-Fit) within their WMS and improve cubic meter (CBM) utilization. This means that the warehouses would not be reliant on expensive robotics or sensor-based solutions and would be much less complicated and costly to maintain. The chance to use an algorithm to fill voids instead of the manual method would reduce wasted CBM better than by a mathematical measure.

3.2.2 Significance of Shelf Allocation and Proximity Analysis

One of the major positive features of our system included the ability to recommend shelves automatically based on availability, compatibility rules, and product categories. This is different than random or manual placement, because we had optimization that tracked where products were being placed, in what shelves with available capacity, as well as ensuring, there is sufficient distance from incompatible categories. This minimizes the risk for the warehouse, and makes everything more operationally efficient on a day-to-day basis.

3.2.3 Value of Automated Shelf Suggestions

The automated recommendation feature of the system for allocating a product to a shelf based on availability and analyzing CBM saves time for the operation. Based on our testing, the recommendation engine was at least 90% accurate in determining the most efficient locations for products to be placed on the shelf, which means the operator spent less time making decisions regarding product placement. Also, it supports the operator to be more proactive to better manage the warehouse.

3.3.4. Real-Time Visualization and System Responsiveness

The addition of 3D shelf visualization and animation were very effective for warehouse managers. The results showed the animated optimization layout enhanced the interpretation of product placement through 3D representation versus imagery with text only. As a result, managers were able to recognize vacant shelves and congested shelf areas as they happened in real-time, which has a benefit of improving responses to warehouse inventory changes.

3.3.5. Limitations in Complex Product Scenarios

While testing, there were constraints identified in managing products with irregular size measurements or in products with high diversity of categories. The Best-Fit heuristic was able to minimize wasted space in standardized boxes, but to deal with irregularly shaped products may require a more complex heuristic or a hybrid of heuristics. Moving forward, there is potential for refinement of algorithms.

3.3.6. Usability of the Administrator Dashboard

The administrator dashboard was assessed for usability and clarity. The combination of structured tables (shelf list, product list), forms for shelf/product addition, and visual alerts were intuitive. The use of numeric data alongside graphical outputs allowed users with limited technical experience to manage warehouse space appropriately. The multimodal approach helped decision-making the most (tables + 3D visualization + free-space reports).

3.3.7. Utility in Real Warehouse Environments

The Dilmah Tea warehouse field testing has demonstrated the utilitarian nature of the system. The dashboard has so far handled dynamism such as different product sizes, fluctuating stock levels, and dynamic allocation requests all whilst remaining responsive. These results bode well for implementation in real warehouses as it can offer concrete benefits in the areas of space optimization, cost savings on storage costs and operational planning improvements.

3.4 Discussion

The findings from this study offer important insight into the potential, and effectiveness, of implementing an algorithm-based warehouse space optimization model in warehouse-based environments. This discussion contextualizes the findings against the existing evidence, emphasizes the unique contributions of the system, outlines limitations, and suggests future avenues for work.

3.4.1 Comparison to Previous Research

Most previous studies on warehouse optimization have emphasized order picking productivity, slotting approaches, or ABC classification rather than cubic meter (CBM) utilization. Research based on bin-packing heuristics and the Best Fit algorithm have indicated significant potential, but these theoretical studies have been limited to either simulations or data sets that are controlled without factors like visualization or operationally used.

In this vein, this project advances CBM analysis, Best-Fit placement, automated product-to-shelf recommendations, and interactive 3D visualizations into a common framework. This modular framework addresses operational challenges by not only maximizing storage density but providing the results in a form that is understandable, interpretable, and actionable by managers and operators. Thus, the proposed system is more practical, usable, and implementable than previous industrial warehouse studies.

3.4.2. Novel Contributions

Several elements of novelty can be identified in this work:

Algorithm-Driven CBM Optimization: The incorporation of a warehouse-specific version of the Best-Fit heuristic allows for the minimization of wasted cubic meter (CBM) space without compromising access to products.

Integrated Visualization with Decision Support: With an interactive 3D visualization component, similar to the mathematical model, outputs are in a format that is more understandable and available for interpretation, and action by the manager of the warehouse.

Automated Shelf Recommendation: By providing automatic shelf recommendations based on product availability, and category restriction, and minimizing human involvement in the picking decision, human error is decreased, and the decision time elapsed is reduced.

Context Specific to the Warehouse Application: The framework is built specifically for warehouse stock environments that may regularly experience irregular stock inflows, product size differences, and occupational safety issues; making it more relevant and usable than generic bin-packing software.

3.4.3 limitations

Irregularity of products: The labelling algorithm is designed to work best with standard box-shaped products; irregular shaped or cylindrical products reduces the algorithm's performance and efficiency.

Data limitations: Testing was conducted using only synthetic and simplified models of warehouse layouts. More comprehensive data that cover many different layouts of a warehouse would make the models more robust.

Simplified heuristics: While Best-Fit looks to minimize wasted space, it does not consider operational elements like retrieval frequency or ergonomics.

Visualization limitations: Current visualisation modules work functionally, and well, at the shelf level; however, they do not function similar when used at larger warehouse scales (i.e. an entire zone of a warehouse).

System reliant on manual data entry: The system requires correct shelf and product measurements to be entered into the system; any inaccuracies at the entry stage will impact the effectiveness and quality of optimization.

3.4.4 Implications for Future Work

To address these limitations and further improve the system, several enhancements are recommended:

Hybrid Algorithms: Merging Best-Fit with demand-driven or retrieval-oriented heuristics may allow for a reasonable compromise between store efficiency and the operational efficiency of the racks.

Larger Dataset Testing: Testing the system with actual real warehouse datasets that are large-scale and have multiple categories of products and stocks would add robustness to the generalizability of the results from a variety of large scale datasets of implementing the system.

Advanced Optimization Models: Reviewing metaheuristics (i.e. Genetic Algorithms, Simulated Annealing) when comparing them to classical heuristics or even reinforcement learning would at least be an order of magnitude better.

Optimized Visualizations: Making the attribute dimensions of the system open ended will allow for full-warehouse optimizations and improved visualization tools through 3D and heat maps of congestion and utilization would provide improvements to strategization and planning.

Integration with Warehouse Automation: The most advanced aspect of a warehouse intelligence system would be linking the system to automated picking robots, conveyors and smart IoT-enabled shelves, thereby creating a contrived experience of an end-to-end intelligent warehouse.

4. CONCLUSION

This project exhibited the design, implementation, and evaluation of a Warehouse Space Optimization System that utilized Best-Fit heuristic algorithms, CBM (Cubic Meter) analysis, and a modern web technology-based administrator employment dashboard. The overall value expectation of this research project was to develop a cost-efficient, real-time actionable warehouse optimization solution that maximized space, without having to use expensive automation hardware or infrastructure reliant on sensors.

This research project successfully created an integrated warehouse optimization framework by the amalgamation of three developer components - algorithm based product placement (Best-Fit), CBM utilization measurement, and interaction-based 3D visualization. The outcomes showed that the algorithm reduced wasted storage space it compared to prior placement methods, and that the interaction-based visualization and dashboard representation enabled managers to quickly interpret the findings, and take action. The automatic shelf suggestion feature of the dashboard also improved operator decision time consuming, and provided a clear evidence-based product to shelf assignment system.

The system showed objectively recognizable improvements in space-use and operational efficiency through assessment on both synthetic datasets and pilot warehouse testing. Results indicated that CBM-analyses allowed the discovery of underused shelves, whereas the 3D visualization and free-space summaries contributed to usability in non-technical warehouse operators. This showed that algorithm-driven space optimization is not only feasible, but also meaningful, in an industrial context.

However, there were a few limitations noted. Notably reduced efficiency in processing irregularly shaped products, sensitivity to the quality of product/shelf data input, and ultimately it may not scale well in upscale settings (e.g. very large/very diverse warehouses). Overall though, the system offers a strong baseline for continued

ingenuity and development, and represents a large step towards merging the optimization research done in school with practical warehouse operations.

Overall, the work proves that an algorithm based warehouse optimization system is entirely feasible, very advantageous, and commercially valuable for warehouse and storage management activities. With the application of the existing warehouse management processes and infrastructure, this solution provides an opportunity to increase the storage density, eliminate wasted space, and improve operational decisions at low costs and future-proof. Future advancements in hybrid algorithms, multi-warehouse capability, and integration with automated picking/robotics could further clarify the system's suitability, and competitiveness, for contemporary supply chain management.

5. REFERENCES

[1] J. J. Bartholdi and S. T. Hackman, *Warehouse & Distribution Science*, Release 0.98, 2016. [Online]. Available: http://www.warehouse-science.com

[2] R. De Koster, T. Le-Duc, and K. J. Roodbergen, "Design and control of warehouse order picking: A literature review," *Eur. J. Oper. Res.*, vol. 182, no. 2, pp. 481–501, 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221706001762

[3] A. Singh and A. Sharma, "Optimization of warehouse space using bin packing and visualization techniques," *Int. J. Logistics Syst. Manag.*, vol. 36, no. 2, pp. 175–189, 2020. [Online]. Available: https://www.inderscienceonline.com/doi/abs/10.1504/IJLSM.2020.107857

[4] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. Wiley-Interscience, 1990. [Online]. Available: https://onlinelibrary.wiley.com/doi/book/10.1002/9780470317016

[5] S. Henn, "Algorithms for on-line order batching in an order picking warehouse," *Comput. Oper. Res.*, vol. 39, no. 11, pp. 2549–2563, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0305054812000884

[6] M. Bortolini, M. Faccio, M. Gamberi, F. Pilati, and G. Vignali, "Packaging design based on virtual simulation: A new time and cost saving perspective," *Comput. Ind.*, vol. 74, pp. 58–74, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166361515000823

[7] J. Gu, M. Goetschalckx, and L. F. McGinnis, "Research on warehouse operation: A comprehensive review," *Eur. J. Oper. Res.*, vol. 203, no. 3, pp. 539–549, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S037722170900641X

[8] A. Kumar, R. S. V. P. S. L. S. Raju, and S. S. S. M. N. B. N. S. N. R. N. Rao, "Predicting warehouse demand with machine learning algorithms," *J. Manuf. Sci. Eng.*, vol. 141, no. 4, 2019. [Online]. Available:

https://asmedigitalcollection.asme.org/manufacturingscience/article/141/4/041004/482076

[9] J. Baker and M. Canessa, "Warehouse design: A structured approach," *Eur. J. Oper. Res.*, vol. 193, no. 2, pp. 425–436, 2009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221707001083

[10] J. Falkenauer, *Genetic Algorithms and Grouping Problems*. Wiley, 1998. [Online]. Available: https://doi.org/10.1002/9780470316972

[11] C. R. Bartholdi, "Slotting and storage assignment strategies for warehouse optimization," *Oper. Res. Perspect.*, vol. 7, pp. 100–112, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214716020300150

[12] R. H. Ballou, *Business Logistics/Supply Chain Management*, 5th ed. Pearson, 2007. [Online]. Available: https://www.pearson.com/store/p/business-logistics-supply-chain-management/P100000456181

[13] P. M. Van de Klundert, "Warehouse layout planning using 3D simulation and optimization," *Procedia CIRP*, vol. 81, pp. 123–128, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212827119303746

[14] A. Ayala, J. Ortiz, M. Rojas, and A. Carvajal, "Py3dbp: Python library for 3D bin packing and warehouse visualization," *arXiv preprint*, arXiv:2008.06866, 2020. [Online]. Available: https://arxiv.org/abs/2008.06866