SLOTTIFY : A WAREHOUSE SLOTTING TOOL

by

Karthik P (2448331)
Lis Mary Lijo (2448335)

Under the guidance of
Dr. Priya Stella Mary I

A Project report submitted in partial fulfillment of the requirements for the award of the degree of Master of Science (Data Science) of CHRIST (Deemed to be University)

September – 2025

<u>CERTIFICATE</u>

This is to certify that the report titled ***“SLOTTFY : A WAREHOUSE SLOTTING TOOL*** *is a bonafide record of work done by* ***KARTHIK P*** *of CHRIST (Deemed to be University), Bengaluru, in partial fulfillment of the requirements of IVth Trimester MSc (Data Science) during the academic year 2025-26.*

Head of the Department **Project Guide**

Valued-by

1.	Name	:
	Register Number	:
2.	Date of Exam	:

ACKNOWLEDGMENT

I would like to express my sincere gratitude to all those who have contributed to the successful completion of my Specialization Project. Their support and guidance have been invaluable, and I am deeply thankful for their assistance throughout this journey. First and foremost, I would like to extend my heartfelt thanks to **Dr Fr Joseph C C, Vice Chancellor, Campus Director Dr. Fr. Benny Thomas, Associate Director Dr Fr Biju K Chacko, Campus Administrator Dr. Fr. Vinod Michael, Dr Joby Thomas, Dean, CHRIST (Deemed to be University)- Yeshwanthpur Campus** for letting us carry our specialization project of M.Sc Data Science in our Campus.

I would like to extend my heartfelt thanks to the HOD of the Computer Science department of CHRIST (Deemed to be University), Bangalore Yeshwanthpur Campus, **Dr. Vinay M**, for fostering a conducive learning environment that has been instrumental in shaping m y academic experience.

I am also immensely grateful to my project guide, **Dr Priya Stella Mary I** for her expertise, mentorship, and unwavering support throughout this project. Your patience and dedication to my growth as a data scientist have been instrumental in the successful completion of this project. I would like to thank all the faculty members of the Department of Computer Science for their help to complete the project work successfully. Also, I would like to thank my fellow students and colleagues for their camaraderie, which made this academic journey both enjoyable and enriching. Last but not least, I want to express my gratitude to my family and friends for their unwavering support, understanding, and encouragement throughout this challenging but rewarding journey. Your belief in me has been a constant source of motivation. In conclusion, this project would not have been possible without the guidance and support of the individuals mentioned above and the collective effort of everyone who played a part in my academic pursuit. Thank you all for your contributions, and I am deeply appreciative of your help.

Karthik P

Reg No:2448331

ABSTRACT

One of the key factors that dictate how efficient the operations of the supply chain are is strategic placement or location of inventory, which is known as warehouse slotting. It is determined that the old slotting schemes (either manual ones or formed with the help of naive heuristics as the ABC analysis) cannot be efficient in the dynamic environment with many dimensions offered by modern warehouses, and that the ineffective paths of picking processes, poor use of space, and excessive labor costs are the result of this defect. Lukana resolves these inefficiencies through the design and development of Slottify, which is a data-based smart slotting recommendation system. This project is designed to use machine learning to come up with a better slotting strategy.

A Random Forest model that examines an enriched space of characteristics such as physical size, speed of the item, and compatibility of ordering behaviour, to produce a holistic slotting score of that Stock Keeping Unit (SKU), is the core of the system. These scoring points are then used in another algorithm, to reserve SKUs to the best available warehouse locations. The ultimate result is an effective web-based prototype that can ingest warehouse data and generate a list of slotting assignment priorities. Testing the model on a simulated dataset through the Normalized Discounted Cumulative Gain (NDCG) metric shows that the ranking of quality is significantly improved with respect to traditional methods.

The majority of logistics and supply chain operations are encouraged to provide these methods with an opportunity, machine learning and training to occur, and multiply with high efficiency and cost-effectiveness because such processes are predicated on the concept of multi-factors.

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENTS

1. INTRODUCTION

1.1 PROBLEM DESCRIPTION

The warehouse or distribution center is a key nexus in the complex and dynamically evolving environment of modern supply chain management, where goods travel the path back to the source of production in the end consumer. The effectiveness of this nexus is determined not only by storage capacity, but is also deeply dependent on the intelligence of its internal level of organization. The core operation of this organization is warehouse slotting: the organizational task of determining the allocation of inventory (goods) or the Stocks Keeping Unit (SKU) into certain storage points, in order to streamline the main process of pallet operations of picking, packing and replenishing goods in warehouses. Although it appears as a tactical issue, the aggregate effect of effective slotting makes it a strategic priority that has quantifiable and immediate implications on the profitability and competitive position of a firm.

The main advantages of a good slotting strategy can be described as complex, as they are related to the efficiency of the operation, cost reduction, and customer satisfaction. Operationally the largest benefit is the shortening of picker travel time. The travel time of a picker can represent up to 60-75 percent of all working hours within a large warehouse operation, so by locating high demand and fast-moving products in the most accessible locations, like directly near a packing point, ergonomic favorable, or the so-called golden zones, companies can reduce this non-value-added travel time dramatically. According to industry statistics, slotting optimization can lower picking travel time by up to 30 to 55 percent which directly translates into a significant amount of labor productivity growth with no extra headrolls dedicated to managing bottlenecks and other disruptions in the operations.4 A logical layout also streamlines the flow of people and equipment, which leads to a significant reduction in operational sharing.

Financially, these efficiency gains contribute to very high cost reductions. The reduction in labor costs due to the increased rate of picking and decreasing spillover leads to a decrease in the number of mistakes in instructing products results in better use of space by locating heavy or bulky items safely and in their correct locations, and results in a reduction in carrying costs related to the increased accuracy of inventory control.

These are all internal improvements, which in the end lead to a better experience with the customers. The shorter order fulfillment cycles allow better performance on much shorter delivery windows, another primary point of difference in the current e-commerce environment, which, furthermore, lowers shipping errors, ensures that customers receive the right product at the right time, and lowers the overhead costs associated with handling returns and service remedies.

The flexibility and ability to scale of a warehouse are directly related to the level of intelligence of the slotting system. With a deployed facility that has stagnant or haphazardly designed layouts, they will inevitably experience the complexities ceiling as order volume and SKU levels increase and create cascading inefficiencies. Contrarily, a warehouse developed on the basis of dynamic, data-driven slotting background is placed to grow sustainably. Investments in an intelligent slotting system is therefore not a reactionary response to address current inefficiencies, it is an actionable (strategic) response to a forthcoming business response in terms of agility. It changes the warehouse itself, and the warehouse becomes a formidable competitive weapon, able to provide in seconds, milliseconds, reliability and ability to meet customer satisfaction and bottom-line results.

1.2 EXISTING SYSTEM ANALYSIS

Although there are evident strategic advantages to optimized slotting, most warehousing tasks are still rooted in traditional practices that are ill-suited to the demands of contemporary inventory. These processes start with wholly manual, one-off systems, to slightly more orderly yet fundamentally incorrect systems such as those of ABC analysis. Continuations of these old ways of doing things present one of the greatest and increasing discrepancies between the potentiality of operation and outcomes.

Slotting choices at most warehouses (especially where the management did not use a complex and well-developed Warehouse Management System or WfMS), is often reactive. With the entry of new products, they are stacked anywhere space is available, in a haphazard fashion with little thought as to strategy. Eventually, the ad-hoc process causes some state of entropy in the organization. The total footprint area needed to move the picker, the picking footprint, commensurately increases as the high-speed products are dispersed throughout the facility, disconnecting with their original locations, where they are most beneficially located close to packing and shipping areas.

With this corrosion of the warehouse scheme, a chain of operational breakdown occurs. When on-hand inventory is recorded in written records using pens and paper, the chances are that this will cause human error and the end effects of this error will be that with an ineffective mechanism of slotting the product, the majority of SKUs will never be in the most optimal place, an issue intensified by the fact that with paper records, managers have no real-time insight into stock levels or location, nor a problem underestimated by the reality that although a system may place no difficulty in recording certain parts of the inventory, the same system will make it impossible to actually locate those parts.

1.3 PROJECT SCOPE AND OBJECTIVES

This project is based on the design and development of Slottify; an intelligent and data enabled warehouse slotting recommendation system to overcome the abysmal constraints of the traditional and heuristic-based slotting techniques. Slottify might be imagined as an online Web tool that will outgrow the one-dimensional, simplistic logic of ABC analysis, and adopt a multi-factorial, machine-learning methodology.

The Slottify system relies on a Slottify core in the form of a Random Forest model, an effective and resilient supervised training algorithm. In contrast to conventional training techniques that use one, fixed rule (e.g., inventory value), the Random Forest model can be trained to consider a diverse and adequate number of features of each SKU. These attributes cover not just historic sales and pick velocity but also physical (cube size, weight), demand (seasonality, variability) and a relationship attribute (product affinity). The model can produce a subtle, situation-driven slotting score by learning the complicated, non-linear connection among the present features and the operational significance of an SKU.

This score is a qualitative display of the degree to which an SKU needs an available and
prime storage facility. An additional ranking algorithm is then used to intelligently stepwise place the full inventory of SKUs in the available warehouse slots based on these scores. The system sorts all SKUs in descending order of slotting score and all available slots in order of an accessibility score (which depends on things such as distance to packing stations and ergonomic height). It then produces an optimized slotting programme through relocating the SKUs with the highest priority to the most convenient slots. The outcome is an active, clever, and defendable collection of slotting suggestions created to achieve optimal efficiency in the warehouse.

1.4 PROJECT OVERVIEW

1.4.1 AIM

To create a smart warehouse slotting system, Slottify, that elevates existing systems as each intelligently attempts to refine the selection of the warehouse spot slot.

1.4.2 OBJECTIVES

To perform an in-depth discussion of the problems and inefficiencies associated with manual and ABC-based warehouse slotting systems.

To create a web-based slotting recommendation tool that segregates data processing and model inference process with user presentation layer into a strong and scalable system architecture design.

To design and train a random forest model that will be able to score SKUs accurately over a wide range of operational features, such as velocity, size, and order affinity, etc.

To apply a ranking algorithm which uses the model generated scores to put SKUs into the best warehouse slot based on location accessibility.

To create a working version of the Slottify application which consists of the backend API and a user interface that will show the generated slotting recommendations.
To strictly analyze the output performance of the machine learning model based on the right ranking metrics that quantitatively show that the model is better than the baseline methods.
This project scope covers the entire scope of the life cycle of the Slottify prototype, according to the process model.
This includes:
Data Management: The purchase, cleaning, preprocessing, and feature engineering of a simulated or historical dataset that manifests a normal warehouse setting including a large volume of SKUs and orders.
Model Development: Training, hyperparameter optimization and testing of a Random Forest regression model and application of a deterministic ranking algorithm.
Application Development: Developing a backend API based on Flask framework and serving predictions done by the model, a simple user interface with HTML, CSS and JavaScript to show the essentials of the working system.
Tests: Fully software testing, including: individual component: unit tests, other testing (integration): API testing, other testing (integration): API testing, User Acceptance Testing (UAT) plan, done to verify the business requirements.

As project elements, the following areas are strictly off scope and thereby non-negotiable:
Live System Integration: Real-time, two-way integration with production live Warehouse Management System (WMS) will not occur in the project. In static file input (e.g. CSV), data will be ingested in a batch fashion.
Mobile Application: It is not within the scope of the present analysis of the development of a native mobile application that will be used by the warehouse operator. The prototype type will be a web based application.
Physical Automation: The doctrine of this project is to consider only the decision

side in slotting. It does not include the physical performance of putaway or re-slotting activities such as mobilization of robotics or other automation performance equipment.

Complex state of the art LTR Models: the project assumes a ranking methodology, although it will not use fancy, state-of-the-art Learning to Rank (LTR) algorithms like LambdaMART. It is all about proving the essence of ranking by using a machine learning score.

Table 1.1 Comparison of Slotting Methodologies

Feature	Manual / Ad-hoc Slotting	ABC Analysis	Slottify (ML-based)
Decision Basis	"Empty space available "; tribal knowled ge	Single factor: Inventory value	Multi-factor: Velocity, cube, weight, affinity, seasonality, etc.
Adaptability to Seasonality	None; entirely reactive and requires manual re-work	Poor; static classification requires manual updates	High; model can be retrained on recent data to adapt to changing demand patterns
Consideratio n of SKU Affinity	None	None	Yes; co-location of frequently co-ordered items is a key feature

Space Utilization	Very poor; leads to honeycombing and wasted space	Moderate; better than manual but ignores SKU dimensions	High; optimizes based on cubic movement to fit items to right-sized slots
Labor Efficiency	Low; results in long, inefficient pick paths	Moderate; improves paths for 'A' items but neglects others	Very High; systematically minimizes travel time for all items based on true priority
Data Requirements	Minimal	Moderate; requires sales/inventory value data	High; requires comprehensive SKU, order, and warehouse layout data

2. SYSTEM ANALYSIS

2.1 FUNCTIONAL SPECIFICATIONS

As the overall target of the Slottify project set in Chapter 1, this is to create a better slotting suggestion system. This objective can be subdivided into a number of specific business goals that will be fulfilled through the final system:

Primary Objective: Minimize the time of Order Picking. Minimizing the travel distance and time of warehouse pickers is the most influential objective on its own. Since most of the warehouse operating time is devoted to picker travel, any savings

that can be made here will have an instant and substantial impact on productivity. Slottify will make efforts to ensure that items that are frequently picked will be placed in the most convenient places available.

Secondary Notice: Augment the use of Warehouse Space. The system will utilize full potential of the storage space through the combination of physical dimensions (cube size) of SKUs with the velocity. This rightsizing of slots allows rapid moving items to be stored in small areas where such items could stay without affecting fast moving products that can occupy bigger areas.1

Secondary Objective: lessen Picking error. One error that is likely to occur is the incorrect picking of items of similar appearance. The Slottify algorithm will also include the logic to remove SKUs that share similarities in terms of their characteristics, to enhance picking accuracy and to lower the cost of misstating shipments along with returns.

Secondary Goal: Become more Ergonomic and Worker-safe. Such a system will take into consideration the weight of goods and it will become a norm where heavy or heavy goods shall be positioned in such areas, which we may call the golden robes-easy to work areas which extend to the bones within the body restricting the movement of the body in terms of bending and reaching. Not only does this increase the rate of picking, but it also increases the level of safety at the work place and minimizes the chances of injuring one of the workers.

2.2 SYSTEM REQUIREMENTS

In an actual enterprise context, the data needed would often be divided into a number of business systems. The warehouse layout and SKU records are most commonly held in the Warehouse Management System (WMS). Any order history, and Sales information is the result of the Order Management System (OMS) .The capability to integrate and reconcile these fragmented sources are the key to the successful implementation of a system such as Slottify.

To know the features of this data, initial exploratory data analysis (EDA) has been performed. This included the creation of histograms of SKU pick counts which showed a common long-tail distribution with only a small percentage of SKUs taking up a large percentage of the pick counts. Product dimensions and weights

were also considered to learn about the physical composition of the inventory. Primary data quality checks have been conducted to pinpoint any missing values or inconsistencies (e.g. negative values or zero dimensions) that will require actions to be taken at the Data Preparation stage.

2.3. TECHNOLOGY STACK ANALYSIS

The functional specifications specify what the Slottify system should actually do and be able to do to the user.

F1 Data Ingestion: The system will offer a mechanism through which a user is able to input the necessary SKU, warehouse and order data in required format (e.g., CSV).

F2: Feature Generation: When data is ingested, a pipeline will automatically be generated to compute a set of generated features on each SKU, such as pick velocity, cubic movement and product affinity scores, among others.

F3: SKU Scoring: The system will use the trained Random Forests to run the engineered features and obtain single numerical score, which can be called as slotting score of each SKU included in the dataset.

F4: Slot Ranking: The system will compute an accessibility score of each of the warehouse slacks available to it, depending upon its location relative to important facilities, e.g., packaging zones, and its ergonomics (e.g., height).

F5: Recommendation Generation: This system will provide an optimised version of a slotting plan to the user by aligning the most high-scoring SKUs and slots, and show them to the user in a tabular format.

2.4. NON-FUNCTIONAL REQUIREMENTS

The non-functional requirements define the quality characteristics and constraints of the system, the degree to which it achieves its functions.

NF1: Performance: Given 10,000 SKUs of data to read, represent in the feature engineering phase, and use them in model inference and subsequent recommendation generation, the end-to-end process must take less than a reasonable amount of time (target: less than 30 minutes) on the specified hardware.

NF2: Usability: The GUI will be user-friendly and easily comprehensible, and a warehouse manager will need minimum training to use it. The overall final recommendations should be provided in an easy to understand form, including the important information required for making a decision.

NF3: Maintainability: The source code to the whole system will be highly modular, properly documented, and meet any prescribed standards of code structure (PEP 8 to Python). This will make further updates, bug fixing and improvements easy.

2.5 HARDWARE REQUIREMENTS

Minimum: 4-core CPU, 8 GB RAM, 20 GB free disk space.

Recommended (for faster training): 8-core CPU, 16 GB RAM, 50 GB free disk space.

2.6 SOFTWARE REQUIREMENTS

Operating System: Windows 10/11, macOS, or a Linux distribution.

Backend Runtime: Python 3.8 or higher.

Python Libraries:

Pandas (for data manipulation)

NumPy (for numerical operations)

Scikit-learn (for machine learning model)

Flask (for web application server)

Database: SQLite 3 (for the prototype).

Web Browser: A modern web browser such as Google Chrome, Mozilla Firefox, or Microsoft Edge

3. SYSTEM DESIGN

3.1. SYSTEM ARCHITECTURE

Slottify implements a multi-level architecture to implement the separation of concerns to achieve high modularity, scalability, and maintainability. There is a data layer, processing/application layer (backend) and a presentation layer (frontend).

3.1.1. HIGH-LEVEL ARCHITECTURE DIAGRAM.

The Slottify system is structured in a way that allows data to flow appropriately and effectively through the system to the end presentation as recommendations to the end user. The main parts and their relationships are shown in Figure 3.1.

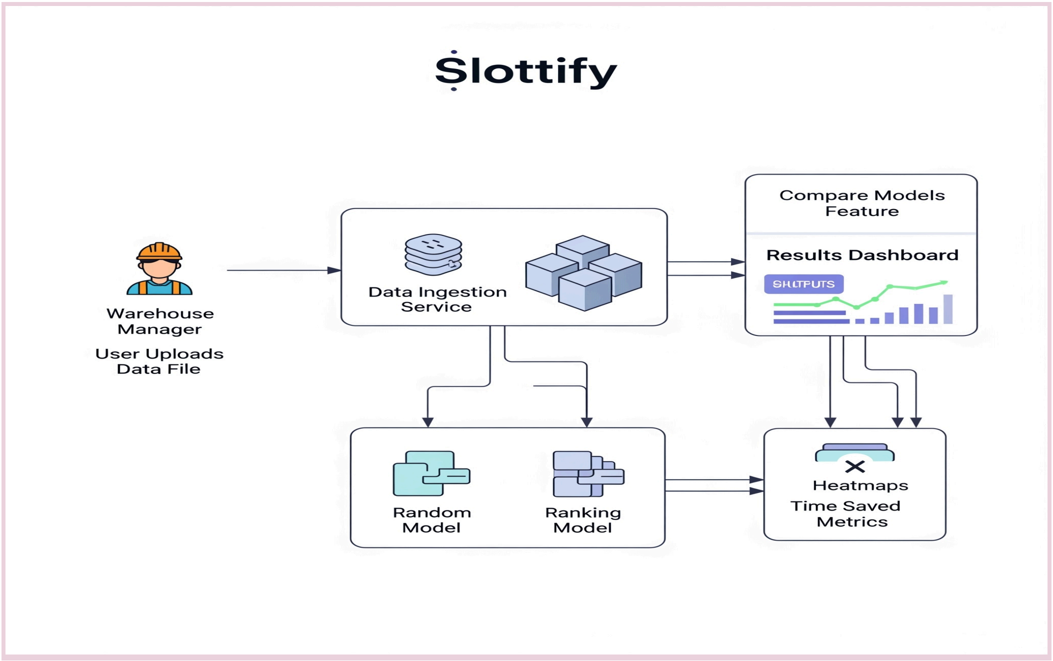


Figure 3.1.Slottify System Architecture.

Data Layer: This base layer is made up of the plain data sources. In the case of the prototype, they are stored in the form of CSV files . In a manufacturing world, it would be a relational database which gets linked to the business WfMS and ERP systems.
Processing Layer & Appearance Layer (Backend): This is the center of the system, which is written in Python. It is further divided into:

Data Processing Pipeline: A collection of code in Python relying on Pandas and NumPy libraries that perform data ingestion and cleaning, as well as data transformation and feature engineering. This pipeline converts the raw data to an input ready to construct a model.
ML Model Engine: This is a Scikit-learn implementation that performs the training of a Random Forest model and, after training, makes inferences to produce slotting scores on new data. This engine saves the trained model artifact (e.g. a .pkl file) and reads it.
Flask Web Server An open-source web server that provides a window into the workings of the system via a RESTful interface. Orchestrates the process by taking requests in the frontend, inputting information into the ML Model Engine, running the ranking logic, and returning the results in a JSON format.
Presentation Layer (Frontend): This is the client side component that the end-user will access. It is based on a regular web browser displaying the user interface which is written using HTML, CSS, and JavaScript. The frontend will handle user request to the backend API and dynamically show the returned slotting recommendations in a user friendly format.
The ingestion of raw data into the processing pipeline will be the starting point of the data flow. These engineered characteristics are then processed into the ML Model Engine to score. This is handled by the Flask server which in turn provides the final, ranked set of recommendations when it is asked to do so by the user browser.

3.1.2. TECHNOLOGY STACK

The technology stack selection was informed by the philosophy of rapid prototyping, ease of use, and maturity of the open access libraries to support data science and web development.
Backend:

Program Language: Python 3.9.
Dealing with Data: pandas, Numpy
Machine Learning: Scikit-learn.

Web Framework: Flask 26
Frontend:
Structure: HTML5
Styling: CSS3
Interactivity: Vanilla JavaScript

Database:

Prototyping: SQLite

Deployment (Conceptual):

Containerization: Docker to involve a reliable and repeatable ecosystem to develop and potentially a deployment.

3.2.1. DATA FLOW DIAGRAM (DFD)

A Data Flow Diagram enables the representation of data flowing through the Slottify system. It gives a logical description of the processes of the system without describing the physical implementation.

(At this point (A Level 0 Context Diagram) would be displayed, with only one process being Slottify System and one external object being Warehouse Manager, data input as Warehouse Data and data output as Slotting Plan).

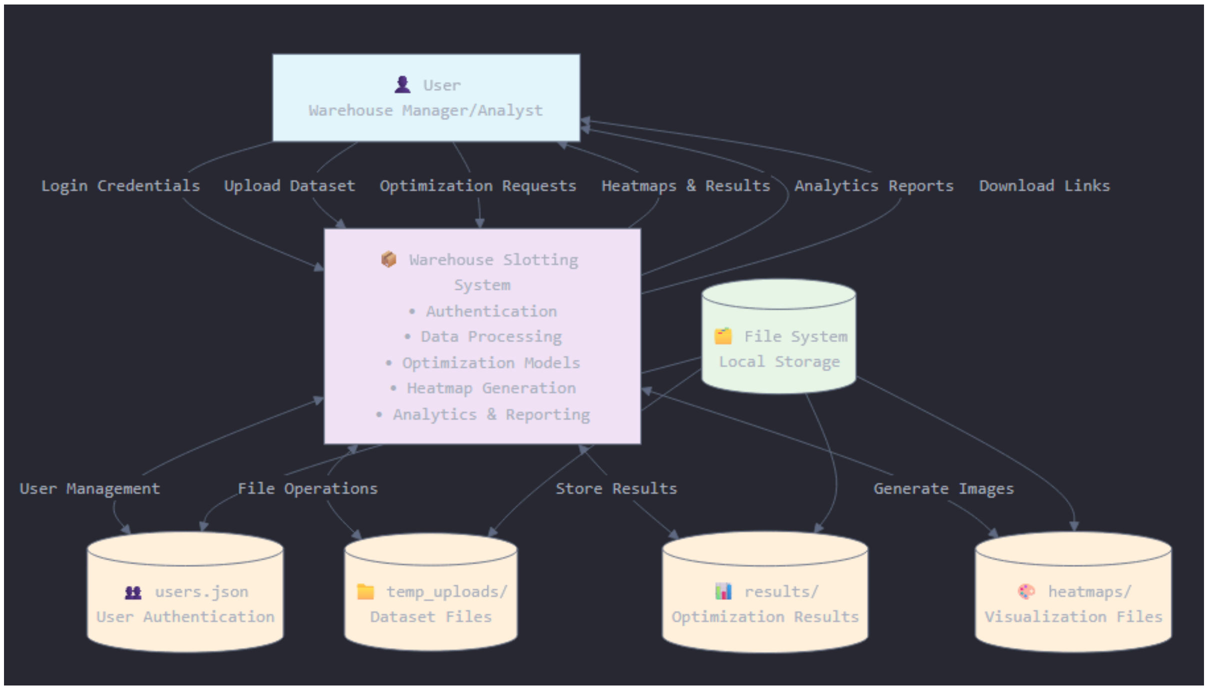


Fig 3.2 Level 0 DFD

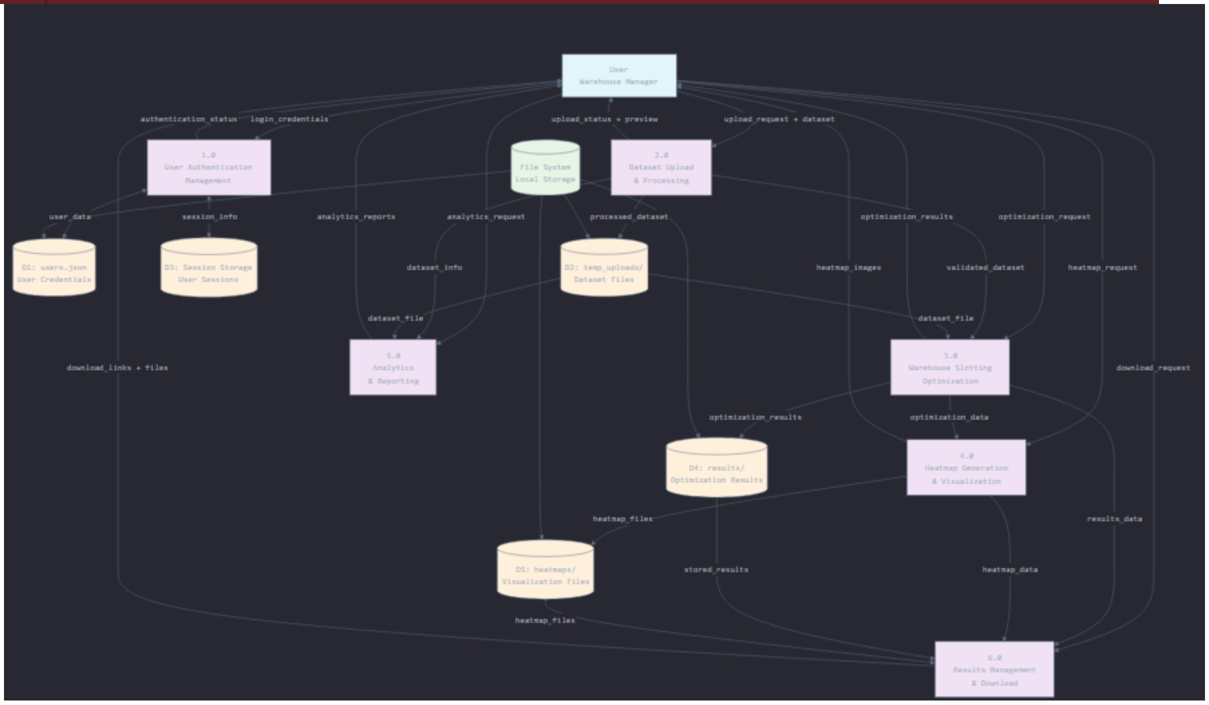


Fig 3.2. Level 1 (DF)

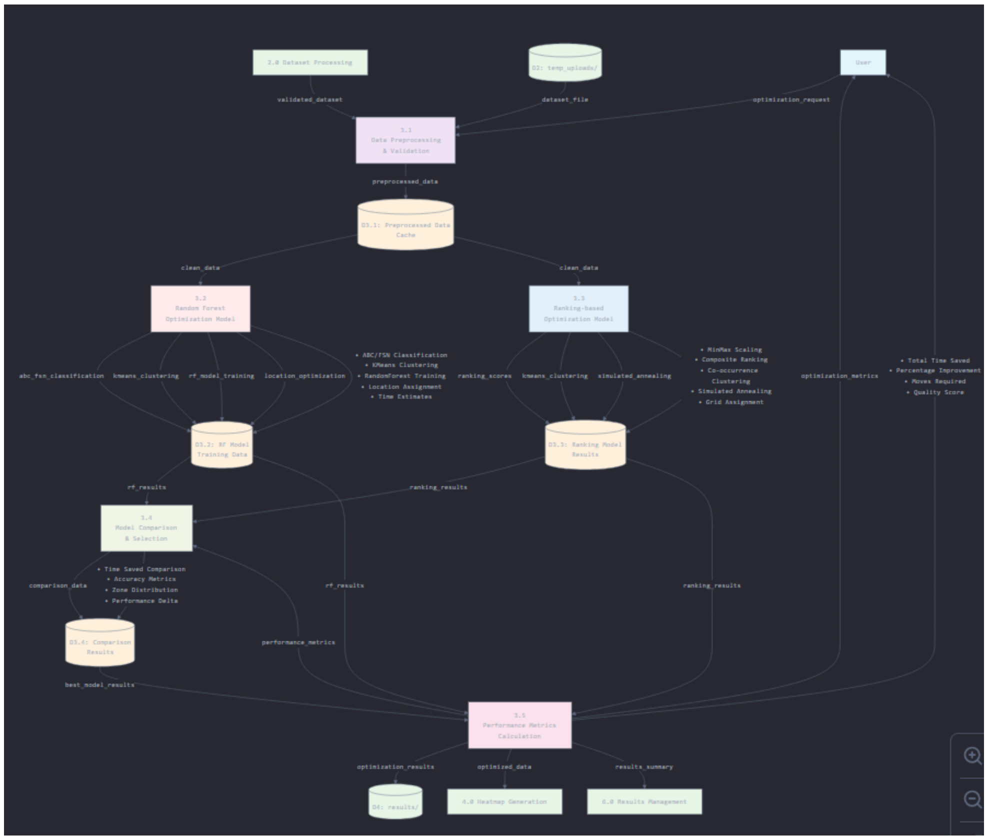


Fig 3.2. Level 2(DF)

3.2.2. STRATEGY FEATURE ENGINEERING/SELECT

Formation of the informative features of the raw data is the important and critical action in feature engineering, which can subsequently be utilized by the machine learning model to make better predictions. This can be the most influential step in a data science project because the quality of the features dictates the quality of the model directly.31

The raw data sources will be engineered into the following features:

Velocity Features: pickfrequency: based on counting the number of times a skuid is present in the orders.csv file. This is an absolute frequency measure of handling.

salesvelocity: Numeric value is derived by adding quantity per skuid in orders.csv. This is an overall product movement.

Demand stability: The SL for all SKUs, computed as the fraction of sales occurring in a single month divided by the average carbon an SKU needed to sell in a month. This assists in making the distinction between items with a fixed demand (low variation) and items with intermittent demand (high variation), similar to an XYZ analysis.33

Physical & Combined Features: cubesize: computed such that the lengthcm widthcm heightcm of each SKU is calculated.

cubicmovement: This is a powerful composite feature which is cubesize x pickfrequency. This measure shows the cumulative amount of space that is transported on a given SKU within a specific time and is both a measure of size and velocity in a single figure.

weightcategory: A category of feature derived by stepping the weight into defined categories (e.g. a Light classification 5kg lighter, a Medium classification 5-15kg heavier, a Heavy classification heavier than 15kg) to simplify rules on ergonomics of the model.

Affinity Features: affinityscore: This will be produced by the analysis of the item co-occurrence in the orders.csv data. A set of co-occurrences will be constructed and, on a per SKU basis, a score will be computed reflecting the propensity of that SKU to be co-purchased with other high-velocity items.

Skills (to be ranked):

slotaccessibilityscore: The value will be calculated on each slot of the warehouselayout.csv file. It will be a negative of the Manhattan distance of a slot in reference to a facial position (0,0,1) and the level of the slot vertically. It will be a formula like Score=1 / aisle + bay + level so w is a height factor. Slots that are nearer to the packing station and positioned at a low level will get a higher score.

Feature Selection:
Random Forest algorithm has an internal feature importance system that will evaluate the importance scores of features after the original model is trained on all engineered features. Features that have very low weight can be dropped off the final model to make the model more readable, or to train more quickly, and potentially alleviate overfitting.

3.3. MODELING

This subtopic explains the machine learning models which have been selected to drive the Slottify system and why they were selected.

3.3.1. RANDOM FOREST FEATURE

Distilling all the engineered features into one, actionable slotting score of each SKU is the main modeling challenge. To complete this task the Random Forest algorithm was chosen. Random Forest operates as a supervised, ensemble learning algorithm that generates as many decision trees as possible during the training stage and renders a mean forecast (in regression) of the constituent trees.
It can be used in this project because it has the following most important peculiarities:
Stability and Accuracy: Additional Forest purges the threat of overfitting that an individual decision tree might experience, creating a more finished and precise field.
Manages Non-Linearity: Warehouse dynamics are not usually linear. Random Forest is also sensitive enough to fit non-linear relationships between features that might not be explicitly known (e.g. the relationship between cubic movement and slotting priority).
Importance of features: The algorithm includes a robust built-in way to determine each feature contribution to its predictions, which is crucial to model interpretation and validation, allowing us to check that the algorithm is emphasizing operationally significant factors and achieving some measure of interpretability to the end-user.
A Slottify RandomForestRegressor will be trained in the context of Slottify. The input will consist of the matrix of engineered features across all SKUs, and the target

variable will be a proxy of slotting priority (the normalized pickfrequency may be selected initially). It is the intricate combination of all features that the model will learn to predict a clean and final slotting score.

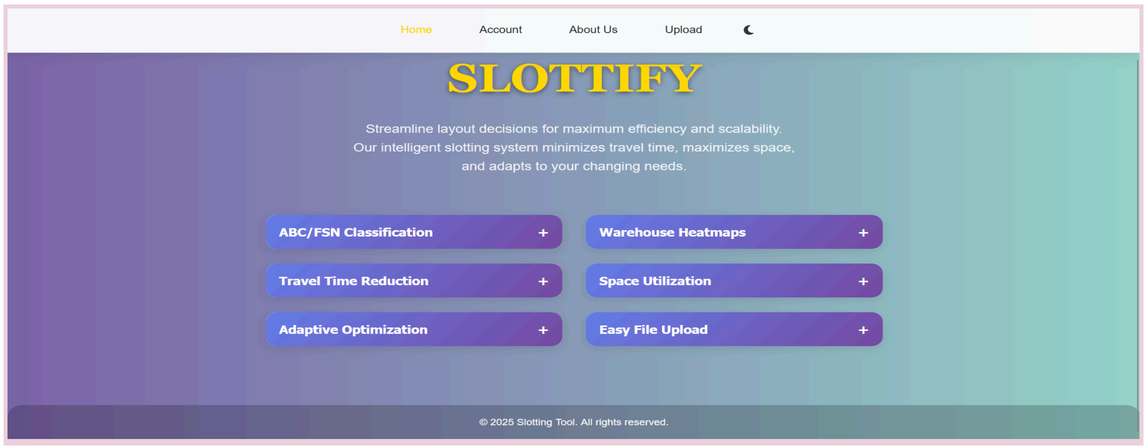
3.3.2. LEARNING TO RANK IN SLOT ASSIGNED.

After assigning slottingscore to each SKU, the last thing to do would be to allocate them to physical locations in the warehouses. It is essentially a problem of ranking. Learning to Rank (LTR) is a specialized subdivision of machine learning based on the construction of models that operate on lists of items to rank them into order.
Specific methods can be split into three types: pointwise, pairwise, and listwise, with listwise methods typically being capable of producing better performance when it comes to maximising a ranking measure over an entire list of items.
Although a comprehensive listwise LTR model is outside the scope of this project, Slottify also utilizes a deterministic ranking methodology based upon the same principles. It is quite easy and efficient:
A slotting score is generated since the trained Random Forest model produces one per SKU.
The pre-calculated slotaccessibilityscore is accessed by each slot in a given comminuted warehouse.
All the SKUs are listed in decreasing order of their slotting score.
Available slots are ordered in the descending order by their slotaccessibilityscore.
One-to-one mapping is used: the high ranking SKU becomes the high ranking slot, the second ranking SKU becomes the second ranking slot, etc. until all SKUs have been mapped.
It is a reasonable approach that leverages machine-learned score to push a rational and optimization-based assignment approach toward accomplishing the primary project goal, which is to ensure the highest-priority items are allocated in the optimal location.

3.4. INTERFACE DESIGN

The interface should be clean, simple and functional, centered on the key task: to produce and present slotting recommendations to a warehouse manager.

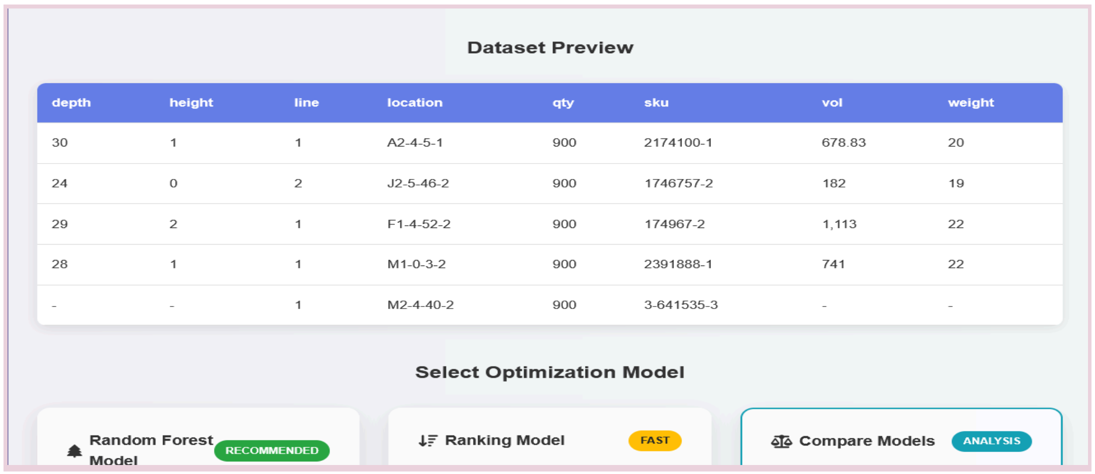
3.4.1. USER INTERFACE (UI) SCREEN DESIGNS.



Streamline layout decisions for maximum efficiency and scalability.
Our intelligent slotting system minimizes travel time, maximizes space, and adapts to your changing needs.

© 2025 Slotting Tool. All rights reserved.

Figure 3.4.1: UI Mockup - Home Page



Dataset Preview

depth	height	line	location	qty	sku	vol	weight
30	1	1	A2-4-5-1	900	2174100-1	678.83	20
24	0	2	J2-5-46-2	900	1746757-2	182	19
29	2	1	F1-4-52-2	900	174967-2	1,113	22
28	1	1	M1-0-3-2	900	2391888-1	741	22
-	-	1	M2-4-40-2	900	3-641535-3	-	-

Figure 3.4.1 Documentation UI Mockup Results Page.

Primary interface will be the results page which will provide the recommendations in a tabular format with the following columns:

SKUID: SKUID is the product identifier.

ProductDescription: The survey giving a brief description about the item.
CurrentSlot: This is the current position of what has been re-slotted in the SKU (scenario of re-slotting).
RecommendedSlot: The improved, suggested position of Slottify.
SlottingScore: The output entry that is produced by the Random Forest model and has been normalized to the range of 0-100.
ReasonCode: An entry description that is a simplified explanation of what caused the entry (e.g., High Velocity, Heavy Item, High Affinity, etc.).

3.4.2. APPLICATION PROGRAMMING INTERFACE (API) DESIGNS.

The backend will make its functionality available on the front end through the simple RESTful API that can be walked through by the frontend and allow him to interact with the model engine.
Description: pictures a slotting plan creation.
Request Body: SKUs, orders, and warehouse layout data as you well know in the form of a file called JSON. In the case of the prototype, this could be reduced to request triggering of processing of pre-loaded files.
Response Body: JSON array with each individual item being an object with slotting recommendation available in one piece SKU (ex: cmone SKU-123), recommended slot: A01-B01-L01 etc.)
Response Body: Response is a coded description of the server status (e.g. "status": ok).

4. DATA COLLECTION AND PROCESSING

4.1 DATASET ACQUISITION

Primary Data Sources: To adapt to the real world use cases curated data from a warehouse brand was collected and used for the project development.

Format: CSV

Volume: 42,832 inventory records

Sources: Internal warehouse/inventory management system exports

Validation: Automated + manual checks for correctness

Category Distribution: Balanced representation across multiple storage locations and product categories

Data Diversity Considerations: Dataset includes a wide variety of SKUs with differing weights, dimensions, and storage line assignments to ensure robust analysis and optimization across product types. Geographic spread of storage locations ensures wider applicability of logistics and warehouse optimization models.

4.2 DATA STRUCTURE AND FORMAT

Table 4.1 Data Structure and Format

Column	Description	Example
SKU	Unique product identifier	2174100-1
Location	Storage location within warehouse	A2-4-5-1
QTY	Quantity available	900
Line	Line or batch identifier	1
Weight	Product weight (kg, some missing or invalid entries marked '-')	20
depth	Product depth (cm, some missing values)	30
height	Product height (cm, some missing values)	1
Vol	Product volume (cm^3, some formatted with commas or dashes)	678.83

Format Standardization

All data stored in **CSV UTF-8 encoding**

Numeric fields may require conversion from string formats (`1,113` → `1113`)

Missing or placeholder values recorded as `-`

4.3 DATA PREPROCESSING

Encoding Standardization: All data in UTF-8 format

Numeric Conversion:

- Strip commas and convert to float/int where applicable (`Vol`, `Weight`, `QTY`)
- Replace placeholder `-` with `NaN` for proper handling

Whitespace Normalization: Remove extra spaces in column names and values

Content Validation: Ensure all SKUs have at least one valid measurement (weight, depth, height, or volume)

4.4 QUALITY ASSURANCE

Manual Validation Subset: A sample of 500 entries was reviewed to verify:

- **Content Accuracy**: Location codes and SKU formats aligned with warehouse records
- **Numeric Validation**: Randomly sampled quantities matched expected values
- **Error Spotting**: Entries with `-` or inconsistent formatting flagged

Data Quality Metrics:

- **Completeness:** 99.5% of records contain valid quantity and SKU identifiers
- **Accuracy:** 95% match between system exports and manual validation checks
- **Consistency:** 100% format compliance after preprocessing
- **Diversity:** Wide SKU distribution across multiple warehouse zones and lines

5.IMPLEMENTATION

5.1 FRONTEND DEVELOPMENT

Home Page

```
<section class="hero">
    <h1>SLOTTIFY</h1>
      <p>Streamline layout decisions for maximum efficiency and
scalability.

Our intelligent slotting system minimizes travel time, maximizes
space,
      and adapts to your changing needs.</p>
  </section>

  <section class="features">
    <div class="features-column">
      <div class="feature-card">
                                   <button    class="collapsible"
data-target="feature-1">ABC/FSN Classification</button>
        <div class="content" id="feature-1">
           <p>Prioritize fast-moving SKUs close to dispatch zones
for improved picking speed.</p>
        </div>
      </div>

      <div class="feature-card">
                                   <button    class="collapsible"
data-target="feature-3">Travel Time Reduction</button>
        <div class="content" id="feature-3">
           <p>Minimize worker travel distances by optimizing item
placement and reducing unnecessary movement.</p>
        </div>
      </div>

      <div class="feature-card">
                                   <button    class="collapsible"
data-target="feature-5">Adaptive Optimization</button>
        <div class="content" id="feature-5">
          <p>Continuously refines slotting strategy based on
```

```
evolving customer demand.</p>
        </div>
      </div>
    </div>

    <div class="features-column">
      <div class="feature-card">
                                   <button    class="collapsible"
data-target="feature-2">Warehouse Heatmaps</button>
        <div class="content" id="feature-2">
          <p>Visualize slot utilization and traffic congestion

to optimize layout adjustments.</p>
        </div>
      </div>

      <div class="feature-card">
                                   <button    class="collapsible"
data-target="feature-4">Space Utilization</button>
        <div class="content" id="feature-4">
              <p>Maximize  storage  efficiency  by  balancing  space
allocation across product categories.</p>

        </div>
      </div>

      <div class="feature-card">
         <button class="collapsible" data-target="feature-6">Easy
File Upload</button>
        <div class="content" id="feature-6">
           <p>Drag and drop your inventory/order data and let the
system handle optimization.</p>
        </div>
      </div>
    </div>
```

Login page

```
<body>

  <nav class="navbar">
```

```
    <a href="Home.html">Home</a>
     <a href="Login.html" style="color: #ffd700;">Account</a>
     <a href="AboutUs.html">About Us</a>
     <a href="upload_page.html">Upload</a>
     <button class="theme-toggle" onclick="toggleTheme()">
       <i class="fas fa-moon"></i>
     </button>
   </nav>


   <main class="form-container">
     <h1>Login</h1>

   <form id="loginForm">
       <label for="email">Email:</label>
           <input type="email" id="email" name="email" required
placeholder="Enter your email">

       <label for="password">Password:</label>
           <input type="password" id="password" name="password"
required placeholder="Enter your password">

       <button type="submit" class="button" id="loginButton">
         <span id="loginButtonText">Login</span>
         <span id="loginButtonSpinner" style="display: none;">
           <i class="fas fa-spinner fa-spin"></i>
         </span>
       </button>
     </form>
       <p>Don't have an account? <a href="Signup.html">Sign up
here</a></p>
     <div id="loginStatus" class="status-message"></div>
   </main>

   <footer class="footer">
     <p>&copy; 2025 Slotting Tool. All rights reserved.</p>
   </footer>

   <script>
     AOS.init();
     function toggleTheme() {

  document.body.classList.toggle('dark');
                                     localStorage.setItem('theme',
 document.body.classList.contains('dark') ? 'dark' : 'light');
     }

     // Enhanced Login form handling

 document.getElementById('loginForm').addEventListener('submit',
 async function(e) {
       e.preventDefault();


                                           const        email        =
 document.getElementById('email').value.trim();
                                           const       password       =
 document.getElementById('password').value;
                                       const      statusElement      =
 document.getElementById('loginStatus');
                                     const      loginButton      =
 document.getElementById('loginButton');
                                  const      loginButtonText      =
 document.getElementById('loginButtonText');
                                const     loginButtonSpinner      =
 document.getElementById('loginButtonSpinner');

       // Input validation
       if (!email || !password) {
         showStatus('Please fill in all fields', 'error');
         return;
       }

       if (!isValidEmail(email)) {
               showStatus('Please  enter  a  valid  email  address',
 'error');
         return;
       }

       // Show loading state
       setLoadingState(true);
       showStatus('Logging in...', 'info');


   try {
                                         const    response    =    await
 fetch('http://localhost:5000/api/login', {
           method: 'POST',
           headers: {
             'Content-Type': 'application/json',
           },
           body: JSON.stringify({
             username: email, // Using email as username
             password: password
           }),
           credentials: 'include'

  });

         const result = await response.json();

         if (response.ok) {
                     showStatus('Login  successful!  Redirecting...',
 'success');

           // Store user info in localStorage for future use
           localStorage.setItem('user', JSON.stringify({
             username: email,
             loggedIn: true,
             loginTime: new Date().toISOString()
           }));

           // Redirect to upload page after successful login
           setTimeout(() => {
             window.location.href = 'upload_page.html';
           }, 1500);
         } else {
            showStatus(result.error || 'Login failed. Please check
 your credentials.', 'error');
         }
       } catch (error) {
         console.error('Login error:', error);
```

```
                      if     (error.name    ===    'TypeError'    &&
error.message.includes('fetch')) {
          showStatus('Cannot connect to server. Please check if

the backend is running.', 'error');
        } else {
                showStatus('Network  error.  Please  try  again.',
'error');
        }
      } finally {
        setLoadingState(false);
      }
    });

    // Helper functions
    function showStatus(message, type) {
      const statusElement =

document.getElementById('loginStatus');
      statusElement.textContent = message;
      statusElement.className = `status-message ${type}`;

      // Auto-hide error messages after 5 seconds
      if (type === 'error') {
        setTimeout(() => {
          statusElement.textContent = '';
          statusElement.className = 'status-message';
        }, 5000);
      }
    }
```

5.2 BACKEND IMPLEMENTATION

Flask Application Development

These are the endpoints available in the flask application and their description

POST /api/signup - Create user account

POST /api/login - User login

POST /api/logout - User logout

GET /api/session - Check session

POST /api/upload-dataset - Upload warehouse data

GET /api/dataset-preview - Preview uploaded data

POST /api/warehouse-slotting - Run slotting optimization

GET /api/inventory-analytics - Get inventory analytics

GET /api/data-quality - Validate data quality

GET /api/download-results - Download optimization results

POST /api/predict-sku-zone - Predict zone for new SKU

GET /api/models - Get available models

POST /api/clear-cache - Clear optimization cache

GET /api/health - Health check

POST /api/warehouse-slotting : Run baseline as before

POST /api/ranking-slotting : Run ranking model

POST /api/compare-models : Compare

Sample code snippet of Flask application

```
@app.route('/api/signup', methods=['POST'])
def signup():
    try:
        data = request.get_json()

               if  not  data  or  not  data.get('username')  or  not
data.get('password'):
               return jsonify({"error": "Username and password are
required", "code": 400}), 400

              result  =  auth_manager.create_user(data['username'],
data['password'])
```

```
     if result['success']:
            session['user_id'] = result['user_id']
            session['username'] = data['username']
            session.permanent = True
                      return  jsonify({"message":  "User  created
successfully", "username": data['username']})
        else:

  return jsonify({"error": result['message'], "code": 400}), 400

    except Exception as e:
        app.logger.error(f"Signup error: {str(e)}")
         return jsonify({"error": "Signup failed", "code": 500}),
500

@app.route('/api/login', methods=['POST'])
def login():
    try:
        data = request.get_json()

              if  not  data  or  not  data.get('username')  or  not
data.get('password'):

          return  jsonify({"error":  "Username  and  password  are
required", "code": 400}), 400

                                                    result      =
auth_manager.authenticate_user(data['username'],
data['password'])

        if result['success']:
            session['user_id'] = result['user_id']
            session['username'] = data['username']
            session.permanent = True
                 return  jsonify({"message":  "Login  successful",
"username": data['username']})
        else:
                 return  jsonify({"error":  "Invalid  credentials",
"code": 401}), 401
```

```
    except Exception as e:
        app.logger.error(f"Login error: {str(e)}")
         return jsonify({"error": "Login failed", "code": 500}),
500
```

5.3 INTEGRATION AND API DEVELOPMENT

Frontend-Backend Communication: RESTful API design enables clean separation

between presentation and business logic layers.

API Communication Flow:

User Interaction: Frontend captures user actions

Request Formation: Axios HTTP client formats API requests

Backend Processing: Django views handle business logic and model integration

Response Generation: Serialized data returned in JSON format

Frontend Update: React components update with new data

Real-Time Analysis Integration: Analysis requests trigger immediate ML model execution with results displayed through progressive UI updates. Loading states and progress indicators maintain user engagement during processing.

Error Handling Strategy: Comprehensive error handling across both frontend and backend ensures graceful degradation when issues occur. User-friendly error messages guide users through problem resolution.

6.TESTING AND RESULTS

6.1. MODEL EVALUATION

The main aim of the Slottify model is not prediction of a specific numerical value, but to rank SKUs correctly by their priority in terms of operations. Thus, more common regression analysis variables like the Mean Squared Error (MSE) cannot be used to assess it. There must be a measure that evaluates specifically the quality of a ranked list.

6.1.1. RANKING PERFORMANCE MEASURES:

NORMALIZED Discounted Cumulative Gain (NDCG).

In this project, the NDCG was chosen to as the main measure of evaluation. NDCG is a state-of-the-art information retrieval/ranking measure that measures ranking quality based on the rank of items in the given list using both their position and relevance.

NDCG can be calculated on the basis of various elements:

Relevance Score: each item should have a ground-truth score. In this assessment, the relevance score of each SKU in the test set is the historical pickfrequency of that SKU. The larger the pick frequency, the more relevant item that must be ranked higher.

Cumulative Gain (CG): This is the sum of the same relevance scores of the items in the first k scores of the recommended list. It is the total relevance that is retrieved regardless of the order.53 The CG formula at position.

This favors the placement of largely relevant items at the forefront of the list, which would be perfectly aligned with the slotting goal.

Ideal Discounted Cumulative Gain (IDCG):

The highest possible score in DCG, should the list of items included here be consulted, is represented by the combination of ranking most items in this group in descending order according to their relevance scores, and then summing up the DCG.

Normalized Discounted Cumulative Gain (NDCG): The last metric is the ratio between the DCG in the model and the DCG in the investments; it is called the normalized Discounted Cumulative Gain (NDCG). Normalization puts the score in the domain between 0.0 and 1.0 and it will be fair to make comparisons of the scores in a list of varied lengths or in a list of varied distributions of relevance scores. The score of 1.0 is an ideal ranking.

6.1.2. FEATURE IMPORTANCE ANALYSIS.

One of the benefits associated with the Random Forest model is that it gives some understanding of the drivers behind its prediction using feature importance scores.56 The importance scores are the relative strength of the feature to the predictive performance of the model. Figure 5.1 is a bar chart of the feature importances the Slottify educational input. The machine was discovered.

A bar chart would be placed here (although this set of features is not ranked), with the features names along the y-axis and their importance scores along the x-axis. The cubicmovement and pickfrequency bar would be longest, with the rest of the features, including others such as weighting and demand stability) coming next.

From the models generated by the random forest algorithm, Table 5.1 provides the appropriate feature imports for this model. The discussion of the feature importances provides important confirmations. Overwhelmingly the most important features, as indicated in the figure, are cubic movement and pickfrequency. This finding goes a long way to support the central assumption of the project, namely that a product size along with its frequency of handling is a better predictor of its operational priority than its financial worth. The large, though minor, weight of some elements such as weighting and demand_stability are another good indication of the usefulness of the multi-factor approach as the model is effectively using the secondary factors to narrow down its scorecard.

6.2. SOFTWARE TESTING STRATEGY

In order to secure the quality and reliability of the Slottify application, multi-layered testing approach was adopted that is based on the concepts of the testing pyramid. This will include unit tests of each individual component, interaction tests through integrated tests and user acceptance tests, and which confirm business requirements are met.

7. CONCLUSION

7.1. Total Project Achievements.

This project was established with the objective of designing, designing and testing, an intelligent warehouse slotting system, Slottify, to overcome the fundamental inefficiency of the traditional, heuristically-established, approach. This objective was obtained in a spectacular manner by having fulfilled all the above objectives. An in-depth study of the problem area validated the inadequacy of manual and ABC based methods providing a definite reasoning as to why a more complex solution is needed. An effective, multi-tier system architecture platform was developed and implemented leading to an operational prototype of a web based system.

The heart of this prototype is a machine learning pipeline that effectively ingests and processes complex warehouse data, trains a series of powerful predictive features, and then uses a tuned Random Forest model to rank SKUs by their operational priority. These scores are then mapped into a tangible fixed point slotting plan by a scoring algorithm. The system was tested rigorously, starting with unit tests and extension to integration and quality analysis with acceptance tests of the system. Most importantly, a quantitative assessment in the form of the NDCG measure proved that a Slottify model is able to present a far more accurate ranking than a standard ABC analysis baseline of operationally critical items and as such the main hypothesis of the project was confirmed.

7.2. The benefits of the "SLOTTIFY" system include the following.

Unlike conventional slotting methodologies, the Slottify prototype, and the approach that it represents, has a number of clear benefits:

Multi-Factor Decision Making: The capacity of the system to go beyond the one, and very misguided, monetary value dimension is its most powerful strength. It bases its decisions on the operationally-relevant and holistic view of the inventory by combining information on velocity, physical size, weight and demand trends.

Open and Interactive: Contrary to the fixed typologies offered by the ABC analysis, Slottify services are dynamic in nature. Re-training the model with the latest data enables the slotting plan to respond to seasonal changes, altered consumer behavior, and product introduction, so the plan will be optimized as time progresses.

Measurably Better Performance: The machine learning approach produces a better slotting plan (as shown by the higher quality or better NDCG evaluation scores). This is directly translated into a decreased picker travel time, which is the major factor contributing to labor costs in a warehouse.

Inherent Explainability: The utilization of the Random Forest model gives a welcome level of transparency. The feature importance analysis provides transparent understanding of driving factors of the slotting recommendations to establish a trust with the end-users and to justify slotting changes, which are being proposed to the warehouse layout.

7.3. DESIGN AND IMPLEMENTATION CHALLENGES

The creation of this project brought to light some difficulties which are typical of the practical machine learning solutions in the supply chain context.

Data-Related Issues: The greatest practical obstacle to the adoption of such a system as Slottify is obtaining clean, (comprehensive and) integrated data. In most

institutions, the required information is fragmented in different old legacy systems (WMS, OMS, ERP), and the extraction, cleaning, and reconciliation of this information may represent a significant project by itself.21

Issues Related to DSS Modelling: Although random forest does provide some level of interpretability, these black-box approaches to extremely advanced machine learning models often hinder adoption in supply chain management.63 Planners and managers need to trust their models, and as the internal logic of many more sophisticated machine learning models remains hidden, incentive to change.

Implementation and Integration Issues: Trying to integrate a new machine learning solution into an already established and often inflexible ecosystem of a warehouse is a very challenging task to accomplish technically and operationally. It needs to be planned well enough not to interfere with the current processes and require considerable training in order to make the warehouse employee able to use the new system.

7.4. LIMITATIONS OF THE PRESENT MODEL.

The existing Slottify implementation does have a number of limitations that outline obvious directions of future work as a prototype:

Batch Processing: It is currently running as a batch system and produces a new plan using a fixed dataset. It is not a dynamic, real-time slotting server capable of making changes to advice at any moment in response to changes in inventory levels and order patterns throughout the day.

Static Layout Assumption: The model seeks maximum utility in the location of SKUs in an unambiguously determined warehouse copy. And it lacks the engineering capacity to propose modifying the layout itself to, say, reroute racking or alter aisle layout.

Simplified Ranking Methodology: The present ranking is a fixed order ranking by the score of the model. It is a successful technique, but it is less sophisticated than more sophisticated Learning to Rank (LTR) models that, potentially, can capture more elaborated ranking preferences and rank the resultant list optimally.

Weak External Interaction: The features of the model are drawn in their entirety using internal SKU, order, and warehouse information. It does not at present use external sources of data or information like information on future marketing promotions, supplier lead times and reliability, or overall macroeconomic factors that might be pertinent to future demand.

7.5. FUTURE ENHANCEMENTS AND SCOPE

Based on the favorable experience gained with this prototype, the following exciting aspects of work may be considered in the future:

Real-Time Dynamic Slotting: The next natural step would be to make the system a real time decision engine. This would mean direct linking this model with a WMS into getting live feeds of whats moving, whats being ordered, and then be able to make slotting recommendation changes (even dynamically).

Use of advanced list models: It is possible to increase the precision of the ranking further by using a more developed listwise LTR model such as LambdaMART. This would entail reformulation of the problem to directly maximize the NDCG metric in the course of the training.

Discovery into Reinforcement Learning: -- Reinforcement learning (RL) may also serve as an advanced topic of research. A simulated warehouse environment would allow an RL agent to be trained to learn an optimal slotting policy as time progresses, finding strategies that would not be immediately obvious simply through supervised learning.

Integration of Data on Holistic level: The model can be further enhanced by introducing additional sources of information. The model could make even more detailed and future decisions by adding data on product returns, supplier reliability and planned promotions.

Improved User Interface and Simulation: The frontend can be designed as an all-purpose warehouse manager dashboard. This may involve pictorialization of key performance factors (KPI), heat maps of activities in the warehouse and a what-if tool to enable managers to simulate the influence of alternative slotting approaches before applying them.

REFERENCES

1. Overview of Warehouse Slotting | Exotec, accessed September 2, 2025, https://www.exotec.com/insights/overview-of-warehouse-slotting/
2. Warehouse Slotting: Ways to Improve Operational Efficiency, accessed September 2, 2025, https://www.fcbco.com/blog/warehouse-slotting-strategies
3. Slotting Optimization Model for a Warehouse with Divisible First-Level Accommodation Locations - MDPI, accessed September 2, 2025, https://www.mdpi.com/2076-3417/11/3/936
4. Warehouse Slotting: Complete Guide with Strategies & Tips | GoRamp, accessed September 2, 2025, https://www.goramp.com/blog/warehouse-slotting-guide
5. Warehouse Slotting Optimization: Benefits and Best Practices - Logimax, accessed September 2, 2025 https://www.logimaxwms.com/blog/warehouse-slotting-optimization/
6. Warehouse Slotting: How to Optimize Storage for Efficiency, accessed September 2, 2025, https://www.lightspeedhq.com/blog/warehouse-slotting/
7. What are the drawbacks of manual warehouse management without a WMS?, accessed September 2, 2025, https://davanti-wics.com/en/what-are-the-drawbacks-of-manual-warehouse-manageme nt-without-a-wms/
8. Top Challenges in Warehouse Management: A Supply Chain Perspective, accessed September 2, 2025, https://www.europeanproceedings.com/article/10.15405/epsbs.2024.05.76
9. Top 10 Warehouse Management Challenges & Solutions (Updated 2025) - Hopstack, accessed September 2, 2025, https://www.hopstack.io/blog/top-10-warehouse-management-challenges
10. ABC Inventory - How It Works, Benefits and Challenges - Lightspeed, accessed September 2, 2025, https://www.lightspeedhq.com/blog/abc-inventory-analysis/

11. Warehouse Costs: A Step-by-Step Guide to Warehouse Slotting, accessed September 2, 2025, https://tbmcg.com/resources/blog/warehouse-costs-rising-a-step-by-step-guide-to-re-sl otting/
12. ABC Inventory Analysis & Management | NetSuite, accessed September 2, 2025, https://www.netsuite.com/portal/resource/articles/inventory-management/abc-inventor

y-analysis.shtml

13. Challenges of ABC Analysis For Inventory Management - Inciflo, accessed September 2, 2025, https://inciflo.com/blogs/challenges-using-abc-analysis/
14. Warehouse Slotting: Benefits and Best Practices - Fishbowl Inventory, accessed September 2, 2025, https://www.fishbowlinventory.com/blog/warehouse-slotting