**PROJECT BASED LEARNING**

**ON**

**CARGO LOADING**

**DESIGN AND ANALYSIS OF ALGORITHMS - BCS-DS-501**

***Submitted by:***

**REVANTH SAI BOMMA 1/23/SET/BCS/417**

**AKSHITA CHAUHAN 1/23/SET/BCS/465**

**BHUMIKA RAWAT 1/23/SET/BCS/408**

**UTSAV 1/23/SET/BCS/427**

***Under the Guidance of***

**Ms. Deepika Kataria**

**(Assistant Professor)**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF TECHNOLOGY**

**IN**

**Computer Science & Engineering**



**Faculty of Engineering & Technology**

**MANAV RACHNA INTERNATIONAL INSTITUTE OF RESEARCH AND STUDIES, Faridabad**

**NAAC ACCREDITED ‘A’ GRADE**

**Aug – Dec, 2025**

**TABLE OF CONTENTS**

**1. Abstract**

**2. Introduction**

**3. Problem statement**

**4. Objectives**

**5. Methodology**

**6. Results**

**7. Conclusion**

**8. Future scope**

**9. Reference**

**ABTRACT**

This project presents the implementation of the Fractional Knapsack Algorithm to optimize cargo loading in logistics using Python. The objective is to maximize profit while ensuring that the total weight of loaded items does not exceed the truck’s capacity. In real-world logistics, many goods such as rice, wheat, sugar, oil, and flour can be divided into smaller portions, allowing for partial loading. This characteristic makes the Fractional Knapsack (Greedy Algorithm) an ideal solution.

The project reads input data from a CSV file containing item names, weights, and profits. It then calculates each item’s profit-to-weight ratio, sorts items in descending order of ratio, and loads items fully or fractionally until the capacity is reached. The final output displays the maximum profit achievable, the load plan (showing selected and fractional items), and a bar chart visualization comparing selected versus skipped cargo.

This implementation demonstrates how algorithmic optimization can be applied to real-world logistics problems, providing a fast, accurate, and scalable solution for maximizing cargo profit while maintaining computational efficiency. The project also highlights how classical algorithms can be effectively integrated with modern visualization tools (Matplotlib) to improve interpretability and decision-making in logistics systems.

**INTRODUCTION**

In today’s competitive logistics and transportation industries, efficient resource utilization is critical to achieving maximum profitability and operational effectiveness. One of the most common challenges faced by logistics managers is determining which goods to load and in what quantity, especially when the available cargo space or truck capacity is limited. Making this decision manually can often lead to suboptimal outcomes — either underutilizing capacity or overloading low-profit goods.

This challenge can be modeled mathematically as a Knapsack Problem, a well-known optimization problem in computer science and operations research. The knapsack problem involves selecting a combination of items, each with a weight and a profit, such that the total weight does not exceed a specified limit while maximizing total profit. However, in real-world logistics, certain bulk goods such as rice, wheat, oil, and sugar can be divided into smaller quantities, allowing partial loading. This variation is known as the Fractional Knapsack Problem, where fractions of items can be included based on their profit-to-weight ratio.

The Greedy Algorithm provides an efficient solution for the fractional knapsack problem. It prioritizes items based on their profit density (profit per unit weight), ensuring that the most valuable items, or fractions of them, are loaded first until the available capacity is fully utilized. This approach ensures an optimal and computationally efficient solution compared to brute-force or dynamic programming methods.

In this project, the Fractional Knapsack algorithm has been implemented to optimize cargo loading decisions. The implementation uses Python, with Streamlit providing an interactive web interface that allows users to upload datasets, visualize results, and dynamically test different loading scenarios. The project demonstrates how classical algorithmic techniques can be applied to practical logistics and transportation optimization tasks. Furthermore, the web deployment on Streamlit Cloud via GitHub makes the solution easily accessible for educational demonstrations, research use, and real-world logistics analysis.

**PROBLEM STATEMENT**

In logistics and supply chain management, one of the most critical operational challenges is optimizing the loading of cargo within vehicles such as trucks, ships, or airplanes that have a limited carrying capacity. The goal is to determine which goods should be loaded—and in what quantities—in order to maximize total profit or value, without exceeding the available weight limit.

Traditional approaches to cargo loading often involve manual estimation or fixed-priority loading, which can lead to inefficient utilization of capacity and reduced profitability. Furthermore, real-world scenarios often allow for the partial loading of goods. For example, a truck may load 20 kilograms of rice out of a 50-kilogram stock if that yields a higher overall return. This requirement makes the problem ideal for the Fractional Knapsack Model, where items can be divided into smaller portions based on their profit-to-weight ratio.

The central problem, therefore, is to develop a computational model and software system that:

1. Accepts inputs in the form of items with associated weights and profits.
2. Calculates the optimal load plan (including fractional quantities) that maximizes total profit.
3. Ensures that the total weight of selected items does not exceed the truck’s capacity.
4. Provides a clear visual and tabular representation of which items are selected, skipped, or partially included.

This project aims to solve the cargo optimization problem using the Greedy approach of the Fractional Knapsack algorithm, implemented in Python and deployed as an interactive Streamlit web application. The app allows users to upload real or simulated datasets, automatically perform profit maximization calculations, and visualize outcomes through intuitive charts.

Ultimately, this solution provides an effective demonstration of how algorithmic problem-solving and data visualization can be leveraged to improve decision-making in real-world logistics and transportation operations.

**OBJECTIVES**

The primary objective of this project is to apply the Fractional Knapsack algorithm to solve real-world optimization problems in logistics, focusing on maximizing profit while respecting cargo weight limitations.

To achieve this, the project is guided by the following specific objectives:

1. To develop a computational model that implements the Fractional Knapsack algorithm using the Greedy approach to determine the most profitable combination of items that can be loaded within a given capacity.
2. To simulate real-world cargo loading conditions using a dataset of items (e.g., rice, wheat, oil, sugar, flour) with defined weights and profits, representing practical logistics operations.
3. To design an interactive web application using Streamlit, enabling users to:
   * Upload or select datasets,
   * Adjust parameters such as capacity,
   * View item selection results in tabular and graphical formats.
4. To visualize the optimization process through bar charts that distinguish between selected and skipped cargo items, thereby making the algorithm’s behavior easy to interpret.
5. To deploy the project on Streamlit Cloud via GitHub, making it publicly accessible for academic, research, and industrial demonstration purposes.
6. To enhance understanding of the Greedy Algorithm, showcasing how it provides an optimal and efficient solution for the fractional knapsack problem compared to other methods.
7. To lay the foundation for future advancements, such as integrating machine learning for demand prediction or extending the model to multi-truck and multi-constraint optimization problems.

**METHODOLOGY**

**Step 1: Dataset Preparation**

A CSV file (knapsack\_5\_items\_new.csv) was used, containing:

* Item weights (W1–W5)
* Item profits (P1–P5)
* Knapsack capacity
* Best price (for reference)

| **Item** | **Weight** | **Profit** |
| --- | --- | --- |
| Rice | 50 | 100 |
| Wheat | 20 | 60 |
| Oil | 30 | 120 |
| Sugar | 40 | 80 |
| Flour | 10 | 30 |

**Step 2: Algorithm Used**

Fractional Knapsack Algorithm (Greedy Approach)

**Why Fractional Knapsack is Used Over Other Types of Knapsack**

The Knapsack Problem exists in several forms—namely the 0/1 Knapsack, Bounded Knapsack, and Fractional Knapsack. The selection of the appropriate variant depends on the type of items, problem constraints, and computational requirements.

In this project, the Fractional Knapsack algorithm has been chosen because it best reflects real-world cargo loading conditions, where goods such as rice, wheat, sugar, and oil can be divided into smaller quantities. The following points justify this selection over other types of knapsack problems.

**1. Nature of Items (Divisible vs. Indivisible)**

* **0/1 Knapsack:** Items are indivisible. Each item can either be taken entirely or not taken atall.  
  *Example:* A laptop or a television cannot be divided and carried partially.
* **Fractional Knapsack (Chosen):**Items are divisible. It is possible to load a fraction of an item if the full weight does not fit within the available capacity.  
  *Example:* A truck can carry 20 kg of rice out of 50 kg available.

Since most cargo goods are divisible, the Fractional Knapsack model is more realistic and practical for logistics applications.

**2. Computational Efficiency**

* **0/1 Knapsack:** Requires Dynamic Programming or Branch and Bound techniques, which have a time complexity of O(n·W) (where *W* is capacity). This approach becomes computationally expensive for large datasets.
* **Fractional Knapsack:** Solved efficiently using a Greedy Algorithm with a time complexity of O(n log n) (for sorting items by profit-to-weight ratio). This makes it faster, simpler, and highly scalable for real-time scenarios.

The Greedy approach ensures optimal results for the fractional case with much lower computational cost.

**3. Real-World Relevance**

In logistics and supply chain management, it is often necessary to partially load goods to make the best use of available capacity. The Fractional Knapsack algorithm models this scenario perfectly by allowing partial selection of items based on their profit-to-weight ratio.

In contrast, the 0/1 Knapsack model restricts loading to full items only, which may leave unused space and reduce overall profitability.

Hence, the Fractional Knapsack provides a realistic and practical representation of actual cargo loading operations.

**4. Optimality and Simplicity**

The Greedy strategy used in the Fractional Knapsack algorithm ensures an optimal solution, as items with the highest profit-to-weight ratio are always chosen first.

The algorithm is simple to implement, easy to visualize, and guarantees that every loaded fraction contributes maximally to profit until capacity is fully utilized.

This simplicity and mathematical optimality make it ideal for an interactive educational application like this Streamlit-based project.

**Step 3: Algorithm Steps**

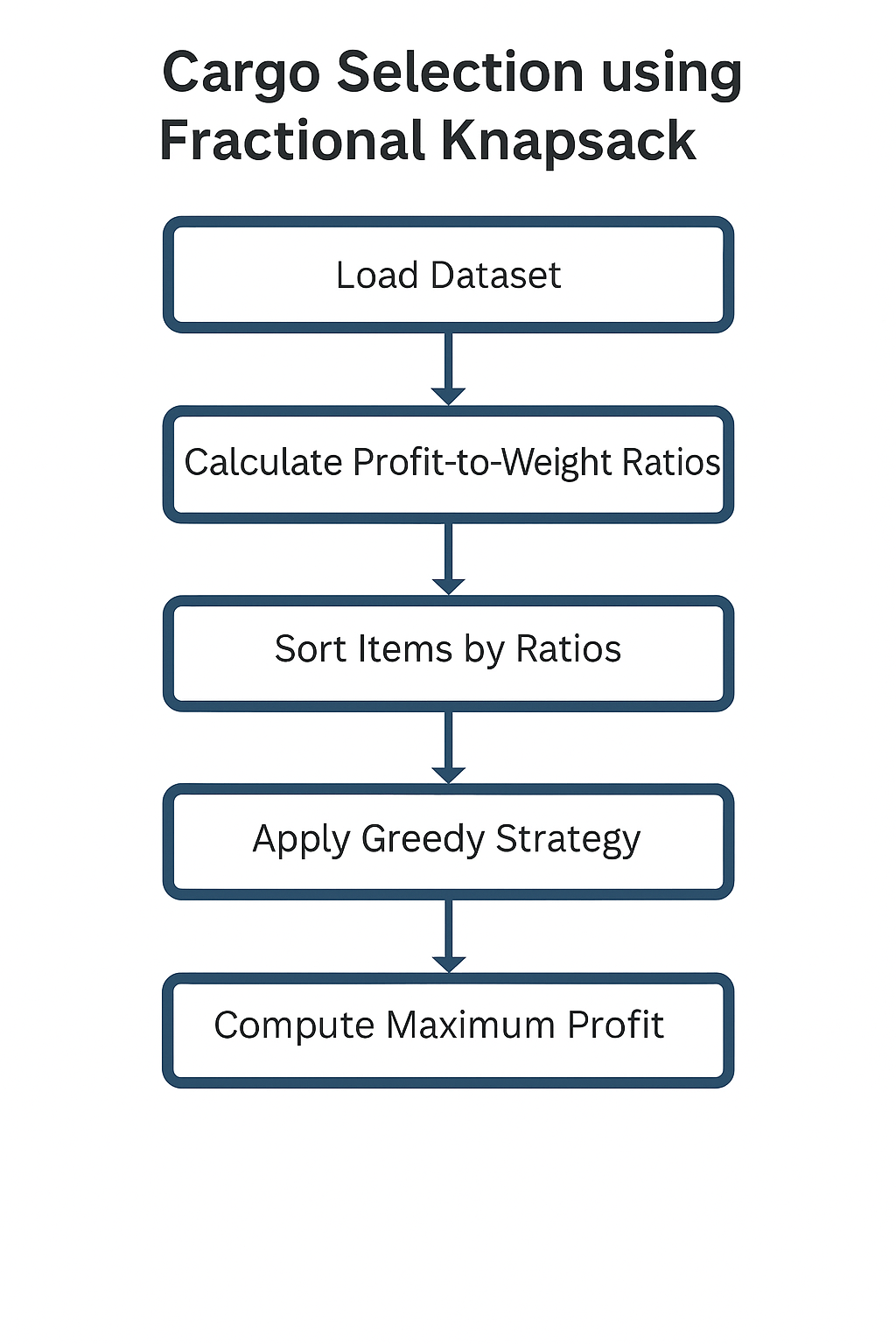
1. Compute Profit-to-Weight Ratio for each item.
2. Sort items in decreasing order of ratio.
3. Pick items until capacity is full:
   * If the next item fits, take it fully.
   * Else, take a fraction of it proportional to remaining capacity.
4. Compute total profit.

**Step 4: Implementation Tools**

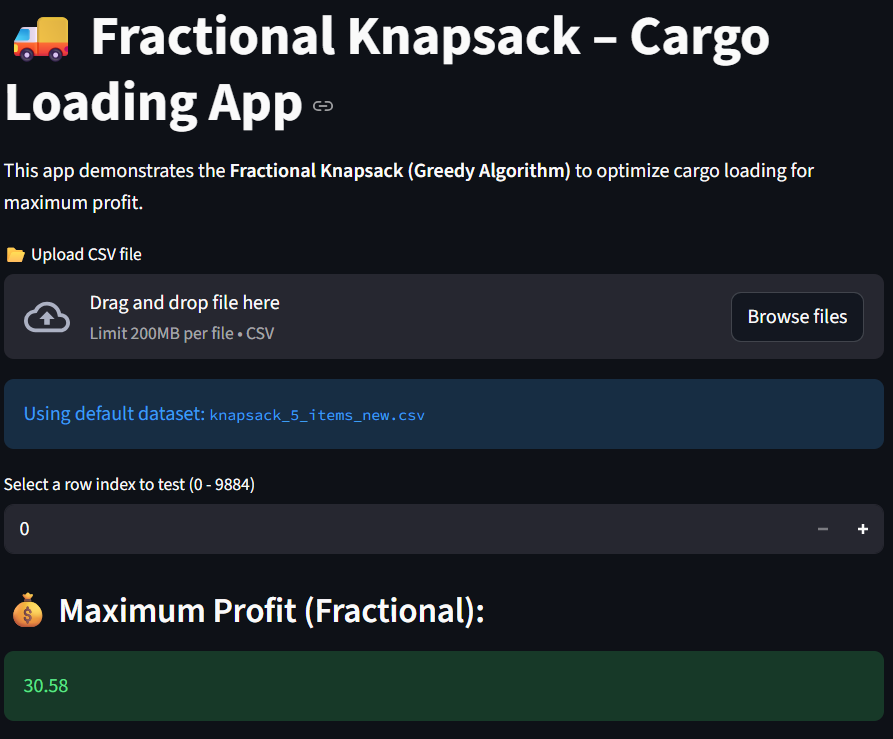
| **Component** | **Tool/Technology** |
| --- | --- |
| Programming Language | Python |
| Visualization | Matplotlib |
| Web Framework | Streamlit |
| Deployment | Streamlit Cloud |
| Dataset | CSV (Pandas DataFrame) |
| Version Control | Git & GitHub |
|  |  |

**Step 5: Application Flow**

1. User uploads or loads the dataset.
2. Selects a row instance (capacity & item weights).
3. Algorithm runs automatically to compute optimal load.
4. App displays:
   * Selected items (with fractions)
   * Maximum profit
   * Visualization (bar chart)



**Results**



### **Overview**

This section of the application interface allows the user to:

* Upload or use a default dataset (knapsack\_5\_items\_new.csv)
* Select a particular row index representing one instance of the cargo loading problem.
* View the maximum profit achieved using the Fractional Knapsack Algorithm.

### **Key Components Explained**

#### 1. Title Section

* **“**Fractional Knapsack – Cargo Loading App”  
  This is the project title displayed prominently at the top of the Streamlit interface.  
  It clearly indicates that the app applies the Greedy Algorithm to solve a Fractional Knapsack optimization problem for cargo loading.

#### 2. File Upload Section

* Allows users to drag and drop a CSV dataset or click “Browse files” to upload a custom dataset.
* If the user doesn’t upload any dataset, the app automatically loads the default dataset (knapsack\_5\_items\_new.csv), as indicated in the blue info box.

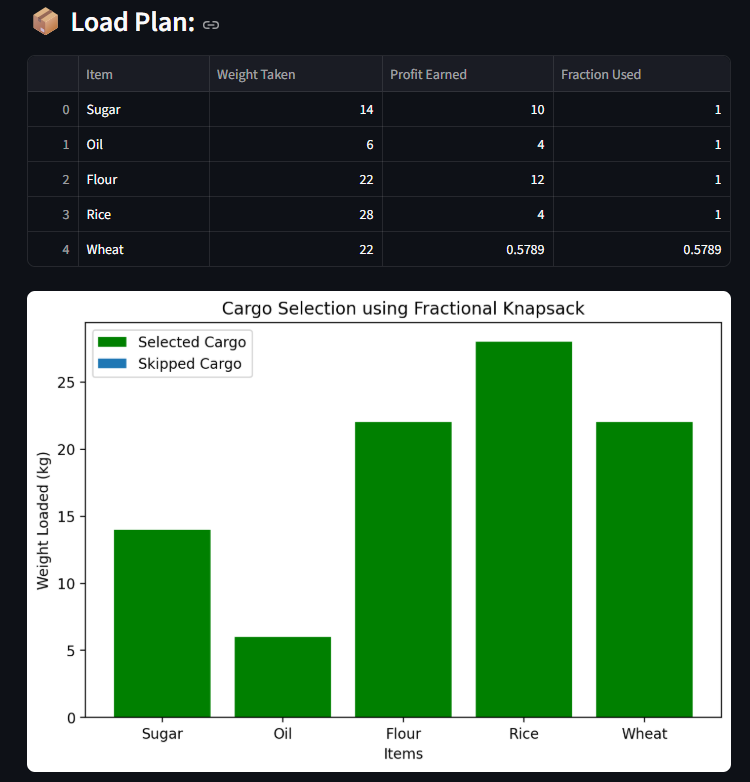
#### 3. Row Selection

* Each row in the dataset corresponds to a different test case (a unique combination of item weights, profits, and capacity).
* The user can choose which row to analyze using a numeric input field.  
  In this output, the selected row index = 0, meaning the algorithm is running on the first record of the dataset.

#### 4. Maximum Profit (Fractional)

* This result displays the total profit obtained by applying the Fractional Knapsack algorithm on the selected data instance.
* The app calculates profit by:
* In the screenshot:

This means the optimal cargo loading plan yields a total profit of 30.58 units while staying within the truck’s capacity limit.



**Overview**

This section provides a detailed breakdown of how the algorithm selected or partially selected each cargo item, along with a visual representation (bar chart) comparing the selected and skipped items.

**1. Load Plan Table**

| **Item** | **Weight Taken** | **Profit Earned** | **Fraction Used** |
| --- | --- | --- | --- |
| **Sugar** | 14 | 10 | 1.0 |
| **Oil** | 6 | 4 | 1.0 |
| **Flour** | 22 | 12 | 1.0 |
| **Rice** | 28 | 4 | 1.0 |
| **Wheat** | 22 | 0.5789 | 0.5789 |

**Explanation**

* Sugar, Oil, Flour, and Rice were taken completely (Fraction = 1) because they fit within the available truck capacity.
* Wheat was taken partially (Fraction = 0.5789) — meaning only 57.89% of its available quantity could fit into the remaining capacity of the truck.
* This partial inclusion ensures that the capacity is fully utilized while maximizing total profit.

The algorithm thus uses fractional inclusion for the last item once the remaining space is insufficient for its full weight.

**2. Cargo Selection Bar Chart**

**Chart Title: “Cargo Selection using Fractional Knapsack”**

* **X-axis:** Displays the item names — Sugar, Oil, Flour, Rice, and Wheat.
* **Y-axis:** Represents the weight loaded (in kilograms) for each item.
* **Green Bars:** Items that were selected (fully or partially) for loading.
* **Red or Blue Bars (if visible):** Represent items that were skipped or not included due to capacity constraints.

In this output, all five items were included (Wheat only partially), so all bars appear green.

**Interpretation**

* The visualization clearly shows how much of each item was loaded.
* The Greedy Algorithm has prioritized items with higher profit-to-weight ratios.
* The truck capacity is completely utilized, resulting in maximum profit without any unused space.
* The user can easily interpret which items contribute the most to profit and how fractional selection optimizes total gain.

**Conclusion**

The Fractional Knapsack – Cargo Loading Optimization project successfully demonstrates how the Greedy Algorithm can be effectively applied to real-world logistics and resource allocation problems. By allowing partial loading of divisible goods such as rice, wheat, oil, sugar, and flour, the algorithm ensures maximum profit while fully utilizing the available truck capacity.

Through this project, it was observed that the Fractional Knapsack algorithm outperforms other variants such as the 0/1 Knapsack in terms of flexibility, computational efficiency, and real-world applicability. The algorithm’s core principle — prioritizing items based on their profit-to-weight ratio — guarantees optimal profit outcomes with minimal computational effort.

The Streamlit-based web application built for this project provides an interactive and user-friendly interface, enabling users to:

* Upload or use default datasets,
* View optimal load plans dynamically, and
* Visualize selected versus skipped cargo using bar charts.

The successful deployment on Streamlit Cloud via GitHub makes the system accessible online, demonstrating how classical algorithms can be integrated into modern web technologies for educational and industrial purposes.

Overall, this project not only strengthens understanding of the Greedy approach in optimization problems but also highlights its direct relevance to logistics management, inventory planning, and profit optimization in real-life scenarios.

**Future Scope**

While the current implementation effectively solves the fractional loading problem for a single truck with predefined items, there is significant potential to extend and enhance the system further. The following are some promising directions for future development:

1. **Multi-Truck Optimization:** Extend the model to handle multiple trucks or transport units, distributing cargo intelligently among them to maximize overall profit.
2. **Integration with Real-Time Data:** Link the application with live logistics or warehouse databases to automatically fetch real-time information about item availability, demand, and profit margins.
3. **Machine Learning-Based Prediction:** Use machine learning models to predict item profitability or demand trends, helping the algorithm make smarter, data-driven loading decisions.
4. **Multi-Constraint Knapsack Model:** Incorporate additional constraints like volume, cost, and delivery time alongside weight, to make the system more applicable to real-world logistics networks.
5. **0/1 and Bounded Knapsack Comparison Module:** Add the ability to compare results from Fractional, 0/1, and Bounded Knapsack models for academic and analytical purposes.
6. **Improved Visualization and Reporting:** Integrate dashboards using tools like Plotly or Power BI to generate richer visual insights, load efficiency graphs, and exportable PDF reports.
7. **Mobile and Cloud API Integration:** Deploy the algorithm as a REST API or mobile application so logistics companies can integrate the optimization service directly into their operational systems.

**REFERENCE**

[1] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). MIT Press.

[2] Kleinberg, J., & Tardos, É. (2005). Algorithm Design. Pearson.

[3] GeeksforGeeks. (2024). “Fractional Knapsack Problem | Greedy Algorithm.” <https://www.geeksforgeeks.org/fractional-knapsack-problem/>

[4] Brilliant.org. (2023). “Knapsack Problems.” <https://brilliant.org/wiki/knapsack-problem/>

[5] Pandas Documentation (2024). “pandas.DataFrame — Data Analysis and Manipulation.” <https://pandas.pydata.org/docs/>

[6] Matplotlib Documentation (2024). “Matplotlib Visualization Library in Python.” <https://matplotlib.org/stable/contents.html>

[7] Streamlit Documentation (2024). “Streamlit: The fastest way to build data apps.” <https://docs.streamlit.io/>

[8] GitHub Guides (2024). “Understanding the GitHub Flow.” <https://guides.github.com/introduction/flow/>

[9] Sharma, S., & Kumar, A. (2020). “Optimization Techniques in Logistics and Supply Chain Management.” International Journal of Engineering Research & Technology (IJERT).

[10] Dantzig, G. B. (1957). “Discrete-Variable Extremum Problems.” Operations Research.

[11]