Georgia Tech College of Engineering
H. Milton Stewart School of
Industrial and Systems Engineering

ISyE 6202 Supply Chain Facilities

Casework 4.1

EasyNode Smart Locker Bank

Guillaume Eymery - Nathalia Gonzalez - Anjulakshmi Karthikeyan - Louise Lacroix

H. Milton Stewart School of Industrial and Systems Engineering

Contents

TASK 1 **6**
Yearly Expected Demand . . . 6
Maximum number of parcels in the room at the same time . . . 8
Maximum number of parcels of each type in the room at the same time . . . 10
Number of different locker sizes . . . 10
Lockers design . . . 11
Number of stacks of each size . . . 13
Overall layout of the locker room . . . 15

TASK 2 **16**
Choosing the right model . . . 16
Implementing the model . . . 16
Implementing and running the code in Python . . . 17

TASK 3 **19**
Step 1: Generate Demand Scenario For The Next Five Years . . . 19
Step 2: Simulating The Daily Demand For Each Year . . . 20
Step 3: Simulating The Hourly Demand For Each Day . . . 23
Step 4: Generating the Deposit and Pickup Time Using Task 2 . . . 26
Step 5: Generating the Package Sizes and integrating a Graphical User Interface 26

TASK 4 **29**
Heuristic Implementations . . . 29
Assignment Methodology . . . 31

TASK 5 and 6 **33**
Heuristic Size X in Locker X Simulation and Analysis . . . 33
Smarter heuristic - Improve Strawman, results and analysis . . . 36
Financial Analysis . . . 37
Multiple Simulation Results . . . 39
Cost Analysis . . . 43

TASK 7 **48**
Phase 1 . . . 48
Phase 2 . . . 49
Phase 3 . . . 50

TASK 8 **54**
Defining the locker sizes 54
Defining the lockers towers sets 55

TASK 9 **57**

TASK 10 **60**
Performance Evaluation 60
Comparison with Previous Assignation Logic 60
Locker Utilization 61
Service Level Analysis 62
Scalability and Robustness 62
Heuristic Validation 63

TASK 11 **64**
Fixed-Configuration Locker Bank 64
Dynamic Modular Configuration 64
Cost Analysis: Modular vs. Fixed Configuration 65
Key Insights and Comparison66
Conclusion 66

TASK 12 **67**
Executive Summary for EasyNode 67
Recommendations 68

TASK 13 **69**

List of Figures

1 Yearly Demand Scenarios with Conditional Inter-Yearly-Scenario Probability . . . 6
2 Probability (X10,000) That a Given Parcel is Deposited In Hour d and Then Picked Up by the Customer in Hour p . . . 8
3 Maximum number of parcels in the locker room over year 5 . . . 9
4 Maximum number of parcels in the locker room over year 5 - sizing for peak pointed by the arrow . . . 9
5 Distribution of the different sizes of parcels at the peak time (Friday, 13th 4th week period, 12:00) . . . 10
6 Total space required for three proposals of total number of different sizes of lockers . . . 11
7 3D Visualization of the 18 different stacks of lockers . . . 12
8 3D design of the locker bank . . . 13
9 First overall layout of the locker bank . . . 15
10 Labels of the different lockers sizes . . . 15
11 Yearly Demand Scenarios with Conditional Inter-Yearly-Scenario Probability 19
12 Monte Carlo Simulation - Different Demand Scenarios . . . 20
13 Demand Share . . . 21
14 Monte Carlo Simulation - Daily Demand - Fixed Yearly Demand . . . 23
15 Monte Carlo Simulation - Daily Demand - Variable Yearly Demand . . . 23
16 GUI using Tkinter library . . . 27
17 5 Year Simulation . . . 27
18 Simulator Results . . . 28
19 Occupation Rate Over Time - Strawman Heuristic - Sept 2027 to Dec 2028 34
20 Occupation Status at 9:16:00 - 2024-01-02 - Locker Group L1 . . . 35
21 Occupation Status at 11:16:00 - 2024-01-02 - Locker Group L1 . . . 35
22 Occupation Rate Over Time - Smarter Heuristic - Sept 2027 to Dec 2028 . 37
23 Instance A . . . 40
24 Instance B . . . 40
25 Instance C . . . 40
26 Instance D . . . 40
27 Locker Assignment Success Rate (5years) . . . 40
28 Number of rejections . . . 40
29 Annual Locker Assignment for Strawman Heuristic . . . 41

30 Annual Locker Assignment for Smarter Heuristic 42
31 Locker Utilization(5years) - STrawman 42
32 Locker Utilization(5years) - Smarter 42
33 Shipping costs for Strawman for the 5 year period 43
34 Shipping costs for Smarter for the 5 year period 44
35 Annual Shipping costs for Smarter Heuristic 44
36 Annual Total cost for Smarter Heuristic 45
37 Total cost (5 years) for Smarter Heuristic 46
38 Additional Locker Requirements 51
39 Evolution of the RR over the iterations 52
40 Evolution of the RR over the iterations L17 and L18 53
41 Total number of packages over the years entering the locker bank 54
42 Percentage of the demand fulfilled by each locker size 55
43 Towers of lockers set 56
44 Maximum number of parcels being held at the same time in the locker bank during each 4-week period 57
45 Number of towers of each type required during each period 58
46 Rental and Maintenance costs 59
47 Transition costs 59
48 Total annual costs 59
49 Modular Configuration with Smarter Heuristic success rate (5 years) 60
50 Locker Utilization - Modular Configuration 61

List of Tables

1 Probability of each scenario 7
2 Expected demand for each year 7
3 Maximum yearly demand for year 5 8
4 Dimensions and stacking details 12
5 Number of packages and stack for each locker type 14
6 Daily Demand for January 2024 22
7 Hourly Distribution of Demand 24
8 Locker Simulation Results for Heuristic 2: Smarter Heuristic - Upgrading Strawman 36
9 Shipping Fare Table by Volume, Fare Range, and Company 39
10 Costs for Fixed Configuration 45

11 Annual Total Costs for Each Instance 46
12 Iterations with Overall Service Level and Rejection Rates 50
13 Initial and Final Locker Counts by Locker Type 51
14 Locker sizes 55
15 Costs 58
16 Comparison of Fixed Configuration vs. Dynamic Modular Locker Systems 65
17 Cost Comparison of Modular Configuration vs. Fixed Configuration 66

TASK 1

In this task, we are to design a first layout of the locker bank based on demand predictions. All computations have been done in the spreadsheet *case4_task01*

Yearly Expected Demand

According to figure 1, the demand has various probable scenarios. Using the law of total probabilities, we will compute the Probability of each scenario then the expected demand for each year. The results are displayed in table 1.

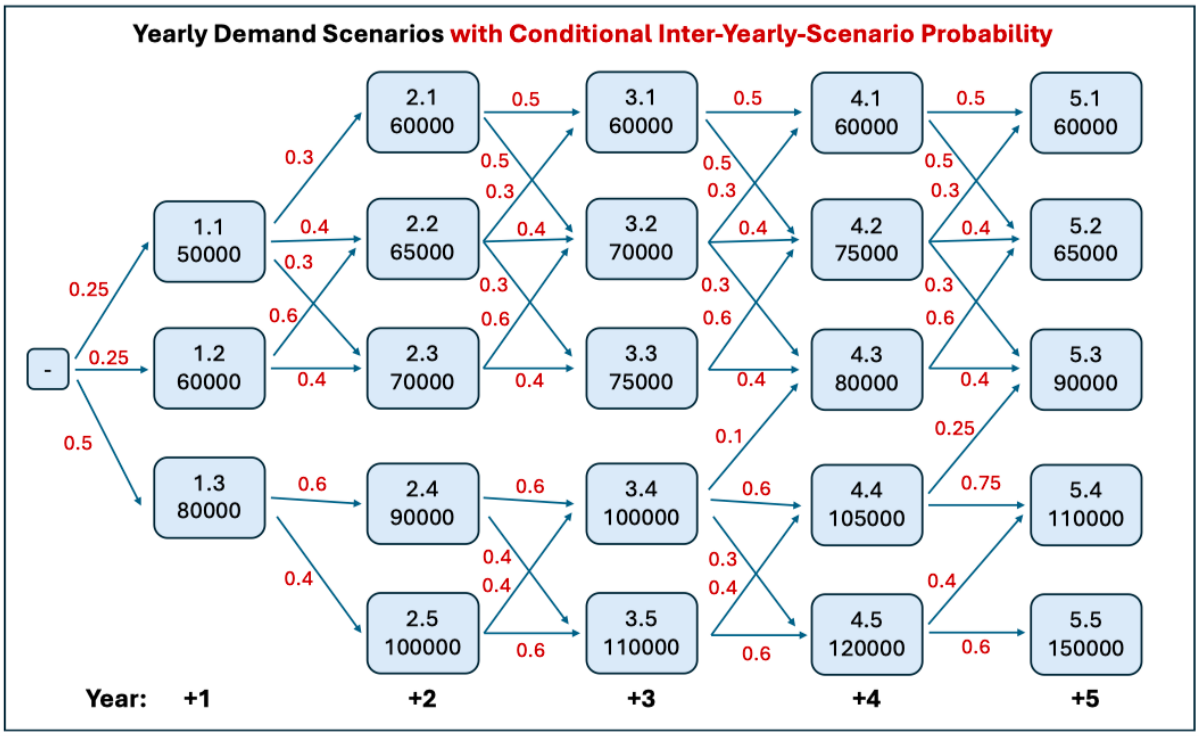

Figure 1: Yearly Demand Scenarios with Conditional Inter-Yearly-Scenario Probability

scenario	demand	probability
1.1	50000	0.25
1.2	60000	0.25
1.3	80000	0.5
2.1	60000	0.075
2.2	65000	0.25
2.3	70000	0.175
2.4	90000	0.3
2.5	100000	0.2
3.1	60000	0.11
3.2	70000	0.24
3.3	75000	0.15
3.4	100000	0.26
3.5	110000	0.24
4.1	60000	0.13
4.2	75000	0.24
4.3	80000	0.16
4.4	105000	0.25
4.5	120000	0.22
5.1	60000	0.14
5.2	65000	0.25
5.3	90000	0.2
5.4	110000	0.28
5.5	150000	0.13

Table 1: Probability of each scenario

We can compute the yearly expected demand:

year	1	2	3	4	5
demand	67,500	80,000	87,000	91,000	93,000

Table 2: Expected demand for each year

Considering a coefficient of variation of 0.3 over the 13 4-weeks periods, we can compute the maximum expected demand for each of these periods for each year:

demand share for period $i\times$ expected demand for a specific year$\times$(1+3.29 coefficient of variation)

We spread this formula over every 4-weeks period and every year.

We also need to consider some daily variation related to a coefficient of variation of 0.2. We will proceed as previously. So as to get the maximum expected demand per day.

Since we want to be able to fulfill 99% of the demand over 5 years, we need to size the locker room for the highest demand, reached during year 5 in this scenario. In table 3, we have the maximum demand for each day of the 4-weeks period for year 5.

4 week period	1	2	3	4	5	6	7	8	9	10	11	12	13	total/day
Su	920	767	920	1073	1380	920	613	1533	1533	920	1227	1533	1993	15333
Mo	1840	1533	1840	2147	2760	1840	1227	3067	3067	1840	2453	3067	3987	30666
Tu	2760	2300	2760	3220	4140	2760	1840	4600	4600	2760	3680	4600	5980	45999
We	2760	2300	2760	3220	4140	2760	1840	4600	4600	2760	3680	4600	5980	45999
Th	3312	2760	3312	3864	4968	3312	2208	5520	5520	3312	4416	5520	7176	55199
Fr	4048	3373	4048	4723	6072	4048	2699	6747	6747	4048	5397	6747	8771	67466
Sa	2760	2300	2760	3220	4140	2760	1840	4600	4600	2760	3680	4600	5980	45999

Table 3: Maximum yearly demand for year 5

Now that we have this, we need to know the maximum number of parcels that can need to be stored in the same time in the room.

Maximum number of parcels in the room at the same time

We know the probability that a parcel arrives at time p and is picked up at time p. These probabilities are displayed in figure 2.

Probability (X10,000) That a Given Parcel is Deposited In Hour d and Then Picked Up by the Customer in Hour p

Deposit Time d \ Pickup Time p	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	Total
0	30	20	4	4	4	10	20	24	32	20	12	12	8	0	0	0	0	0	0	0	0	0	0	0	200
1	0	24	12	6	4	10	20	24	32	20	16	12	12	8	0	0	0	0	0	0	0	0	0	0	200
2	0	0	12	6	3	5	10	12	16	10	8	6	6	4	2	0	0	0	0	0	0	0	0	0	100
3	0	0	0	12	6	3	10	12	16	10	8	6	6	4	4	3	0	0	0	0	0	0	0	0	100
4	0	0	0	0	36	24	30	36	48	30	24	18	18	15	12	9	0	0	0	0	0	0	0	0	300
5	0	0	0	0	0	36	45	45	48	30	24	18	18	15	12	9	0	0	0	0	0	0	0	0	300
6	0	0	0	0	0	0	60	60	72	44	32	24	24	20	16	16	20	12	0	0	0	0	0	0	400
7	0	0	0	0	0	0	0	72	100	48	32	24	24	20	16	16	20	16	12	0	0	0	0	0	400
8	0	0	0	0	0	0	0	0	150	120	60	48	48	36	24	24	30	24	18	18	0	0	0	0	600
9	0	0	0	0	0	0	0	0	0	120	96	90	60	36	24	24	36	42	30	18	12	12	0	0	600
10	0	0	0	0	0	0	0	0	0	0	90	108	108	90	30	24	36	42	30	18	12	12	0	0	600
11	0	0	0	0	0	0	0	0	0	0	0	120	138	90	48	36	36	48	30	18	12	12	12	0	600
12	0	0	0	0	0	0	0	0	0	0	0	0	80	100	40	32	32	40	40	12	8	8	4	4	400
13	0	0	0	0	0	0	0	0	0	0	0	0	0	120	108	60	48	60	60	60	36	24	18	6	600
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	160	144	112	112	80	80	48	32	24	8	800
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	120	120	90	66	36	24	18	6	600
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	150	120	90	48	36	24	12	600
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120	180	120	84	60	24	12	600
18	0	0	0	0	0	0	10	40	20	10	0	0	0	0	0	0	0	0	100	120	100	70	20	10	500
19	5	0	0	0	0	0	15	50	40	15	0	0	0	0	0	0	0	0	0	100	150	75	35	15	500
20	8	0	0	0	0	0	20	60	40	12	8	8	4	0	0	0	0	0	0	0	80	100	40	20	400
21	10	4	0	0	0	0	12	36	30	8	4	4	2	0	0	0	0	0	0	0	0	30	40	20	200
22	20	10	4	0	0	0	12	36	30	8	4	4	2	0	0	0	0	0	0	0	0	0	30	40	200
23	30	14	6	0	0	0	16	40	36	16	6	4	2	0	0	0	0	0	0	0	0	0	0	30	200
Total	103	72	38	28	53	88	280	547	710	521	424	506	560	558	496	517	610	786	790	720	626	495	289	183	10000

Figure 2: Probability (X10,000) That a Given Parcel is Deposited In Hour d and Then Picked Up by the Customer in Hour p

We use this table as well as the demand from table 3 to code the maximum number of parcels that can be in the room in the same time. The code is written in the python file *case4_task01_parcelNumber*. The results are plotted in figure 3.

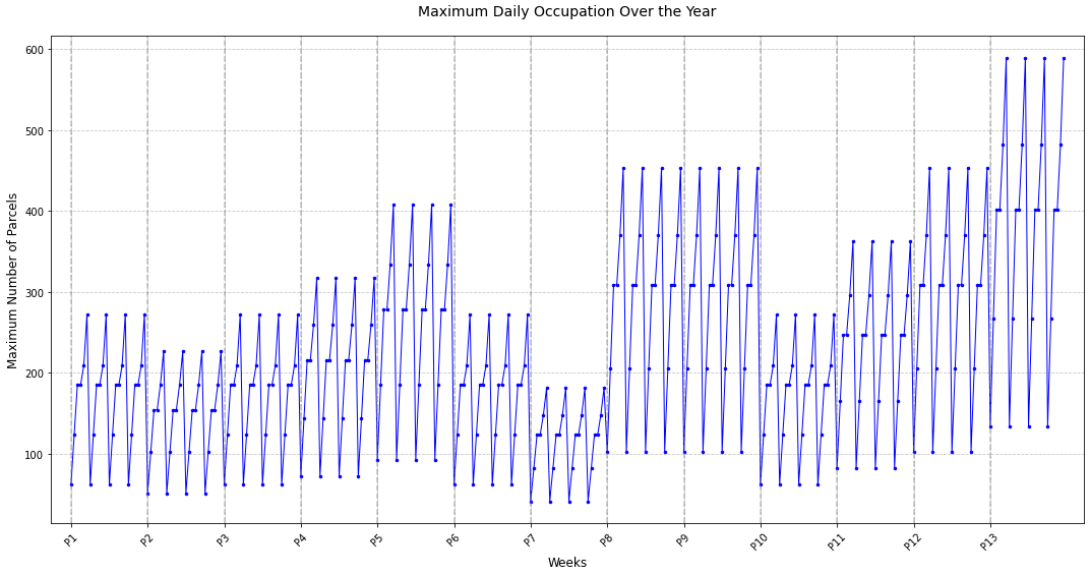

Figure 3: Maximum number of parcels in the locker room over year 5

These numbers are to fulfill 100% of the demand. Nevertheless, we only have to fulfill 99% of the demand and there are only a few days when the demand can be higher than 450 parcels. So we will try to see how much demand can be fulfilled if we decide to size the locker room in order to fulfill the demand up to the peak displayed in figure 4 (red line).

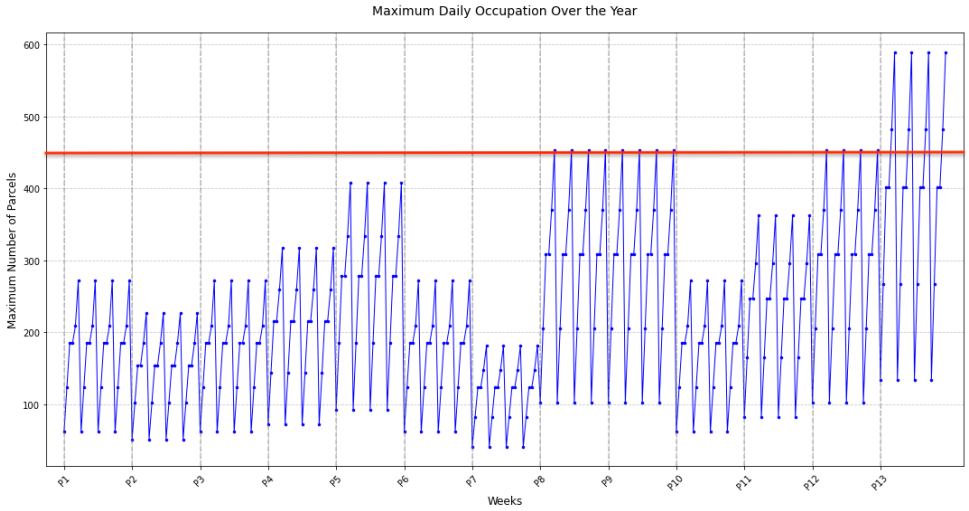

Figure 4: Maximum number of parcels in the locker room over year 5 - sizing for peak pointed by the arrow

We run the code and get that **99%** of the demand can be fulfilled. So we will size the room for this demand.

The output of the code gives that a maximum of **450** parcels can be in the locker room in the same time.

Maximum number of parcels of each type in the room at the same time

Using the table showing the probability of having a parcel of a specific size, we get the maximum number of parcels of each type in the locker room at the same time. We note that in average, parcels remain in the locker room for 2 hours and 45 minutes.

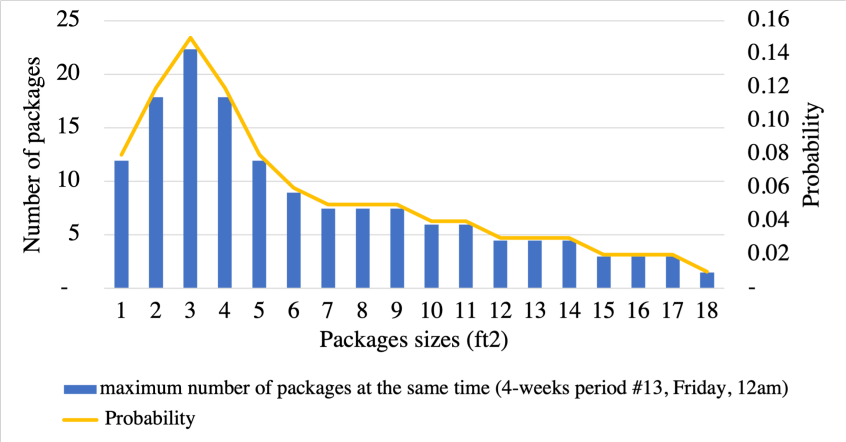

Figure 5: Distribution of the different sizes of parcels at the peak time (Friday, 13th 4th week period, 12:00)

Number of different locker sizes

In this subsection, we are evaluating the space saved in three different configurations:

- 8 different locker sizes
- 10 different locker sizes
- 18 different locker sizes

We have done all computations in the excel file and we display the results in figure 6

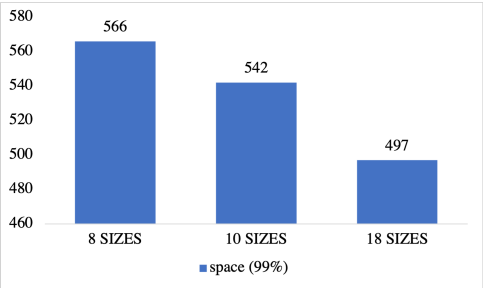

Figure 6: Total space required for three proposals of total number of different sizes of lockers

We notice that we reduce our space requirement by 12% if we have 18 different locker sizes instead of 8. So we will go with a design having 18 different sizes (1 to 18 square feet).

Lockers design

We will create stack of lockers of the same type. All stacks will be 6 feet tall and 2 feet deep. These dimensions are displayed in table 4.

	x	y	z	number/stack	volume	floor area
L18	1.5	2	6	1	18	3
L17	1.42	2	6	1	17	2.8
L16	4	2	3	3	16	8
L15	2.5	2	3	2	15	5
L14	3.5	2	2	2	14	7
L13	3.25	2	2	3	13	6.5
L12	3	2	2	2	12	4
L11	2.75	2	2	3	11	5.5
L10	2.5	2	2	2	10	5
L9	2.25	2	2	3	9	4.5
L8	2	2	2	2	8	4
L7	1.75	2	2	2	7	3.5
L6	1.5	2	2	3	6	3
L5	1.25	2	2	3	5	2.5
L4	1	2	2	3	4	2
L3	1.5	2	1	6	3	3
L2	1	2	1	6	2	2
L1	1	2	0.5	12	1	2

Table 4: Dimensions and stacking details

We have designed these lockers using the python code *case4_task01_lockers*. They are displayed in figure 7.

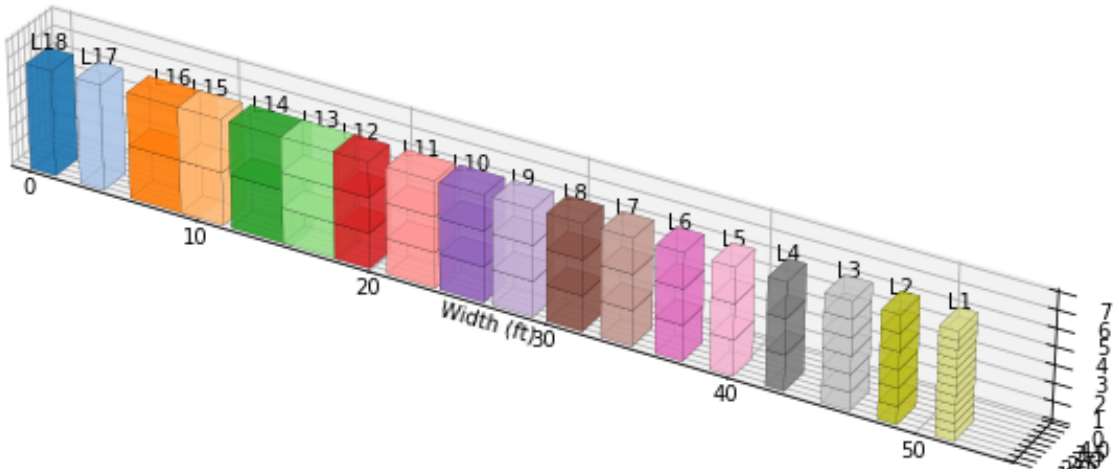

Figure 7: 3D Visualization of the 18 different stacks of lockers

Number of stacks of each size

Considering the following sizes and the number of lockers per stack, we will determine, according to the volume, the number of stacks of each type we will need. So as to have a locker bank that is not 200 feet wide, we will propose 4 groups of lockers that would be back to back as shown in figure 8.

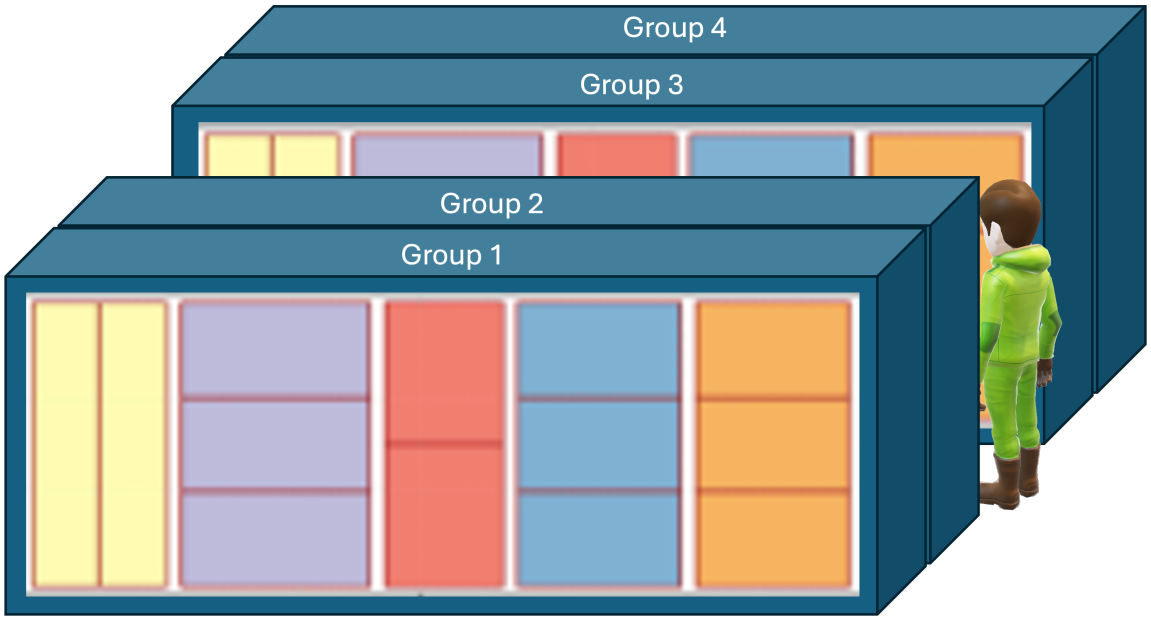

Figure 8: 3D design of the locker bank

We will put all different types of locker in each of these groups and make sure we have enough lockers of each type to fulfill 99% of the demand as required. The results are displayed in table 5.

Locker type	# of stacks group 1	# of stacks group 2	# of stacks group 3	# of stacks group 4	# of lockers
L1	1	0	1	1	36
L2	2	2	1	3	54
L3	3	3	3	3	72
L4	4	5	5	4	54
L5	3	3	3	3	36
L6	2	2	2	2	27
L7	2	2	2	2	24
L8	2	2	2	2	24
L9	2	2	2	2	24
L10	1	1	1	1	18
L11	1	1	1	1	18
L12	1	1	1	1	14
L13	1	1	1	1	12
L14	1	1	1	1	15
L15	1	1	1	2	9
L16	1	1	1	1	9
L17	3	2	2	2	9
L18	1	1	2	2	5

Table 5: Number of packages and stack for each locker type

We will use these number to code the layout of the locker room.

Overall layout of the locker room

We have generated a python code (*case4_task01_layout*) that will create a possible layout. Here is the result we get:

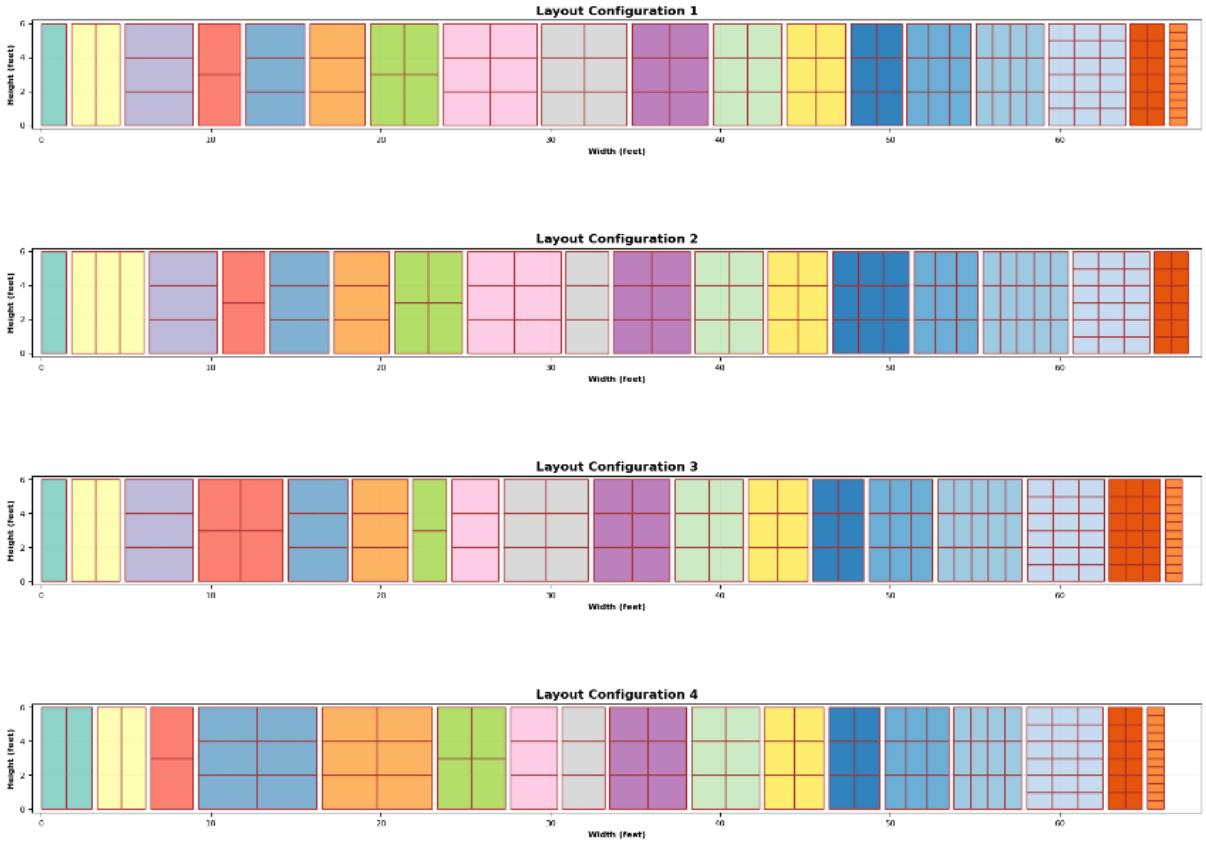

Figure 9: First overall layout of the locker bank

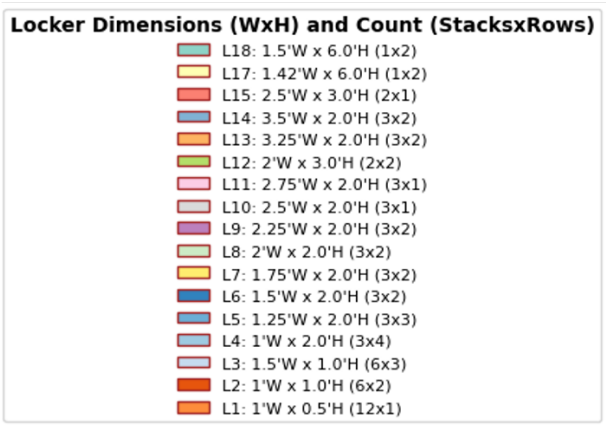

Figure 10: Labels of the different lockers sizes

This locker bank is 70 feet long and 15 feet wide.

TASK 2

Choosing the right model

In order to simulate the arrival time of N packages into a locker for one given hour, we first need to understand how they are delivered. Since there is no fixed replenishment time for the lockers, we assume that the N packages are delivered randomly during this hour. Each package arrival is independent. This is why we decided to model this event by an **Exponential Law**. The exponential law is interesting relative to its continuity and its memoryless properties. It is a commonly used distribution that models the time between independent events in a Poisson process. Here is why we believe the exponential law is a relevant law compared to others (Uniform Law).

1. Each package's arrival is independent of the others. A logistic service provider or customer might bring packages randomly, without a fixed schedule, which can create a continuous arrival pattern.

2. While there might be slight variations, we can assume the average rate of arrivals is relatively constant within the hour. If the mean number of arrivals per hour is N, we can use an exponential inter-arrival time distribution with rate $\lambda = \frac{N}{60}$ (assuming we measure time in minutes).

3. The exponential distribution captures randomness, meaning that there could be some clustering of arrivals or gaps, which aligns with the unpredictable nature of human behavior and logistic service operations.

Implementing the model

Given N packages to arrive within a given hour, the process for generating arrival times with an exponential distribution would work as follows:

1. We first need to define the rate parameter, which is $\lambda = \frac{N}{60}$

2. Use an exponential distribution with rate N inter-arrival times, representing the time between each consecutive package's arrival.

3. Starting from time the start of the hour, compute each package's arrival time by cumulatively summing these inter-arrival times.

4. Each generated set of arrival times within the hour will represent one possible scenario. We may then save the scenarios (in a list or an array).

Implementing and running the code in Python

Now that the process is implemented, we will model it in a function that will later on help us with the simulations.
Here is an explanation of the code.

- **Objective:**
  Generates a list of N arrival times in `datetime` format on a specific day and hour, with arrivals following an exponential distribution within that hour.

- **Parameters:**
  `start_datetime` (datetime): The start date and time as a datetime object (for instance: `'2024-01-01 15:00:00'`). This sets the day and hour for the arrivals.
  N (int): Number of arrival times to generate within the hour.

- **Returns**
  list: List of datetime objects representing arrival times within the specified hour.

Here is an example of the code running for 20 expected packages, arriving at 3:00 p.m., January the 1rst, 2024 (`start_date = dt.datetime(2024,1,1, 15)`).
The main function in the code is `generate_datetime_arrivals(start_date, N=20)`, it returns a list of 20 different times, from 3:00 to 4:00 p.m..

```
   arrival_times = generate_datetime_arrivals(start_date, N=20)
for arrival in arrival_times:
print(arrival)
```

```
2024-01-01 15:01:30
2024-01-01 15:05:24
2024-01-01 15:09:06
2024-01-01 15:13:03
2024-01-01 15:16:12
2024-01-01 15:24:13
2024-01-01 15:24:27
2024-01-01 15:24:46
2024-01-01 15:27:28
2024-01-01 15:27:51
```

```
2024-01-01 15:31:52
2024-01-01 15:33:50
2024-01-01 15:34:51
2024-01-01 15:39:04
2024-01-01 15:42:27
2024-01-01 15:43:08
2024-01-01 15:44:36
2024-01-01 15:45:56
2024-01-01 15:51:42
2024-01-01 15:54:11
```

This function will prove to be very important when coding the simulator for later tasks.

TASK 3

For this Task, we will develop a simulator that generates instances of dynamic demand across the five-year horizon according to the information above relative to the projected demand patterns and scenarios. This can be achieved in a few steps. Each step corresponds to a function/method written in Python code. The code is available in the annex files. *However, please take into consideration that running this code takes a lot of time, given the important number of computations we need to achieve for each package arrival and departure.*

Step 1: Generate Demand Scenario For The Next Five Years

In this section we will be taking into account the Yearly Demand Scenarios with Conditional Inter-Yearly-Scenario Probability to generate different scenarios over the years, taking into account the conditional probabilities.

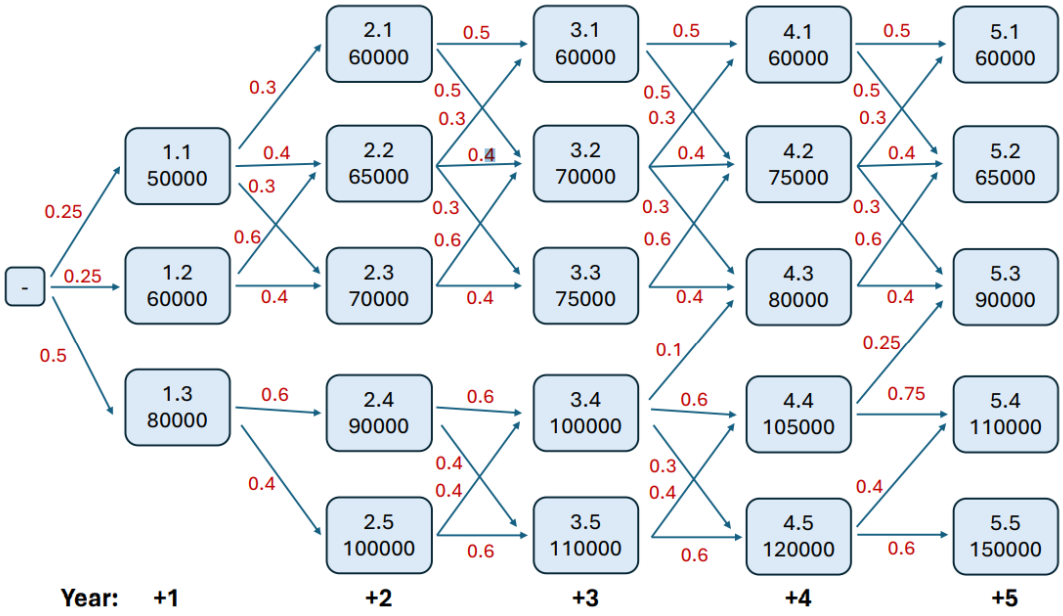

Figure 11: Yearly Demand Scenarios with Conditional Inter-Yearly-Scenario Probability

For this, we will generate 5 random numbers between 0 and 100 that will be then assigned to a different outcome for each year. When the demand is determined for year y, we save it into a list and use it to calculate scenario $y + 1$.
As an example let's look at what happens in year 1. We can suppose that the generated number is X_1.

- if $X_1 \leq 25$, we will consider scenario 1.1.
- if $26 \leq X_1 \leq 50$, we will consider scenario 1.2.
- else, we will consider scenario 1.3.

For year $y+2$, we use the same method with X_2, but also taking into account the previous result.
We then compute this for the five next years. LLM models (such as Chat GPT) are very useful in this particular case as they wrote the 130 lines of repetitive code in a matter of seconds.
This is the output for a scenario that was generated using this code:

```
In[3]:generate_scenarios()
Out[3]:  [80000, 90000, 100000, 80000, 110000]
```

We can run Monte Carlo Simulations in order to understand how does the different scenarios fluctuate from one another.

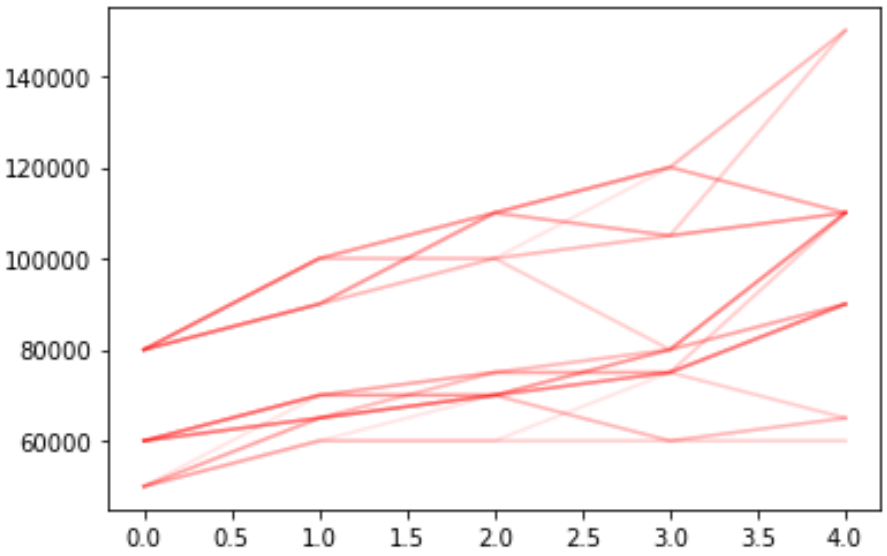

Figure 12: Monte Carlo Simulation - Different Demand Scenarios

The more the lines are red, the more this event has chances to occur (and vice versa). Now that we have taken into consideration the yearly demand for each year, we will dive into the daily demand, and later on, we will focus on the hourly demand.

Step 2: Simulating The Daily Demand For Each Year

Now that the yearly demand has been established. We can focus in taking into account the monthly and the weekly seasonality.

In order to make each simulation unique and integrate more randomness into each processes, we will take into consideration the *Demand Share For Each Week Period*, the *Demand Share for Each Day of the Week*, but also the coefficient variation for each of these parameters.

4-week period:	1	2	3	4	5	6	7	8	9	10	11	12	13	Total
Demand share:	0.06	0.05	0.06	0.07	0.09	0.06	0.04	0.10	0.10	0.06	0.08	0.10	0.13	1.00
Coefficient of variation:		*0.3*	*for each 4-week period, subject to annual sum equals to 1*											
Day-of-week:	Su	Mo	Tu	We	Th	Fr	Sa	Total						
Demand share:	0.05	0.10	0.15	0.15	0.18	0.22	0.15	1.00						
Coefficient of variation:		*0.2*	*for each 4-week period, subject to annual sum equals to 1*											

Figure 13: Demand Share

In order to take into consideration this randomness, we will modify the Demand Share for each iteration by adding a random generated number following a normal distribution where the ratio $\frac{\sigma}{\mu} = CV$. Following this transformation, the sum of the shares may not be equal to 1. This is why we apply a normalization process on them. After that, the sum of the shares are equal to one. However there are some benefits and inconveniences with this method:

- We are assured the Yearly Demand will remain the same then after the Step 1.
- By normalizing the values, we may 'lose some randomness' in the process.

After iterating through each day, we obtain the following table (it spans all the way to 2029

Date	Daily Demand	Year
01/01/2024	23	2024
02/01/2024	107	2024
03/01/2024	121	2024
04/01/2024	84	2024
05/01/2024	168	2024
06/01/2024	171	2024
07/01/2024	164	2024
08/01/2024	16	2024
09/01/2024	75	2024
10/01/2024	122	2024
11/01/2024	140	2024
12/01/2024	133	2024
13/01/2024	224	2024
14/01/2024	115	2024
15/01/2024	30	2024
16/01/2024	101	2024
17/01/2024	111	2024
18/01/2024	68	2024
19/01/2024	134	2024
20/01/2024	156	2024
21/01/2024	100	2024
22/01/2024	21	2024
23/01/2024	64	2024

Table 6: Daily Demand for January 2024

The table was computed using the yearly parameters:
`[60000, 70000, 70000, 80000, 110000]`
The function used to compute the Daily Demand is called `y_demand`.
Similar to the previous step, we can visualize the volatility by computing Monte Carlo Simulations:

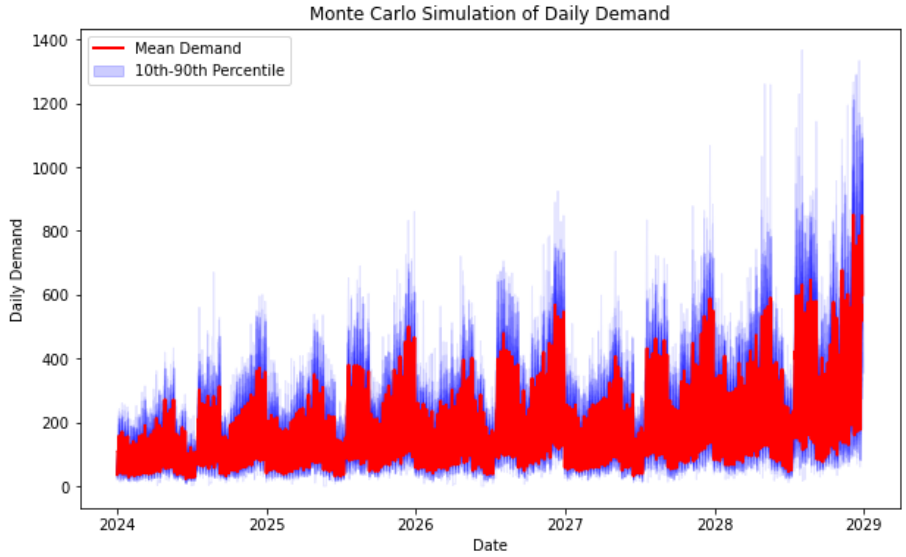

Figure 14: Monte Carlo Simulation - Daily Demand - Fixed Yearly Demand

Depending if the variable `yearly_demand` is local or global, we may modify the different scenarios. We can then compute Monte Carlo Simulations by with fluctuating scenarios

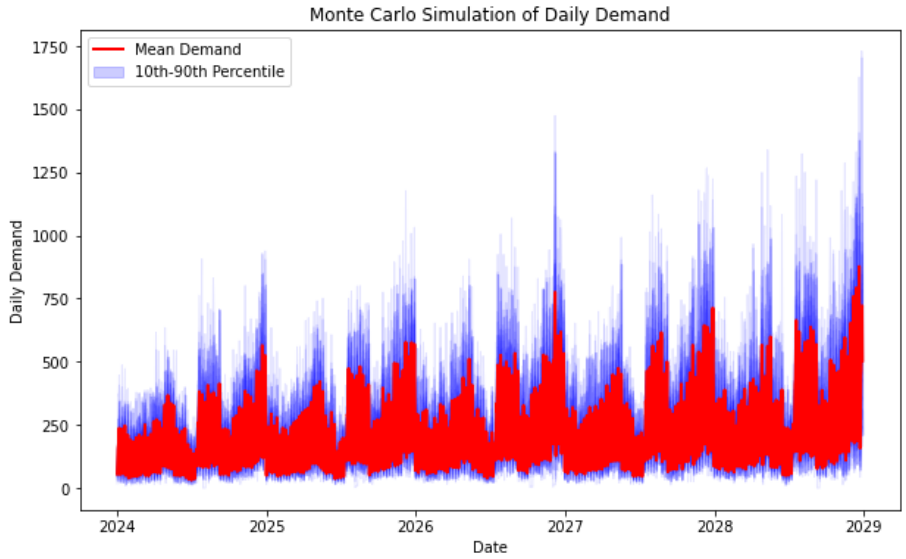

Figure 15: Monte Carlo Simulation - Daily Demand - Variable Yearly Demand

Compared to the penultimate figure, here, the Percentiles are more spread out.

Step 3: Simulating The Hourly Demand For Each Day

Now that the Daily Demand has been evaluated and saved in a `DataFrame` object (in the case of this simulation `daily_demand_df`. We may start looking into the hourly demand

for each day. First of all, we will focus on the Hourly Deposit rate.
In order to compute the number of package deposit per hour, we may use the table of the joint distribution of D the random variable associated to the deposit time, and P the random variable associated to the pickup time.
Using the property of the marginal distributions, we can evaluate the marginal distribution of D:

$$f_D(x_i) = \sum_{j=0}^{23} f_{D_i,P_j}$$

Similarly, even if it is less relevant for this part:

$$f_P(x_j) = \sum_{i=0}^{23} f_{D_i,P_j}$$

We obtain the following marginal distribution: Once again, we could have added random-

Hour	Distribution
0	0.02
1	0.02
2	0.01
3	0.01
4	0.03
5	0.03
6	0.04
7	0.04
8	0.06
9	0.06
10	0.06
11	0.06
12	0.04
13	0.06
14	0.08
15	0.06
16	0.06
17	0.06
18	0.05
19	0.05
20	0.04
21	0.02
22	0.02
23	0.02

Table 7: Hourly Distribution of Demand

ness in thee values by calculating a Coefficient Variation, unfortunately, we lacked the necessary data and time to take it into account. Furthermore, the volatility generated by this CV would be negligible compared to the previous ones.
The function that is used to evaluate the yearly demand is:

`calculate_hourly_demand(daily_demand_df, year)`

Here, we started filtering the data by year, otherwise, the computational time would be 5× longer. If it is not too time consuming for this Step, it will become in the next.
The subtlety in this part is to have an hourly demand that is an integer. If it wasn't too important in the previous Steps, it is now, since we will be wanting to apply the function of **Task 2** (that can only take an integer as a parameter).
To overcome this problematic we used the following algorithm for each day:

- We start by calculating how much the hourly demand values add up to in total, and then we compare it to the desired total demand for the day. If there's a difference between these two numbers, we call it the "discrepancy." This discrepancy represents the adjustment we need to make to meet the exact daily demand
- If the total of the hourly demand is less than the daily demand, we need to add some extra demand. To do this, we go through the hourly values one at a time, adding 1 unit of demand to each hour, distributing it evenly throughout the day. We keep adding until we reach the desired daily total.
- If the total of the hourly demand is more than the daily demand, we need to reduce it slightly. In this case, we subtract 1 unit of demand from each hour, again distributing the reduction evenly across the day, until we match the daily target.

Evidently, there is some downsides of using this method. However the upside is that it ensures that the daily demand is not altered:

$$D_{Daily} = \sum_{i=0}^{23} D_{Hourly,i}$$

Combined to the normalization process in Step 2, we can assure that the Yearly Demand remains the same as the one evaluated in Step 1.

Step 4: Generating the Deposit and Pickup Time Using Task 2

Using the results in Task 2, we are able to generate N different times within an hour. We can therefore return a table where each line represents an order. We will now try to compute the retrieval time for each line. To do so we will be modifying the table of joint probabilities (that was given to us in the Casework) to a table of conditional probabilities, since from now on we know the deposit time for each package.
Let P_i and D_i be respectively the random variables for the deposit time and the pick up time for package number i:

$$P(P_i = j | D_i = k) = \frac{P(P_i = j, D_i = k)}{P(D_i = k)}$$

The table is saved in the file *Conditional Pickup.csv* and is called in as a date frame in the Python script when it will come to determine the pickup time.
To simulate the Pickup Time we will us the method `random.choice()` in the `numpy` library. This will return an simulated pickup hour (for instance `19:00`). We can then choose a random value between `19:00` and `20:00`, this returns the final pickup hour for the package.

Step 5: Generating the Package Sizes and integrating a Graphical User Interface

With the data we were given for this study, we utilized the probability distribution function of the package sizes to add a Size (S) for each package. Similarly as was already did in the pickup time, we used `numpy.random.choice()` in order to generate a random package size (from `L1` to `L18`). The L letter gives us the minimum locker size in which the package could be deposited.
We can now run the simulation.
In order to help the user understanding the simulator, we decided to add a GUI. Here are the different windows that will appear after running the code:

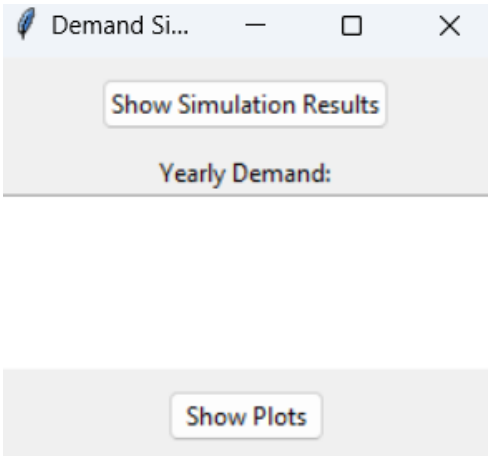

Figure 16: GUI using Tkinter library

After clicking on the *Show Simulation Results*, we can see the yearly demand simulated for the next five years.

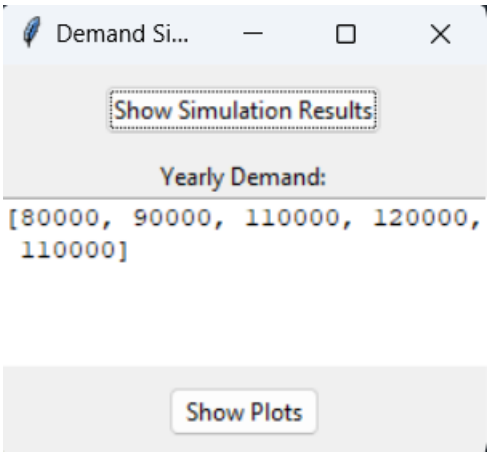

Figure 17: 5 Year Simulation

Finally, the *Show Plots* button opens another window that is designed to help with the overall visualization:

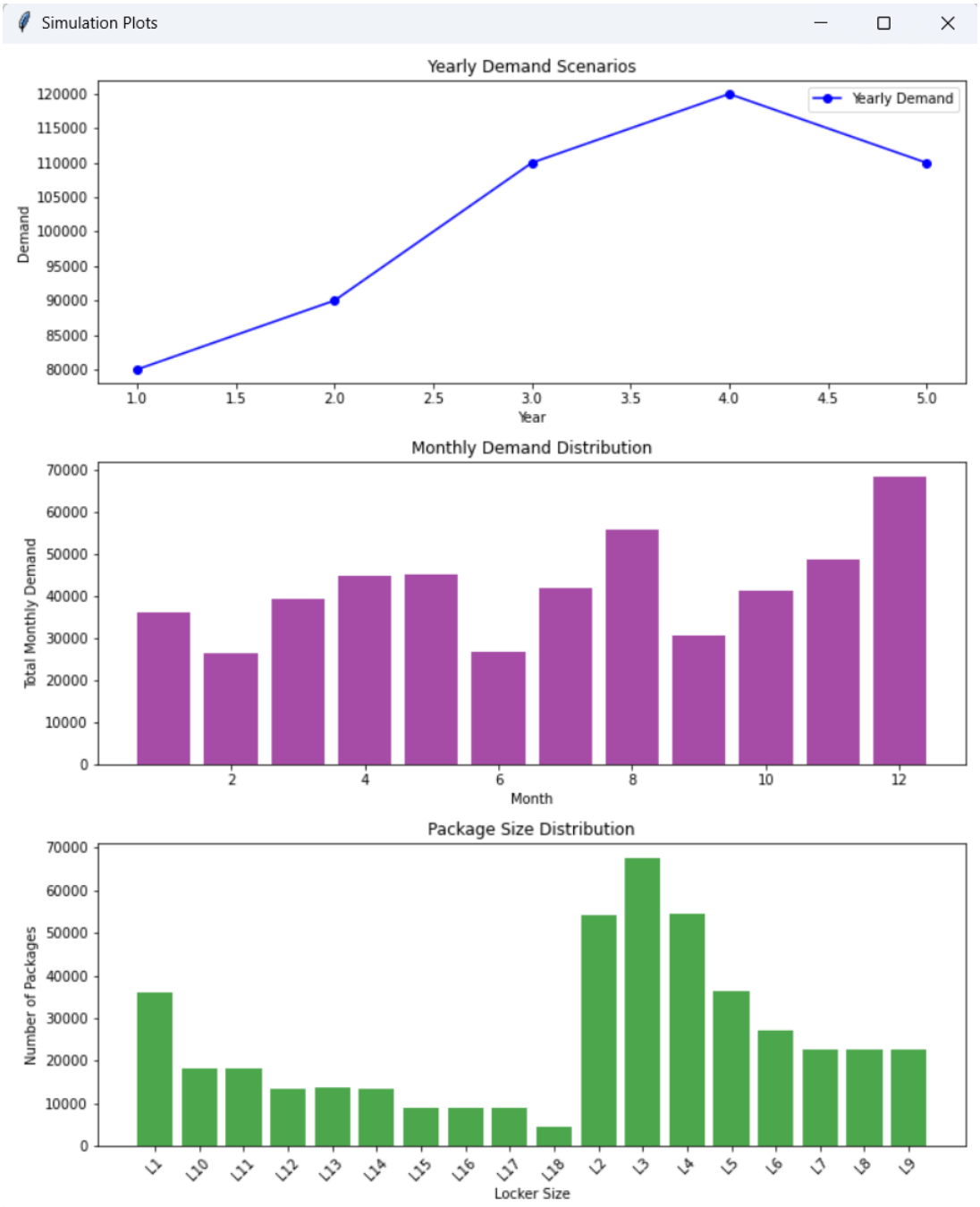

Figure 18: Simulator Results

TASK 4

The simulation is designed around a few critical components:

- **Locker Setup:** Lockers are organized by size and managed dynamically based on occupancy. Each locker size category has a set number of lockers available, and each locker can only be assigned if it is unoccupied at the time of a request. Lockers are freed based on their expected pickup times, making them available for future assignments. This approach simulates a realistic, time-dependent locker management system.
- **Demand Forecasting:** Demand is modeled over several years using task 3 results, which introduce variability to reflect different usage scenarios. Probabilistic demand scenarios are created for each year, and seasonal adjustments are applied on a monthly and weekly basis. This step ensures that demand patterns include high and low usage periods, giving a comprehensive view of locker utilization across different times.
- **Arrival and Pickup Simulation:** Based on the generated demand, hourly arrivals are simulated for each day. Using exponential distribution to control arrival rates, this process generates individual package requests with specific arrival and pickup times, based on historical pickup probabilities. The arrival times simulate realistic user behavior by clustering around certain hours while accounting for lower activity during early morning or late-night hours.

Heuristic Implementations:

Strawman Heuristic: Size X in Locker X

For the Strawman, we have decided that since the number of lockers have been dimensioned relative to the size of the object, we will implement the following heuristic:

1. Initialize 18 empty dictionaries that will represent the 18 different Locker Groups. Each locker will be identified by a number. For instance: `L1_08`, would be locker number 8, max size = 1 sqft. For each key in the dictionary, there will be associated 3 values:
    - the binary value (0,1), will show if the locker is available.
    - the departure date, useful when it comes to deleting a package from the dictionary.

- the size of the package.

2. Run the simulation detailed in Task 3 and retrieve `arrival_df`, which contains the expected arrival and departure times for each package.

3. Iterate for every 15 minutes and evaluate the situation in a certain Locker Group `LX`:

   - Iterate through all the awaiting packages expected pickup date, if one pickup time has been met: delete the value in the dictionary and replace it by 0 to indicate that the locker is available and add +1 to the counting variable `finalized_orders`.
   - If a package of size X has been received: put it in the next available Locker `LX`.
   - If a package has been received but there is no more space available in Group Lockers `LX`. Delete this package and add +1 in the counting variable `rejected_orders`

4. Move the `timestamp` to the next M minutes and reiterate the process (M will have influence on the sensitivity of the results)

5. Return the results

Smarter Heuristic - Upgrading Strawman

For the smarter heuristic, we will take the *Strawman Heuristic* Model and improve it slightly.
Let's suppose that a package size X arrives at the locker group.

1. First we will apply the same strategy as the *Strawman Heuristic*: we look if there is space available for a locker size `LX`.

   - if there is space available: assign package `X` to Locker Group `LX`.
   - else look at Locker Group `LX + 1`, if there is space available, assign the package to this Locker Group.

2. repeat the same process as in the Strawman Heuristic:

   - If a package has been received but there is no more space available in Group Lockers `LX + 1`. Delete this package and add +1 in the counting variable `rejected_orders`

- If a package of size X has been received: put it in the next available Locker `LX` or `LX + 1` and add one to the counting variable `finalized_orders`. Furthermore, if a package of size X is assigned to Group Locker`LX + 1`, add one to the counting variable `heuristic2`. This will be very helpful for the other tasks when we will have to analyze KPIs, simulator efficiency and overall strategy coherence.

3. Move the `timestamp` to the next M minutes and reiterate the process (M will have influence on the sensitivity of the results).
   In our case, for the first simulations we will be taking $\Delta T = 15$ minutes per iteration.

4. Return and analyze the results.

Assignment Methodology

In this setup, the locker assignment system works by leveraging a defined methodology within the Locker Management class in the code. It assigns lockers based on current locker availability and demand. Once a package is picked up, the locker status is updated within the system, marking it as available for future assignments. This continuous tracking ensures that lockers are efficiently allocated based on real-time needs.

Locker Management Class

The LockerManagement class is central to the simulation. It manages locker assignments, releases, and availability updates based on real-time locker usage. The Locker Management Class has two primary dictionaries: **lockers**: This dictionary stores available lockers by size, with each locker size managed as a queue. When a locker is assigned, it's removed from the queue. When released, it is added back. **occupied_lockers**: This dictionary tracks assigned lockers with their expected release times, ensuring they become available after their pickup times.

Key Functions in LockerManagement:

1. **Locker Release**: The `release_locker` function releases a locker by adding it back to the available lockers once its scheduled pickup time has passed.

2. **Updating Availability**: The `update_locker_availability` function checks each occupied locker and releases those whose pickup times have been reached. This ensures that lockers are dynamically available based on real-time demand.

3. **Strawman Heuristic**: This heuristic assigns lockers only if an exact size match is available. If no exact match exists, the request is rejected. The function checks if the requested locker size is available. If so, it assigns the locker and records it in `occupied_lockers`. If not, the request is rejected.

4. **Smarter Heuristic**: This heuristic first checks for an exact match but, if unavailable, searches for the next larger locker size. This approach improves flexibility and reduces rejection rates. The function first tries to find an exact match, then checks larger sizes in ascending order, reducing rejections when the exact size isn't available.

Simulation Execution

The simulations run each heuristic and record assignment statuses and success rates. Each function returns:

1. *Assignment Log* : Detailed records of locker assignments and rejections.

2. *Success Rate*: Percentage of successful assignments for each heuristic.

Results and Analysis

Results are saved to an Excel file with detailed logs, success rates, and utilization data. Both heuristics' results are written to an Excel file with separate sheets for assignment logs, summary statistics, and utilization data. The `calculate_locker_utilization` function calculates utilization rates for each locker size, helping identify which sizes are in high demand.

The simulation results indicate that the Strawman Heuristic has Limited flexibility that results in a higher rejection rate, as requests are only assigned if an exact match is available. As compared to Smarter Heuristic: that shows a higher success rate by allowing assignments to the next larger size if an exact match is unavailable. Overall, the Smarter heuristic outperforms the Strawman approach in terms of success rate and locker utilization, demonstrating the advantage of a flexible assignment strategy in managing high-demand environments.

TASK 5 and 6

Heuristic Size X in Locker X Simulation and Analysis

Now that we have implemented this heuristic, we may evaluate the number of Assignments, Rejections, as well as the rejection rate for the last 110,000 orders. We chose to look at the 110,000 last orders because it is the moment where the demand will be as high as possible, thus putting the maximum stress on the facility. The choice of 110 000 can also be explained because it shortens the run time of the simulation by four (at least), while still maintaining a consequent sample. We obtain the following results:

Package Size	Total Orders	Assignments	Rejections	Rejection Rate
Total	**110000**	**109268**	**731**	**0.66%**
L1	8859	8816	43	0.49%
L2	13182	13108	74	0.56%
L3	16186	16138	48	0.30%
L4	13192	13145	47	0.36%
L5	8924	8877	47	0.53%
L6	6663	6622	41	0.62%
L7	5554	5530	24	0.43%
L8	5406	5387	19	0.35%
L9	5561	5511	50	0.90%
L10	4395	4354	41	0.93%
L11	4407	4382	25	0.57%
L12	3262	3243	19	0.58%
L13	3401	3367	34	1.00%
L14	3241	3223	18	0.56%
L15	2168	2118	50	2.31%
L16	2224	2165	59	2.65%
L17	2211	2164	47	2.13%
L18	1164	1119	45	3.87%

We can see that the Lockers have a very low rejection rate for the *Strawman Heuristic*. This strategy can satisfy over 99 % of the demand. However, this is not surprising as the Lockers have been designed in Task 1 to support 99% of the max demand. We can plot the overall occupation rate over time for all the lockers:

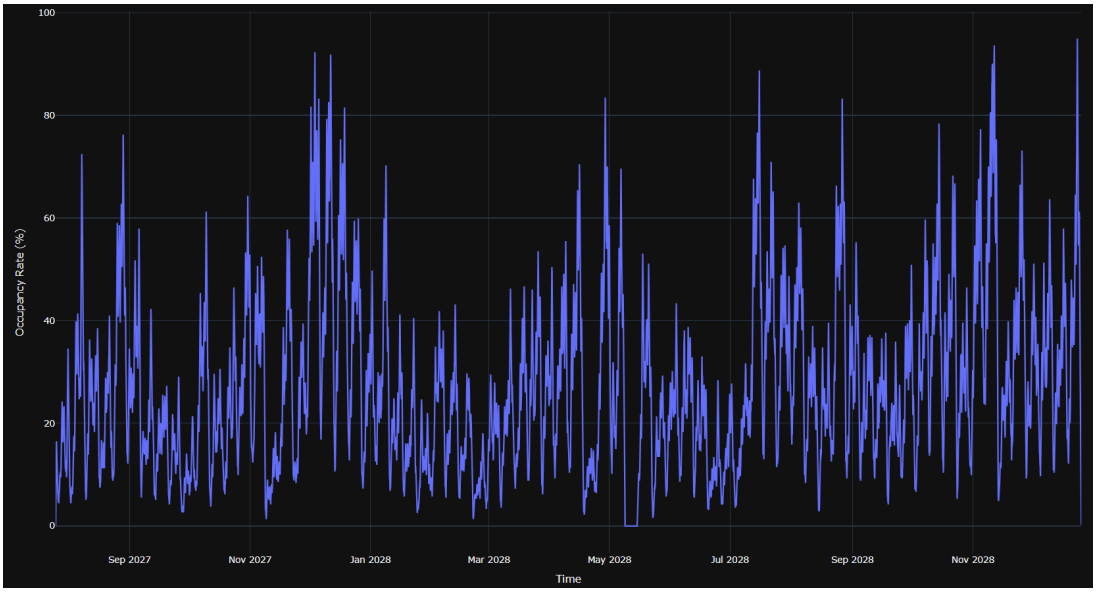

Figure 19: Occupation Rate Over Time - Strawman Heuristic - Sept 2027 to Dec 2028

This plot also returns the occupation rate for each Locker Group individually. A first look at the plot shows a very large standard deviation, sometimes reaching 100 %, but rarely. We can see some periods of lower demand, where only about 20 to 40 % of the lockers are utilized. Such low utilization explains why there is as much as 99% $\leq$ of the demand that is satisfied.

Finally, it would not be considered a real simulator if it was impossible to visualize dynamically the evolution of the state of the locker through time. This is why we created an Object Oriented Code that enables anyone to dynamically visualize any Locker Group at any given period of the year.The values on the Y-axis are the binary numbers (0,1) that represents if a package is in a given locker or not. Below are two screenshots of the program for the Locker Group L1 at two hours interval:

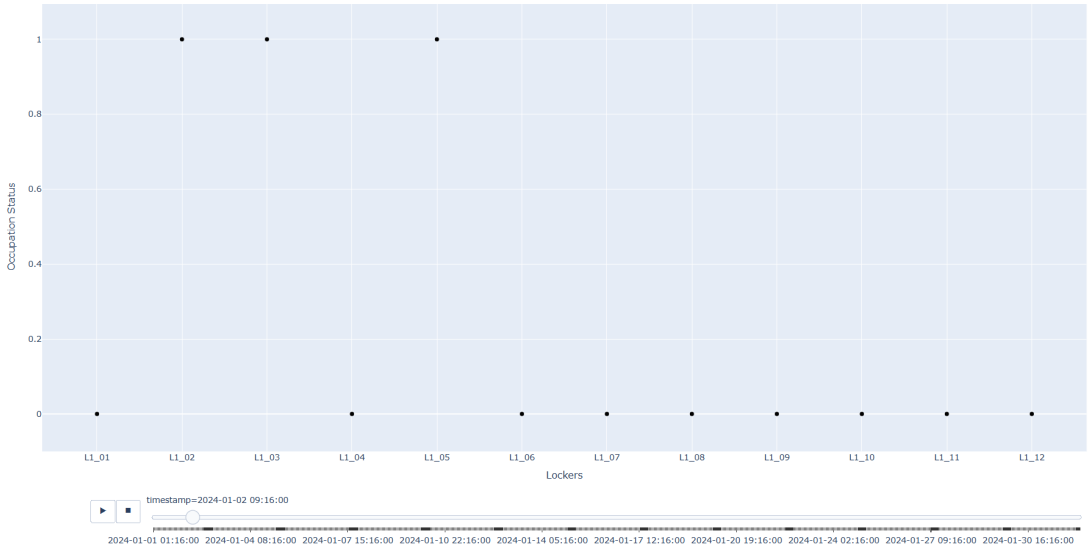

Figure 20: Occupation Status at 9:16:00 - 2024-01-02 - Locker Group L1

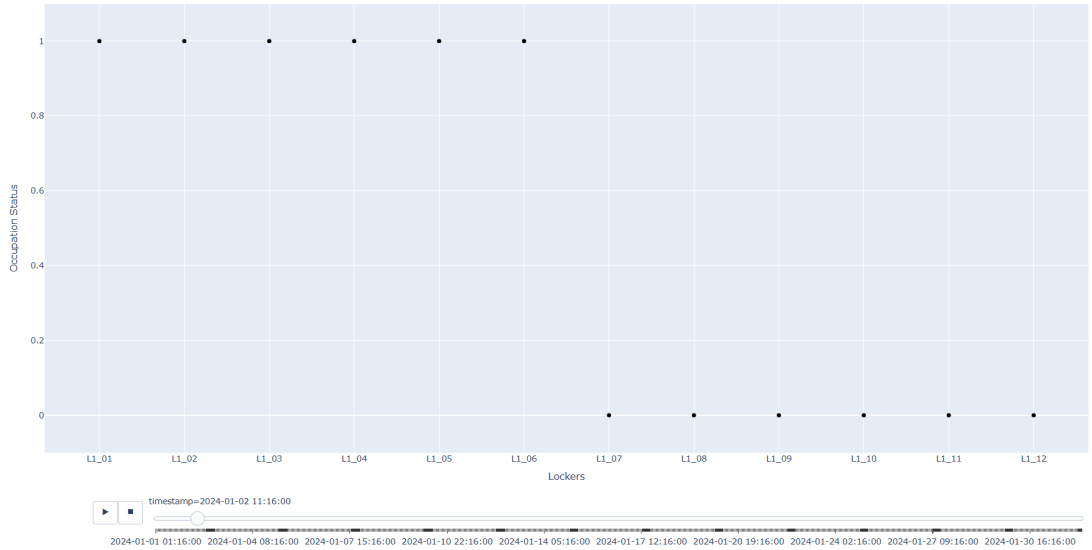

Figure 21: Occupation Status at 11:16:00 - 2024-01-02 - Locker Group L1

As we can see, `L1_01` and `L1_04` have been replenished between 9:00 and 11:00. Even though this simulator does not display a precise layout, it gives a more general overview of the Locker Bank Facility. This simulator would work for any number of lockers, demand and for any type of facility.

If this visualization method can prove to be less effective when we want to focus on larger time deltas (we would prefer to look at the previous plots), it does give a precise idea on how the heuristic works for individual packages.

Smarter heuristic - Improve Strawman, results and analysis

Similarly to the strawman, we run the code over the last 110,000 lines of the demand table. After running the code for a couple of minutes, we obtained the following results:

Table 8: Locker Simulation Results for Heuristic 2: Smarter Heuristic - Upgrading Strawman

Package Size	Total Orders	Assignments	Rejections	Rejection Rate	Upgraded Assignments
Total	**110000**	**109573**	**426**	**0.39%**	-
L1	8859	8841	18	0.20%	25
L2	13182	13150	32	0.24%	53
L3	16186	16149	37	0.23%	46
L4	13192	13151	41	0.31%	42
L5	8924	8893	31	0.35%	37
L6	6663	6634	29	0.44%	33
L7	5554	5541	13	0.23%	28
L8	5406	5397	9	0.17%	16
L9	5561	5539	22	0.40%	36
L10	4395	4379	16	0.36%	43
L11	4407	4405	2	0.05%	32
L12	3262	3256	6	0.18%	19
L13	3401	3388	13	0.38%	32
L14	3241	3231	10	0.31%	21
L15	2168	2153	15	0.69%	45
L16	2224	2184	40	1.80%	38
L17	2211	2183	28	1.27%	41
L18	1164	1100	64	5.50%	0

The last column represents the 'upgraded packages': this would be the packages that have been assigned to a locker that is bigger than its' optimal size. We can see that the rejection rate is still extremely low. Which is unsurprising, since this heuristic is a slight upgrade from the previous one.
Similarly as in the strawman heuristic, we can visualize the utilization rate of the Lockers throughout the last year (generally considered as the one with the most demand).

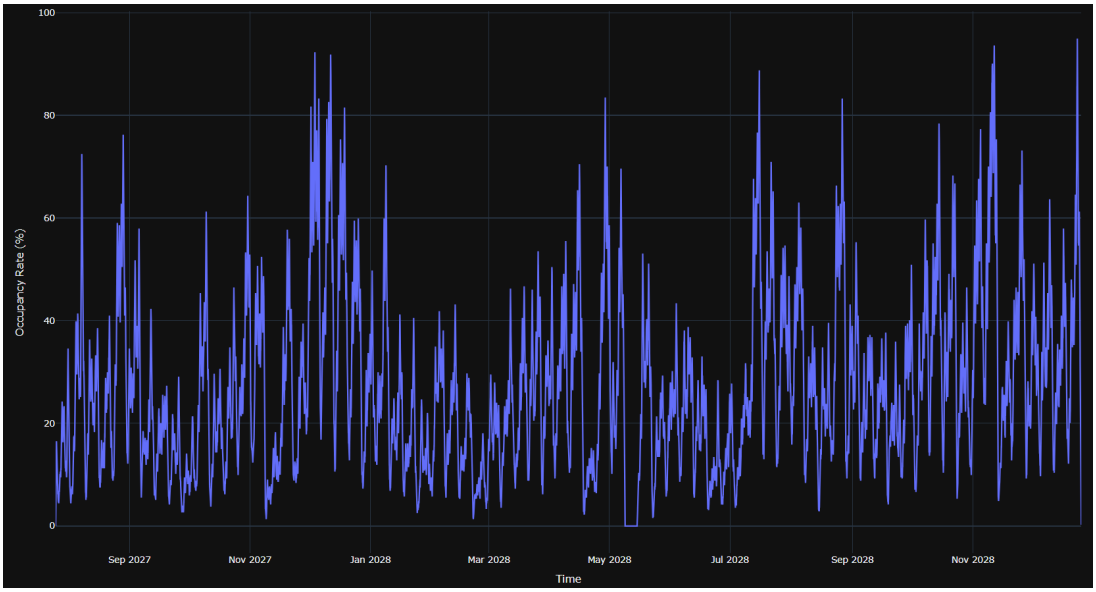

Figure 22: Occupation Rate Over Time - Smarter Heuristic - Sept 2027 to Dec 2028

We can see very close results compared to the Strawman Heuristic. This is mainly due to the fact that the lockers are most of the tine far from full capacity.
Overall, the daily, monthly and yearly service level performance are all about the same and very near to 99 %.

Financial Analysis

We will try to implement and evaluate a financial strategy derived from this Casework. To do so: we must tackle the yearly cost of installation, maintenance, and unsatisfied demand.

- For installation, rental and maintenance, please refer to Table 15
- For compensation towards the unsatisfied demand, we worked out three possible solutions:

    1. **Not deliver the package** to the customer. Therefore, compensating him for the full price of the item as well as an possible additional compensation if necessary.
        - **Pros:** Very client oriented solution, an additional compensation would help keep the client satisfaction pretty high.
        - **Cons:** Sometimes difficult to price the client's object, does not seem financially viable. If the package is rejected, the client may not receive it at the end. It is possible that the package was urgent for the customer...

2. **Store the package** in a waiting room, or a buffer area. Place the package in a locker when some room opens up.
   - **Pros:** Not a costly process to implement. Can be efficient in situations where there is a lot of variability, thus free spaces can open up the day after.
   - **Cons:** It is not a solution suggested in this Casework. There might be some safety issues with the packages. Need for a buffer space. The client will get his package later than expected.
3. **Pay another group to deliver** the package at their home. When the demand is unprecedented, their still is the possibility to ask for the assistance of 3PLs. For instance UPS or US Postal Service offers rates to transport a package from two different locations.
   - **Pros:** Very easy to implement and the fees are already listed on different websites. The package is delivered to the clients home, maximizing customer satisfaction.
   - **Cons:** The costumer may not receive the package the same day. in certain situations, he does not want the package delivered to his home. The solution can become very costly if their is too much demand.

After deliberation, we decided to go on with solution number 3. Looking on the Internet, we founds the following fees for each different sizes. We decided to consider the cheapest ones, with less regards to the transit time (this of course could be modified later on).

Volume	Min Fare	Max Fare	Company
1	$12	$15	USPS Ground Advantage
2	$12	$15	USPS Ground Advantage
3	$15	$20	UPS Ground
4	$15	$20	UPS Ground
5	$15	$20	UPS Ground
6	$16	$18	FedEx Express Saver (2–3 days)
7	$16	$18	FedEx Express Saver (2–3 days)
8	$16	$18	FedEx Express Saver (2–3 days)
9	$16	$18	FedEx Express Saver (2–3 days)
10	$16	$18	FedEx Express Saver (2–3 days)
11	$20	$25	USPS Priority Mail
12	$20	$25	USPS Priority Mail
13	$20	$25	USPS Priority Mail
14	$20	$25	USPS Priority Mail
15	$20	$25	USPS Priority Mail
16	$20	$30	FedEx 2Day
17	$20	$30	FedEx 2Day
18	$20	$30	FedEx 2Day

Table 9: Shipping Fare Table by Volume, Fare Range, and Company

Multiple Simulation Results

After running the simulation, we obtain four instances to compare the success rates between both heuristics across the years. By extending the code from task 4, the simulation now outputs additional data:

1. **Locker Utilization**: Provides both annual and 5-year locker utilization statistics for both heuristics. This allows us to analyze how effectively lockers are being used over different time frames.

2. **Success Rates**: Shows annual success rates for each heuristic, helping us understand the performance of each approach over time. Additionally, the simulation captures daily success rates, offering granular insight into short-term effectiveness.

3. **Total Rejection Counts**: Tracks the number of failed locker assignments, highlighting instances where locker demand could not be met.

This extended output enables a comprehensive comparison between heuristics, assessing both their efficiency in locker usage and their capability to meet demand across various timeframes.

The below figures are the 4 instances of yearly demand:

```
Yearly demand scenarios: [60000, 70000, 75000, 80000, 90000]
```

Figure 23: Instance A

```
Yearly demand scenarios: [80000, 90000, 110000, 120000, 150000]
```

Figure 24: Instance B

```
Yearly demand scenarios: [50000, 65000, 60000, 75000, 65000]
```

Figure 25: Instance C

```
Yearly demand scenarios: [80000, 90000, 100000, 120000, 110000]
```

Figure 26: Instance D

The simulation output is saved in an Excel file, with each instance organized into sheets for easy analysis. Each sheet includes:

- **Assignment Log**: Records each locker assignment with details.
- **Success Rate**: Provides daily, annual, and overall success rates.
- **Locker Utilization**: Shows annual usage and a global summary over 5 years.

This layout enables quick comparison of success rates and utilization between heuristics and the 4 instances.

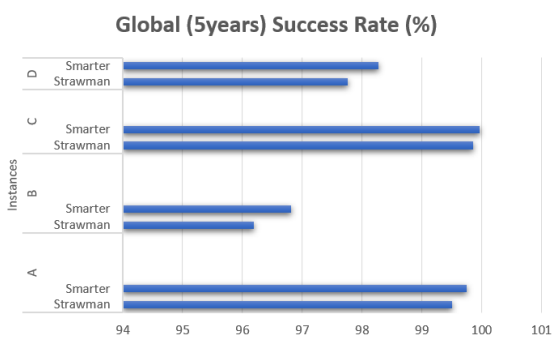

Figure 27: Locker Assignment Success Rate (5years)

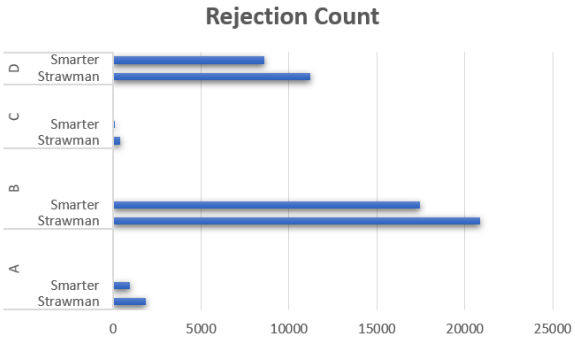

Figure 28: Number of rejections

Figure 27 tells us the success rate for 5 years for the instances A to D for both the heuristics. We can see that the *smarter heuristic* is better, and instance C has the highest success rate. Instance B, with the maximum demand of 150,000, has a success rate of 96.8% for the smarter heuristic, which is the lowest here but still performs well overall.

From figure 28 we can observe the rejection rate. Instances A and D have very low rejection of the packages, indicating that up to this demand level, we encounter no significant issues. We can optimize the locker assignment and sizes to reduce space requirements and further decrease rejection counts.

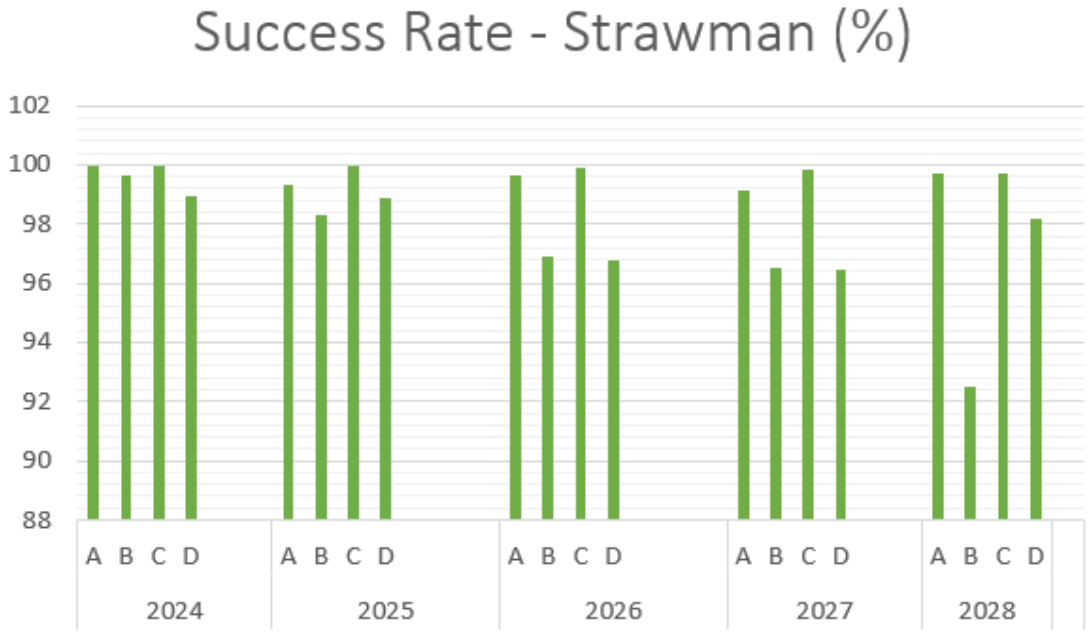

Figure 29: Annual Locker Assignment for Strawman Heuristic

We can observe the success rate of locker assignment for the strawman heuristic annually across the 4 instances in the Figure 29. It is noticeable that there is a drop in the success rate for the year 2028, which can be attributed to an increase in demand. This trend highlights potential limitations in the heuristic when demand rises.

The figure below displays the annual success rate of locker assignments using the smarter heuristic. We observe that the success rate for instances B and D dips below 98% between 2026 and 2028. To consistently achieve the target 99% service level, further optimization may be required. Nevertheless, the smarter heuristic demonstrates a higher success rate over the years compared to the strawman approach.

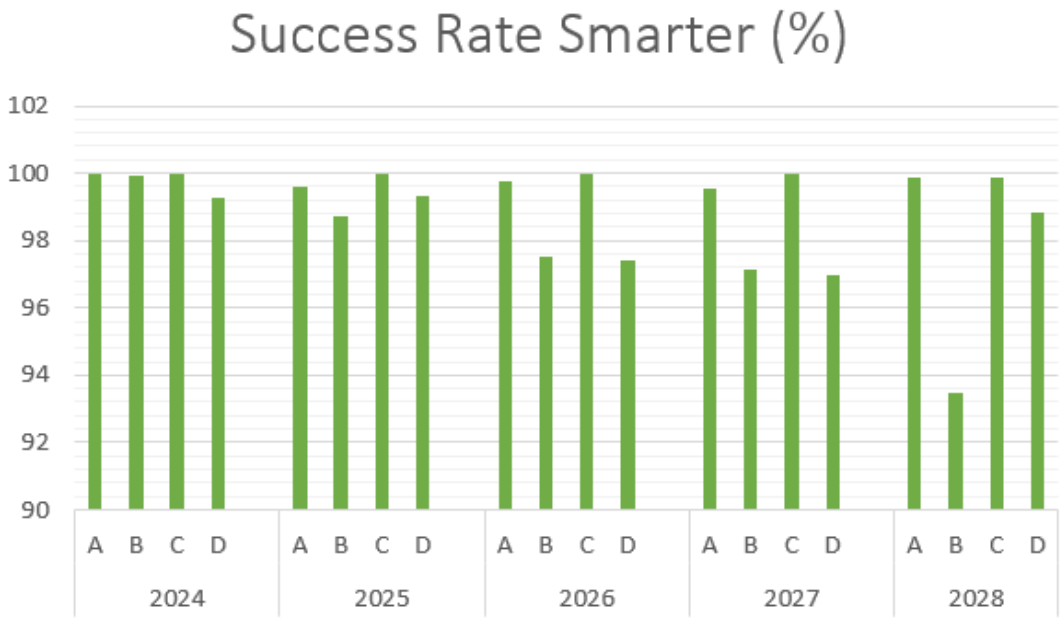

Figure 30: Annual Locker Assignment for Smarter Heuristic

Locker utilization indicates the occupancy level of locker size.We can see the locker utilization for the 18 sizes for the period of 5 years in the below figure. Both heuristics yield similar utilization rates, showing that at lower demand levels, utilization is around 50%. This suggests that for anticipated demand in 2026 and 2028, we might gradually increase the number of lockers rather than using a fixed number, aligning capacity more closely with demand.

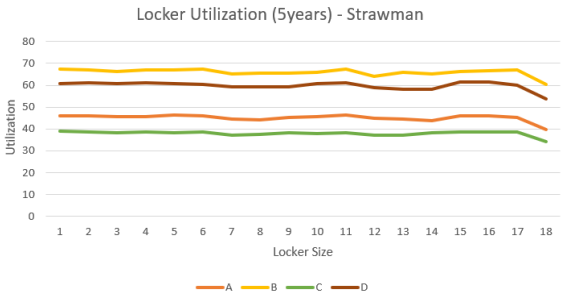

Figure 31: Locker Utilization(5years) - STrawman

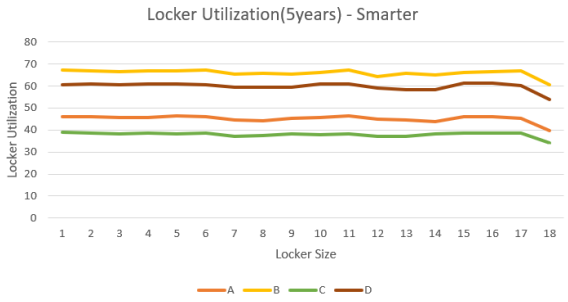

Figure 32: Locker Utilization(5years) - Smarter

Cost Analysis

Shipping costs

Using Table 9 and the rejection counts of each locker from the Excel output, we can calculate the shipping costs, applying the maximum possible cost for our representation. Figure 33 illustrates the costs for the strawman heuristic across the 4 instances over the 5-year period, while Figure 34 shows the shipping costs for the smarter heuristic. It is evident that the smarter heuristic generally performs better; however, higher rejection counts, as seen in instances B and D, lead to increased costs. By optimizing the locker layout, we can further reduce these costs effectively.

Figure 33: Shipping costs for Strawman for the 5 year period

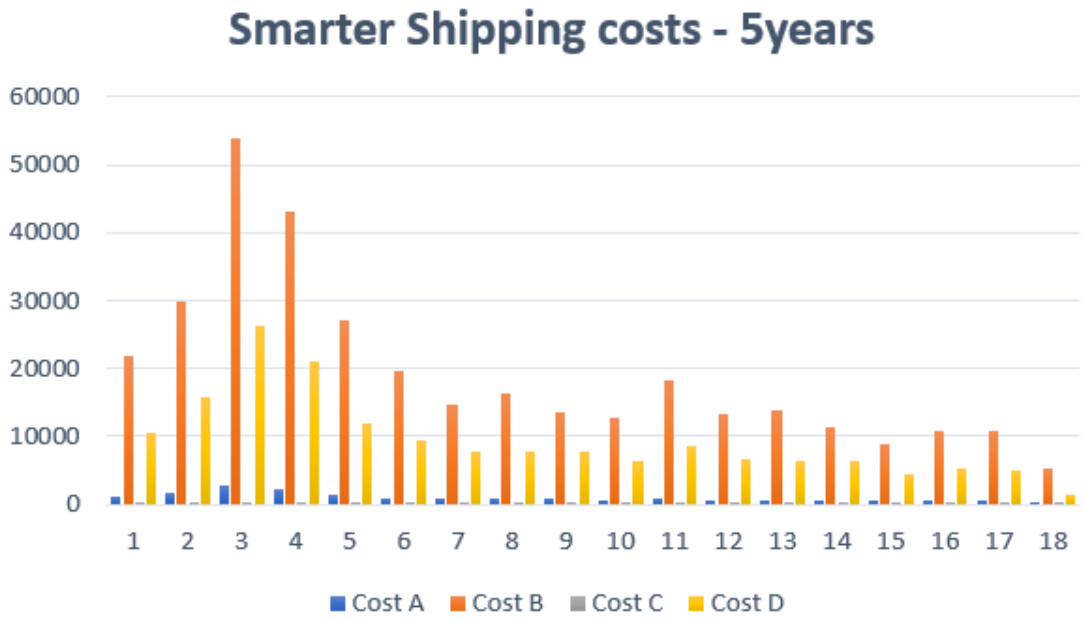

Figure 34: Shipping costs for Smarter for the 5 year period

We can additionally find the total shipping cost for each year for smarter heuristic in Figure 35. The chart presents annual shipping costs for four different demand instances from 2024 to 2028. The cost for the highest demand level sees a significant spike in 2028, exceeding $2,000,000. Moderate demand levels show gradual increases over the years, while the lowest demand consistently incurs the least cost. This distribution indicates that shipping costs escalate sharply as demand increases, especially in the final year.

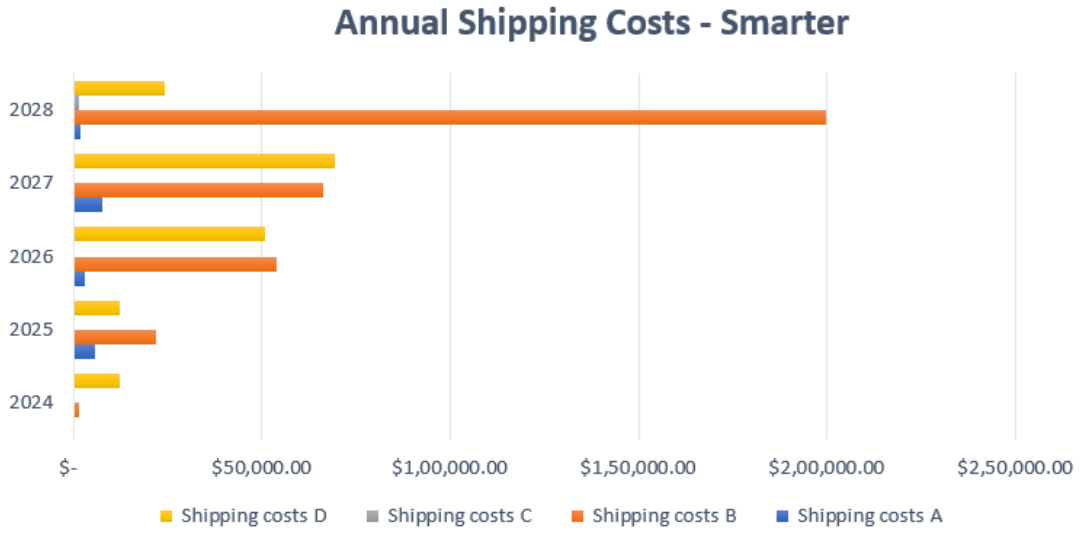

Figure 35: Annual Shipping costs for Smarter Heuristic

Annual Costs

Type of cost	Cost
Annual rental & maintenance	$6000
Installation	$1500

Table 10: Costs for Fixed Configuration

As we have 18 different locker sizes and from Figure 6 we can see that the total area is approx. $500\,\text{ft}^2$ and Table 15 gives us the costs. Hence we get the above table for the annual rent and an one-time installation cost.

Total Costs

Adding the shipping costs from Figure 35 and the costs from table 10 we get the following graph for the annual total costs. We can see that for instance B in the year 2028 has the highest cost owing to the immense shipping costs due to the unsatisfied demand.

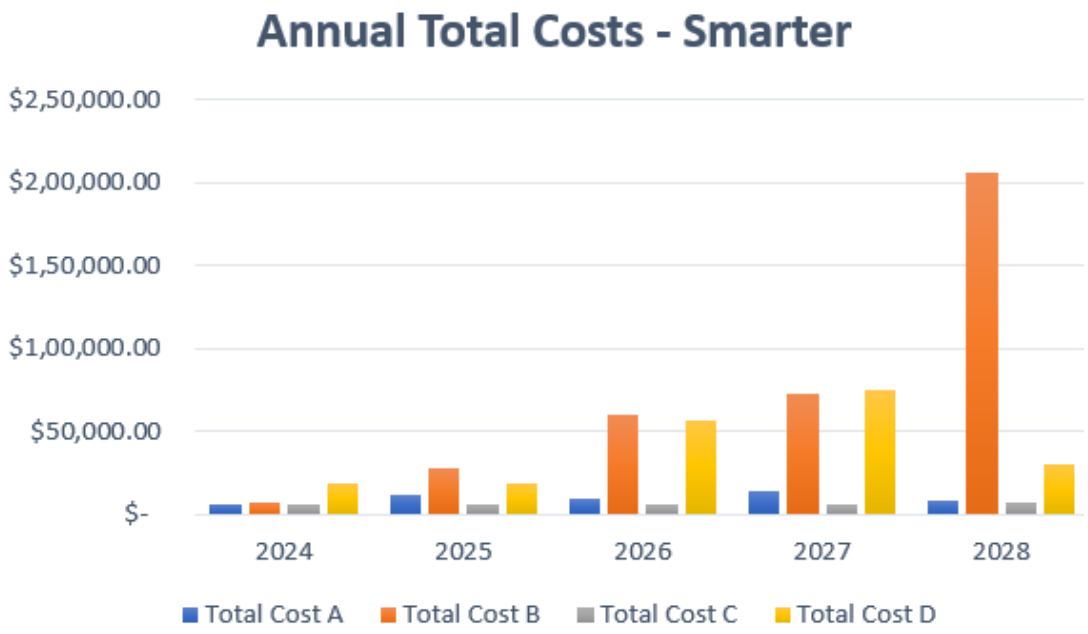

Figure 36: Annual Total cost for Smarter Heuristic

The below table gives us the detailed breakage of the costs for all the instances and the years. Figure 37 gives us the Total costs for the 5 years for all the heuristics and for the smarter heuristic which will be used to compare in the later tasks

Year	Total Cost A	Total Cost B	Total Cost C	Total Cost D
2024	$6,090.00	$7,303.94	$6,092.61	$18,233.89
2025	$11,641.41	$28,083.01	$6,153.14	$18,414.20
2026	$9,053.68	$60,007.35	$6,207.36	$56,899.69
2027	$13,753.50	$72,345.25	$6,474.10	$75,179.55
2028	$7,855.04	$205,874.18	$7,420.97	$30,358.82
Total	**$49,893.63**	**$375,113.73**	**$33,848.18**	**$200,586.14**

Table 11: Annual Total Costs for Each Instance

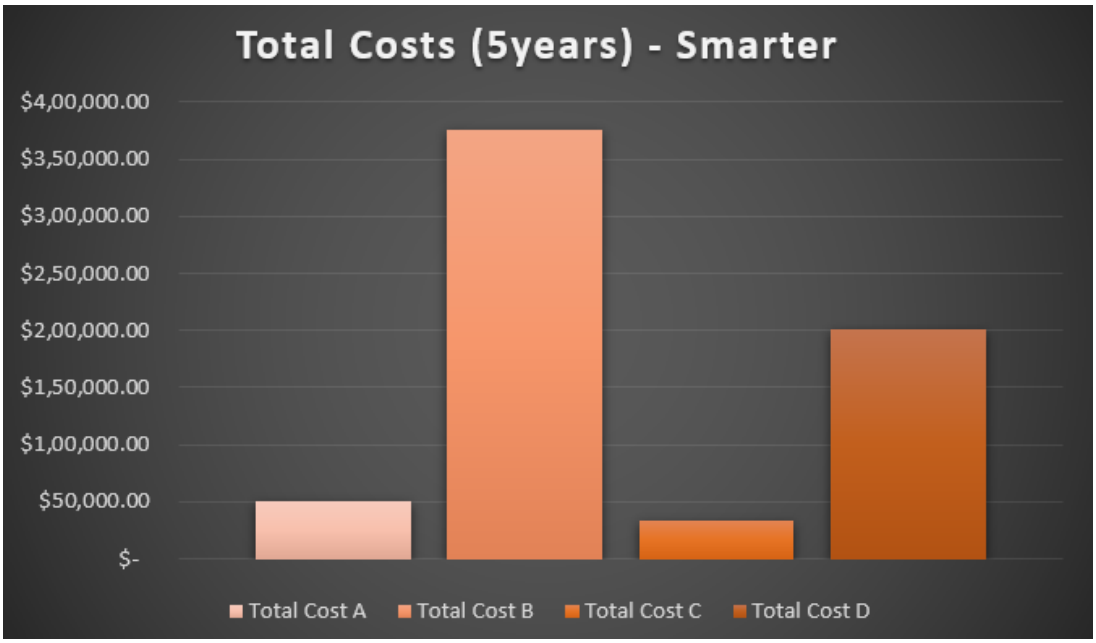

Figure 37: Total cost (5 years) for Smarter Heuristic

In this analysis, we evaluated the performance of the strawman and smarter heuristics for locker assignment over a 5-year period, focusing on metrics such as success rates, rejection counts, locker utilization, and shipping costs. Both heuristics demonstrated relatively high success rates, though the smarter heuristic consistently outperformed the strawman, particularly under high-demand scenarios. However, in instances with elevated rejection counts (like B and D), shipping costs increased, emphasizing the importance of minimizing rejections to control expenses.

Additionally, we observed that locker utilization was only around 50% at lower demand levels, suggesting that gradually adjusting locker quantities to match demand could enhance efficiency. Financially, the smarter heuristic offers potential savings, but further optimization of locker layout and assignment methods could further reduce costs and improve service levels.

TASK 7

In this task we will be trying to improve the basic configuration by attaining a 99% service level for each level of demand. Indeed, we have a service level of $\leq 99\%$ for the highest level of demand.

Phase 1

Determining Performance Targets

When it comes to determining performance targets, we have retained two major key values that will help us enhance our design:

- **Financial Target:** the overall yearly cost have to be taken into consideration. The objective is to minimize these costs, therefore minimize the number of lockers, while still maintaining a good service level.
- **Service Level Target:** There are some constraints to add to the minimization problem, the primary one is the base service level. Easy Node is looking to have a 99% service level. As we have seen in Task 5, this implies a great number of lockers, and a great amount of these are empty for most of the year. Furthermore, this amount of service level is not attained for the max demand scenario (even though this scenario has only a 4.3 % of chance of happening).Therefore, we need to modify the amount of lockers calculated in Task 1,using the Smarter heuristic implemented in Task 4, by minimizing the overall cost and still satisfying the service level constraints.

Implementing the Minimization Problem and heuristic

The decision variables for this problem are the following:

$$N_{L1}, N_{L2}, ..., N_{Li}, ..., N_{L18}, \forall i \in [1, 18]$$

The number of lockers required in the facility.
Let's also introduce:

- SL: the overall service level, which is the ration between: F (number of finalized orders) and T (total amount of request/ or package arrivals in the facility). SL is a function of the number of lockers, and depends on the heuristic that is used.
- R_s: the number of rejected packages of size s over the span of the experiment, which is the Total number of orders T, minus the finalized orders F. R = T-S.

R_s is a function of $N_{Li}, \forall i \in [1, 18]$: $R_s = R_s(N_{L1}, N_{L2}, ..., N_{Li}, ..., N_{L18})$ depending on the heuristic that is used.

The objective function is:

$$\min \sum_{s=1}^{18} R_s(N_{L1}, N_{L2}, ..., N_{Li}, ..., N_{L18}) \cdot RC_s$$

With:
RC_s: the rejection cost depending on the size of the package. The constraints are:

- $SL(N_{L1}, N_{L2}, ..., N_{Li}, ..., N_{L18}) \geq 0.99$
- $N_{L1}, N_{L2}, ..., N_{Li}, ..., N_{L18} \in \mathbb{N}$

This is a Integer Program with non linear objective function and constraints. Therefore is pretty hard to solve.

Phase 2: Implementing the heuristic and running the code

Therefore, we may consider a simpler approach by only considering the Service Level target. The objective would be to find the minimum number of lockers to satisfy $SL \geq 0.99$. To do so, we will be defining a derived heuristic approach:

1. Consider the highest demand of 150,000 lines (2028), which has the lowest SL. If we manage to bring $SL \geq 99\%$, our model will be able to sustain any scenario.
2. Run the program with the current number of lockers.
3. Create a loop:
    - evaluate the current rejection rate (over a year)
    - note the lockers who have the highest rejection rate
    - add +1 to the number of these lockers
    - run the code once again and repeat the loop until 99% SL is achieved

This should return a quasi optimal number of lockers that are required to satisfy a SL of .99.
Even though this heuristic does not take into account any financial parameters compared

to the first method, it is way easier to implement. Furthermore, the financial data that we used for this casework has been estimated by our team, therefore is probably inaccurate. We run the code for some time and the results are presented below.
Let's note that a much more precise heuristic can be computed by adding 1 to Locker LX at each iteration and once the 99 % SL is attained go to the next Locker Group LX+1. This heuristic is probably more space efficient as it would increase the Lockers with lower volumes first. However the time required to run this code is extremely long (at least 18× *more than the one previously described, which is the reason why we decided to keep the code as it is.*

Phase 3 : Results and analysis

Based on the previous phase we obtain the following results after 14 iterations:

Iteration #	**Overall SL (%)**	**Rejection Rate (%)**
1	93.85	6.15
2	94.64	5.36
3	95.31	4.69
4	95.88	4.12
5	96.39	3.61
6	96.81	3.19
7	97.50	2.50
8	97.73	2.27
9	97.92	2.08
10	98.25	1.75
11	98.63	1.37
12	98.76	1.24
13	98.93	1.07
14	**99.01**	**0.99**

Table 12: Iterations with Overall Service Level and Rejection Rates

The evolution of locker counts are shown below:

As we can see in the table and in the figure, it takes a lot of additional lockers just to bump the yearly Service Level (2028 - 150,000 packages) from 94 % to 99 %.

Locker Type	Initial Locker Count	Final Locker Count
L1	36	47
L2	54	77
L3	68	90
L4	54	76
L5	36	55
L6	27	42
L7	23	34
L8	23	36
L9	23	34
L10	18	29
L11	18	29
L12	14	21
L13	14	21
L14	14	22
L15	9	16
L16	9	16
L17	9	16
L18	5	13
Total	454	674

Table 13: Initial and Final Locker Counts by Locker Type

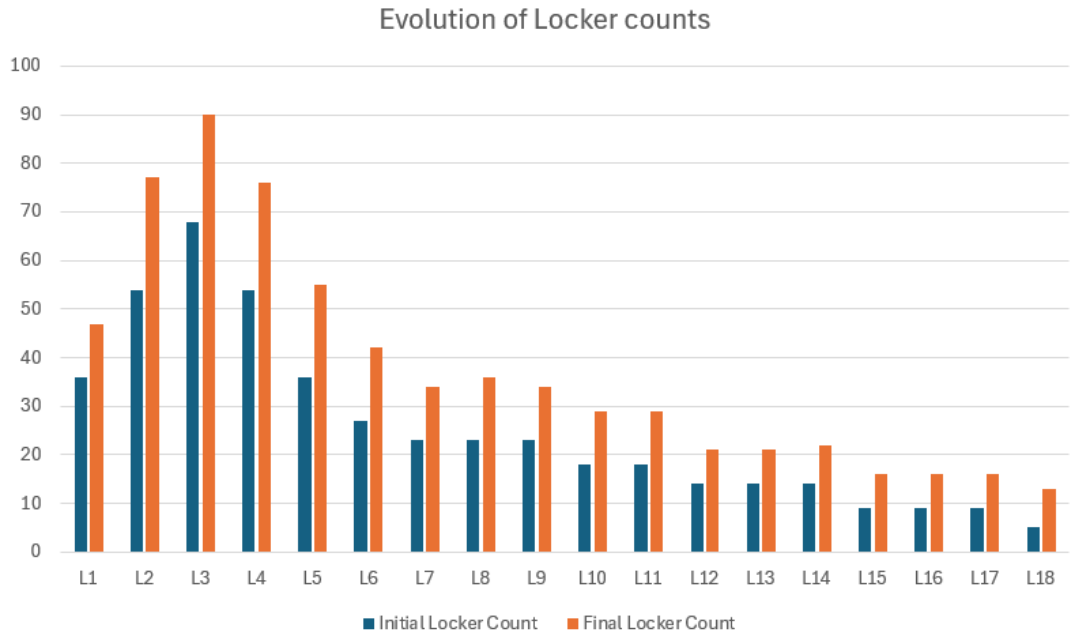

Figure 38: Additional Locker Requirements

Furthermore, we can visualize the decrease of the rejection rate throughout the iterations. Let's note that when the rejection rate for a locker falls beneath 1 %, we stop adding lockers to the Locker Group. The packages that required the most iterations before reaching the $\leq 1\%$ rejection rate are Locker Groups L2, L3, L4, and L5. There are also the same groups that required the most Lockers.

Figure 39: Evolution of the RR over the iterations

As we can see the tendency is pretty linear. This is very different from the larger locker size, which rejection rates tend to decrease following an inverse log trend.

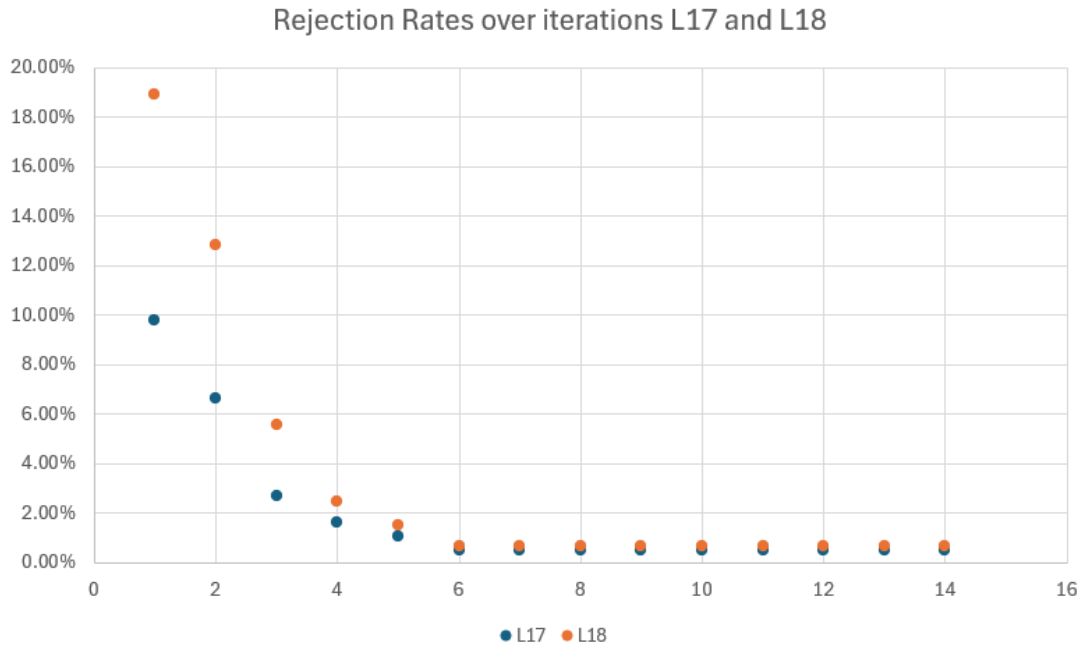

Figure 40: Evolution of the RR over the iterations L17 and L18

Conclusion The disparity of the demand throughout and over the years has lead to a great amount of uncertainty regarding the upcoming demand. If we want to take into consideration each possible outcome while still assuming a 99 % service level, this might require too many lockers, therefore resulting in potential loss due to non utilization. If we had more precise data, especially regarding the financial aspect of this Casework, we may be able to apply optimization process such as the one detailed in Phase 1, this would have helped us regarding our final decision.

Our final decision: If we want to assume a 99 % service level throughout the 5 years, while keeping a fixed locker configuration, it appears that we would have great losses in space utilization and on the financial side. For this reason, we would recommend to add additional lockers once every year or two, depending on how the demand evolves. This will progressively lead us to abandon the idea of a fixed locker configuration.

TASK 8

Defining the locker sizes

As the casework goes on, we have found out that having 18 different sizes to save 6% space might not be worth it and make the task a bit complicated. So we will reduce the number of sizes we will provide. To do so, we have computed the overall number of packages of each size that will come in the bank over one year. The results are plotted in figure 41.

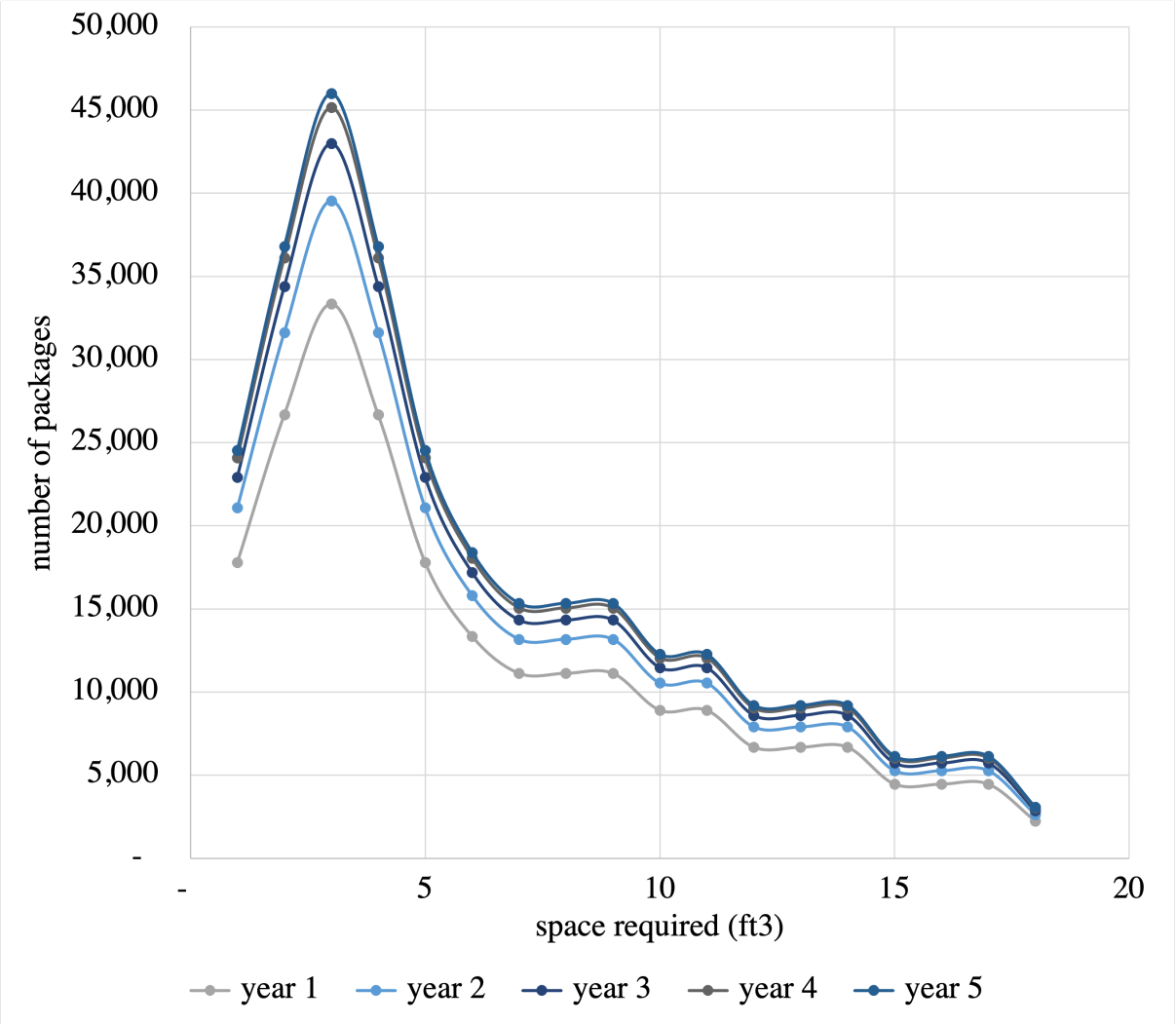

Figure 41: Total number of packages over the years entering the locker bank

This graph shows that we have a lot of packages that have a size inferior to 6 ft^2. So we need to have various sizes for all these packages so as to minimize space. We have more than 6 times less packages of size bigger than 10 ft^2 so we can have a wider range of locker size for these parcels. This is why we propose to have the following sizes of lockers,

displayed in table 14.

	x	y	z
L18	3	2	3
L15	3	2	2.5
L9	3	2	1.5
L6	3	2	1
L3	1.5	2	1
L2	1	2	1

Table 14: Locker sizes

In this case, we paid attention at the width (x) of the lockers so that we can stack them more efficiently. Indeed, they are all 3 feet wide or can be put together so as to form a 3 feet wide set of lockers.

Defining the lockers towers sets

To determine our set of lockers towers, we will look at the percentage of the demand that goes into each type of locker. We will create a set of lockers that matches this demand (figure 42).

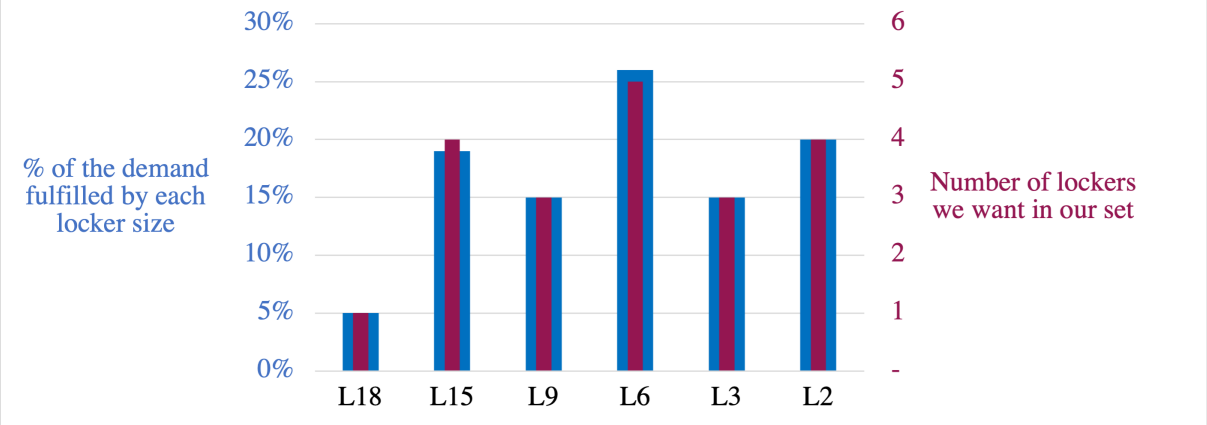

Figure 42: Percentage of the demand fulfilled by each locker size

In this case, the lockers of size 18 fulfill 5% of the demand. Let's say we want one locker of type L18 in our set then. L15 lockers fulfill 20% of the demand so we want 4 time more lockers of size L15 in our set and so on for the other sizes.

This has lead us to propose the following set of 4 towers displayed in figure 43.

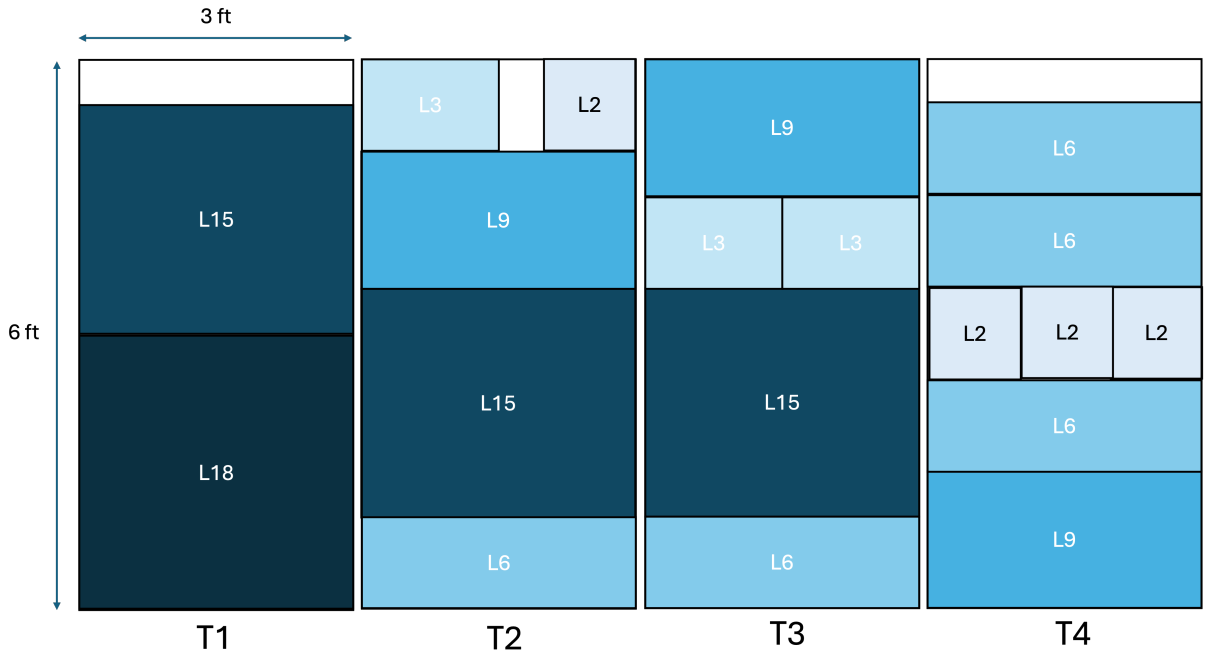

Figure 43: Towers of lockers set

We note that this design makes our towers pretty compact with only 3% of the overall volume being unutilized and offering a nice flexibility over the periods.

TASK 9

In a dynamic configuration, we will be able to adjust the number of lockers to the demand over each 4-weeks period. To do so, we will compute the number of lockers we will require for each time period. This will correspond to the maximum parcels being held at the same time in the bank during the 4 week period. The results are displayed in figure 44.

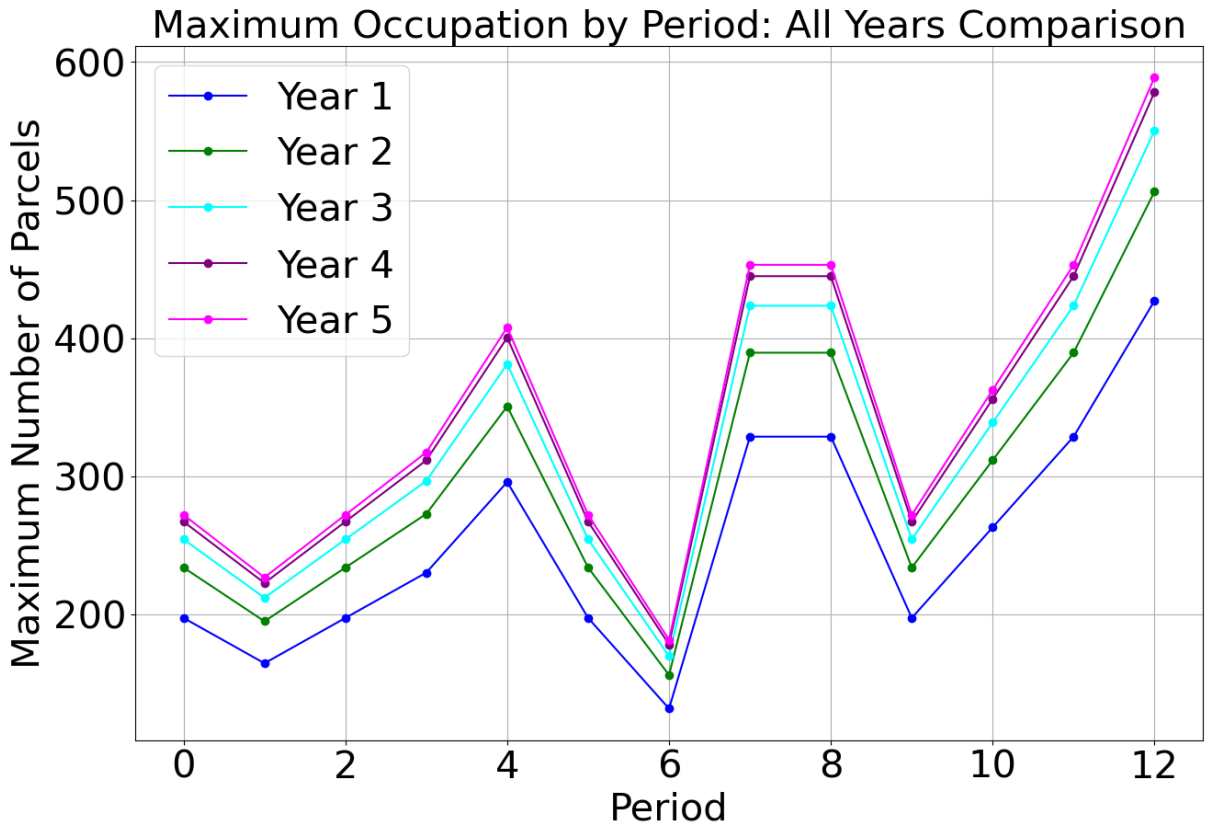

Figure 44: Maximum number of parcels being held at the same time in the locker bank during each 4-week period

We will iterate through each period and compute the number of towers of each type we will need during each period. We will count the number of lockers of each size we will need and use this result to determine the number of towers of each type we need and will do that minimizing the total number of towers so as not to have too many unused lockers.

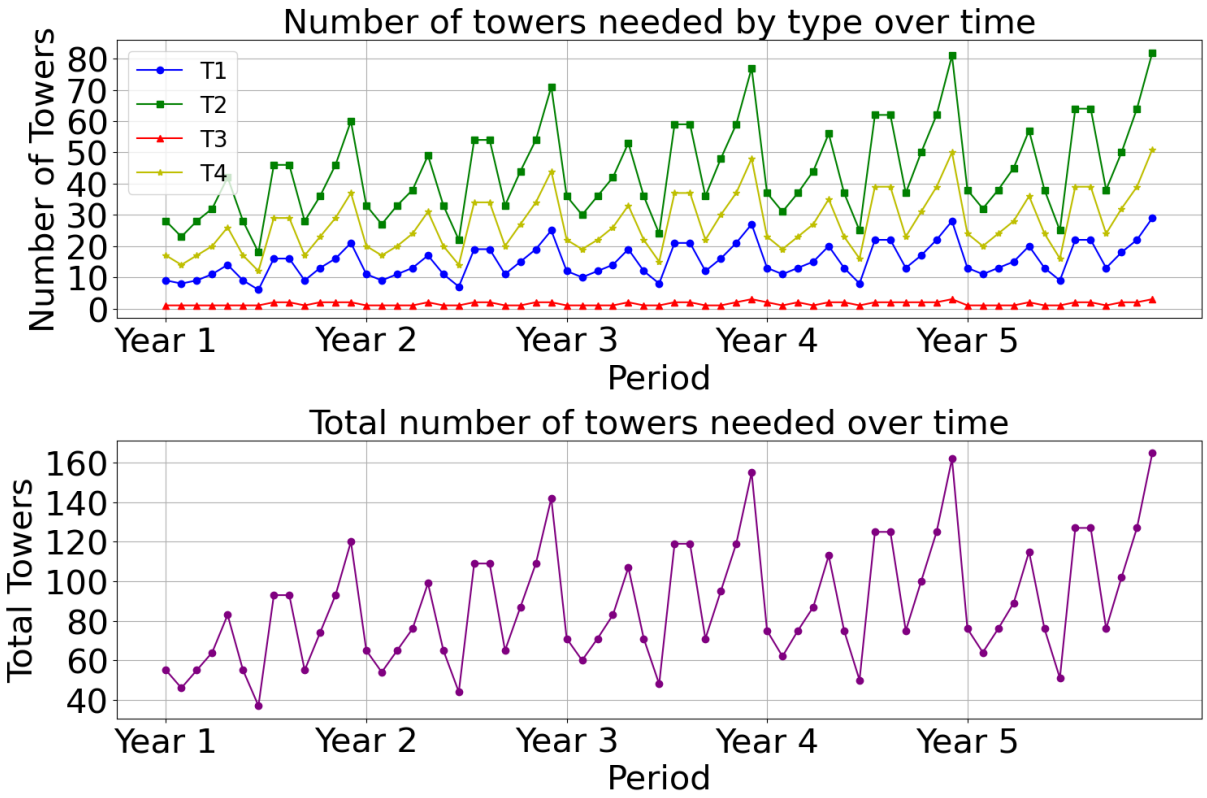

Figure 45: Number of towers of each type required during each period

Figure 45 shows the number of towers we need during each period. We see that the demand is pretty versatile which is an argument in favor of having dynamic lockers. We notice that tower of type T3 is not used very often while T2 is the most used. This is due to the fact that in order to have L3 lockers we can only have T2 or T3 and T2 is more adaptable as it proposes almost all sizes of lockers which is why this set is used more often.

So over the years and over the periods, the number of parcels entering the bank varies. So we will adapt the number of each of our towers T1, T2, T3, T4 we will have for each period and evaluate the costs. For the costs we will consider as in the previous tasks the followings:

type of cost	cost	unit
rental & maintenance	1	\$ per period per ft^2
installation	2	\$ per ft^2
removal	3	\$ per ft^2

Table 15: Costs

We have iterated through the years to determine the number of towers we will need to add and/or remove across every period and computed the corresponding costs. We note that one tower has a floor area of $2 \times 3 = 6ft^2$.

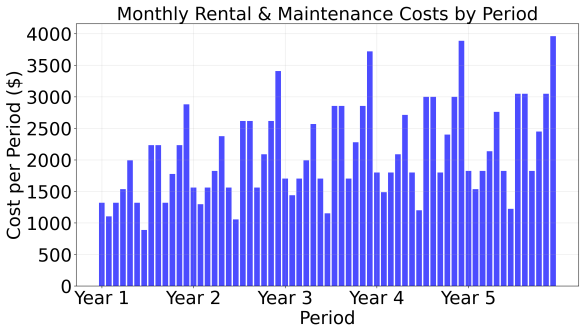

Figure 46: Rental and Maintenance costs

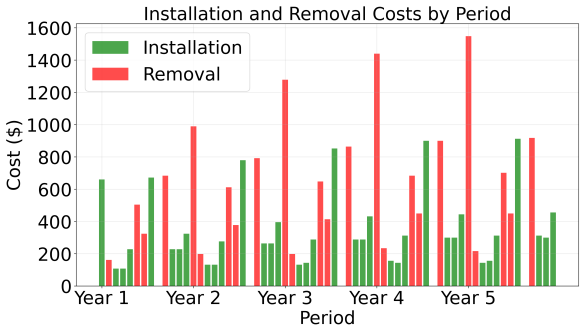

Figure 47: Transition costs

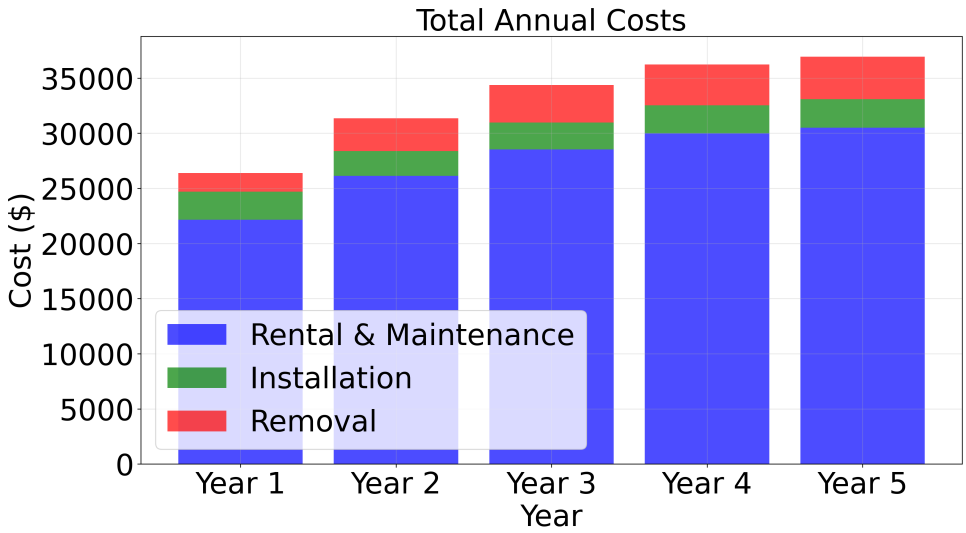

Figure 48: Total annual costs

Having this modular system has allowed us to decrease significantly the initial space cost. Over the years, installation and removal costs are way lower than rental costs. This is due to the fact that we actually don't need to move so many lockers every period and also to the fact that our transition costs are pretty low which might not necessary be the case in reality.

TASK 10

Performance Evaluation

The below figure shows the success rate for the modular configuration. The modular locker system was meticulously designed to achieve a 98.9% demand fulfillment rate over the 5-year simulation period. This objective was met through the dynamic allocation of locker resources, ensuring optimal utilization of the modular locker banks. From a total demand of 125,450 packages, 98.8% of parcels were successfully assigned to lockers, equating to 99,990 packages. This demonstrates the system's robustness in meeting high demand levels. Conversely, only 1.18% of packages, or 1,491 packages, were rejected due to insufficient locker capacity, highlighting the minimal impact of capacity constraints on the overall service level. These results emphasize the efficiency and reliability of the modular locker configuration in sustaining high service performance across varying demand conditions.

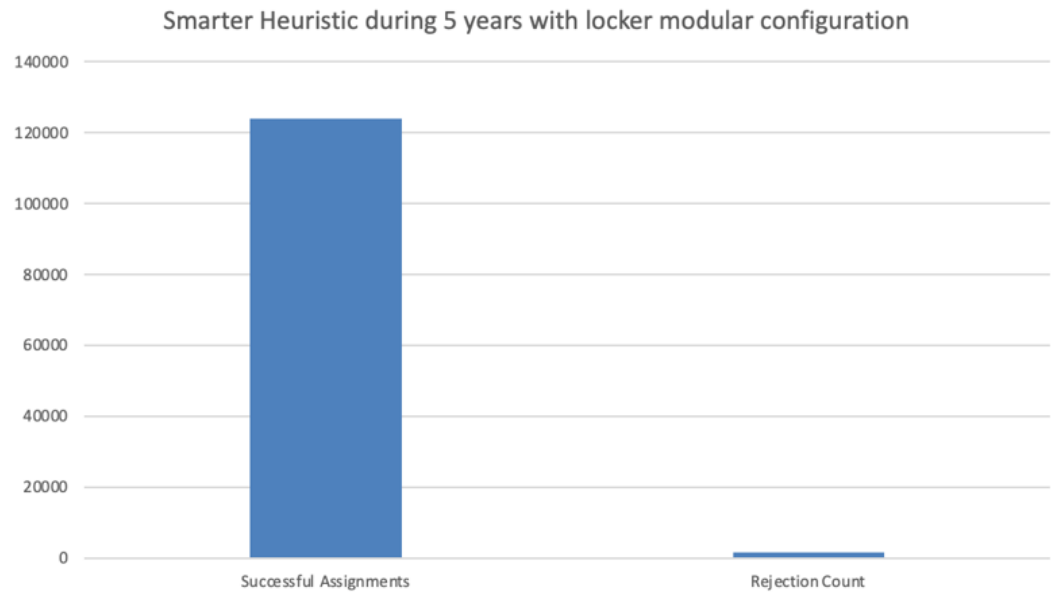

Figure 49: Modular Configuration with Smarter Heuristic success rate (5 years)

Comparison with Previous Assignation Logic

When comparing the modular locker assignment results to the smarter heuristic developed in Task 5, it becomes evident that the utilization, rejection, and assignment rates are quite similar. Both strategies demonstrate high efficiency in meeting demand, with minimal rejection rates and optimal locker usage. However, the key distinction between

the two approaches lies in the cost evaluation. While Task 5's heuristic provides effective demand management, the modular locker system introduces significant cost advantages by optimizing rental, maintenance, installation, and removal expenses. This highlights the modular system's added value in delivering comparable operational performance while achieving greater cost efficiency.

Locker Utilization

Utilization by Locker Type and Size

The modular locker system demonstrates a well-balanced utilization across various locker types, as shown in the graph. L6 lockers, found in multiple tower types (T2, T3, and T4), have the highest utilization, accounting for approximately 25% of total assignments. This highlights the system's ability to efficiently handle medium-sized packages, which appear to dominate the demand. Similarly, lockers like L2 and L15, which represent 20% and 15% utilization respectively, show steady demand for both small and larger medium packages. The consistent usage of these sizes across towers like T2 and T4 underscores their importance in meeting the majority of parcel storage requirements.

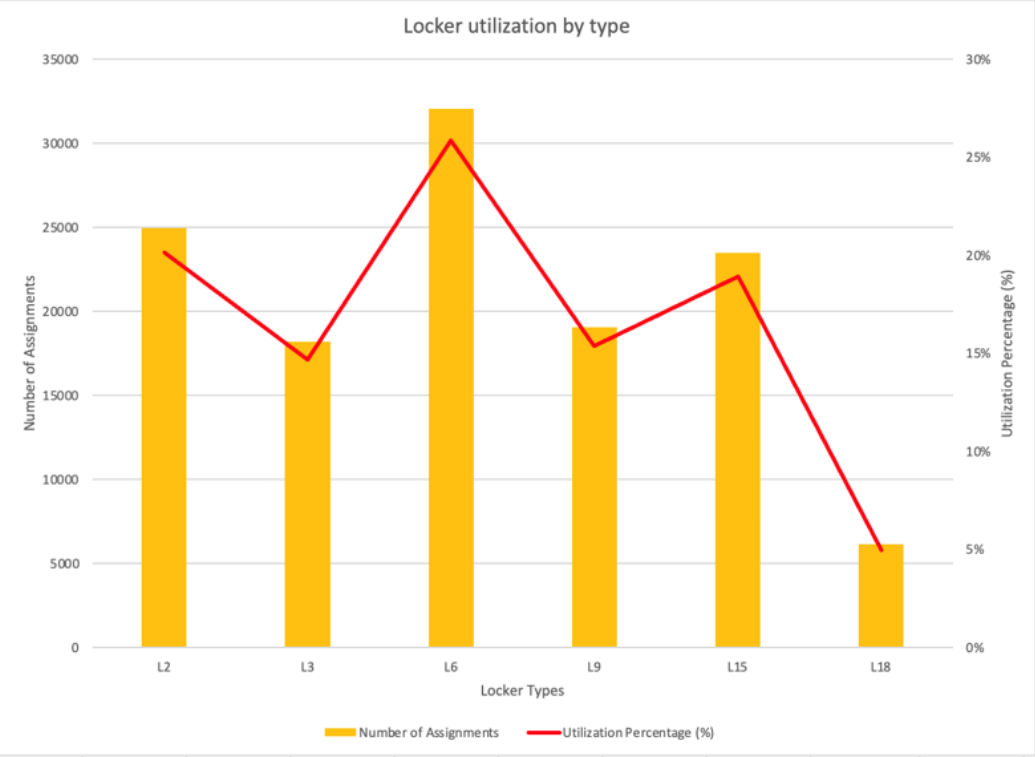

Figure 50: Locker Utilization - Modular Configuration

Conversely, the L18 lockers in T1 towers show the lowest utilization at 5%, indicating that large packages are less frequent. This highlights an opportunity to optimize the allocation of T1 towers in areas where smaller or medium-sized lockers, such as L2 or L9, might be more beneficial. Despite this, the modular design provides flexibility, allowing the system to adjust locker configurations based on demand patterns. Overall, the distribution of assignments across tower types and locker sizes ensures that the system remains adaptable and efficient, with each tower type contributing uniquely to demand fulfillment.

Service Level Analysis

Achieving a service level of 99.8% underscores the effectiveness of the modular locker system in minimizing rejections and optimizing pick-up times. By ensuring that nearly all parcels are assigned lockers promptly, the system significantly reduces instances of failed deliveries, which is a key driver of customer satisfaction. Additionally, the streamlined pick-up process, facilitated by the modular configuration, minimizes waiting times and enhances the overall user experience. This high level of service reliability fosters trust and confidence among customers, positioning the locker system as a dependable solution for managing parcel deliveries efficiently.

Scalability and Robustness

Handling Future Demand Growth

The modular locker system is inherently designed to adapt to fluctuating demand, making it well-suited to handle potential future growth beyond the 5-year simulation period. Its flexibility allows for the seamless addition or removal of towers to match evolving demand levels. Should demand increase in the future, the system can be expanded by incorporating additional modular towers without disrupting operations or compromising service levels. This adaptability ensures that the locker network remains effective and efficient, maintaining high performance even as demand scales, thereby future-proofing the solution against long-term growth scenarios.

Stress Testing

The system's robustness was evaluated under extreme demand conditions, with the highest demand observed in Year 5. Even during this peak period, the modular locker design successfully supported 99.8% of the demand, maintaining exceptional service levels. This

demonstrates the system's ability to handle unexpected surges without compromising performance. The results affirm that the modular configuration is not only effective under normal operating conditions but also resilient during periods of heightened demand, ensuring a reliable and successful process regardless of fluctuations.

Heuristic Validation

Effectiveness of the Heuristic

The heuristic implemented in the modular locker system builds on the principles established in Task 4, employing a smarter approach to locker assignment. It prioritizes finding an exact match for each parcel's size requirements, and if an exact match is unavailable, it dynamically searches for the next larger locker size. This flexibility significantly reduces rejection rates by ensuring that parcels are accommodated even when the preferred locker size is fully occupied.

Decision Dynamics

Moreover, the heuristic adapts its decisions in real-time based on current locker availability and forecasted demand. By continuously monitoring locker usage and anticipated demand fluctuations, it optimizes the allocation process, maintaining high service levels and minimizing rejections. This dynamic decision-making capability ensures that the system remains responsive and efficient, even under varying demand conditions, thereby maximizing the effectiveness of the modular locker configuration.

TASK 11

Fixed-Configuration Locker Bank

In the fixed configuration, the primary objective was to achieve a 99% service level even under peak demand conditions. To meet this target, the number of lockers had to be significantly increased from an initial count of 454 to 674. This setup ensured that the system could handle high demand scenarios, such as the 2028 peak of 150,000 packages. However, this approach had substantial drawbacks in terms of cost and efficiency.

The fixed setup guaranteed the required service level during peak demand but led to high operational costs due to the underutilization of lockers during off-peak periods. Many of the additional lockers, especially sizes L2, L3, and L4, remained unused for much of the year, driving up costs unnecessarily. This inefficiency highlighted the financial burden of a fixed configuration that cannot dynamically adjust to changing demand.

Another drawback was the low utilization of lockers for most of the year. Although designed to handle rare high-demand events, the fixed configuration led to wasted space and financial inefficiency during normal periods. This underlines a critical limitation of fixed setups—they lack the flexibility to adapt to fluctuating demand, resulting in resource wastage when demand is lower than peak scenarios.

Moreover, the fixed configuration approach required overprovisioning to account for demand uncertainty. While this ensured readiness for peak periods, it also created inefficiencies during regular operations. The need to allocate many lockers for rarely occurring high-demand events demonstrated the system's inability to optimize resources cost-effectively.

Dynamic Modular Configuration

A dynamic modular locker system was introduced, offering a more adaptable approach to locker management. This system allowed for monthly reconfiguration using the heuristic from Task 9, which dynamically adjusted the number of towers based on demand. The modular system consistently maintained a 98.8% service level, balancing demand fulfillment and resource optimization.

One key advantage of the modular system was its efficient handling of demand. By adjusting the number of towers monthly, the system avoided overprovisioning during low-demand periods. This adaptability ensured effective resource allocation, reducing operational costs associated with maintaining unused lockers during off-peak times.

The modular design also provided significant flexibility and cost efficiency. Unlike the

fixed configuration, which maintained a constant number of lockers, the modular system enabled real-time adjustments. Towers could be added or removed as needed, ensuring high utilization rates throughout the year. This flexibility minimized resource wastage and lowered costs substantially compared to the fixed setup.

The modular system also demonstrated excellent scalability for future demand growth. As demand increases, additional towers can be seamlessly integrated or existing towers reconfigured to meet new requirements. This scalability ensures that the system remains efficient and cost-effective, even as demand evolves, without excessive upfront investment.

Key Comparisons

Metric	Fixed Configuration	Dynamic Modular
Service Level	99%	98.8%
Total Lockers Needed	674 lockers	Dynamically adjusted monthly
Peak Utilization	Low during off-peak periods	High throughout all periods
Rejection Rate	1%	1.2%
Cost Efficiency	High costs due to underutilization	Lower costs due to dynamic allocation
Scalability	Limited, requires manual adjustments	Highly scalable and flexible

Table 16: Comparison of Fixed Configuration vs. Dynamic Modular Locker Systems

Cost Analysis: Modular vs. Fixed Configuration

The cost comparison between the modular locker system and the fixed configuration provides valuable insights into the financial efficiency of both approaches over the 5-year simulation period.

Modular Configuration Costs

The modular locker system dynamically adjusts the number of towers each month, optimizing costs based on real-time demand. The breakdown of the costs includes:

- **Rental and Maintenance Costs:** These represent the primary recurring expenses, varying by period but generally kept efficient due to dynamic resource allocation.

- **Transition Costs (Installation and Removal):** These costs arise from reconfiguring the system to match changing demand. They are relatively low compared to the fixed configuration as the system minimizes unnecessary transitions.

The total cost for the modular system over five years amounts to $49,893.63, as shown in Figure 47, with consistent savings in both peak and off-peak demand periods.

Fixed Configuration Costs

The fixed configuration incurs significantly higher costs due to its inability to adapt dynamically:

- **Consistent Overprovisioning:** The system maintains excess capacity even during low-demand periods, leading to high rental and maintenance costs.
- **Inefficient Utilization:** A large portion of lockers remains unused for much of the year, further inflating costs.

The total cost for the fixed configuration, based on the average of the four cost scenarios, is $164,860.42, which is still significantly higher than the modular system, being more than 3 times the total cost of the modular configuration.

Key Insights and Comparison

Metric	Modular Configuration	Fixed Configuration
Total Cost (5 Years)	$49,893.63	$164,860.42
Rental & Maintenance Cost	Lower due to dynamic allocation	High due to constant overprovisioning
Transition Costs	Controlled with minimal transitions	Nonexistent but overshadowed by rental costs
Cost Efficiency	High	Low

Table 17: Cost Comparison of Modular Configuration vs. Fixed Configuration

Conclusion

While both systems meet the required service level, the modular configuration is far more cost-effective. The fixed configuration's inability to adapt to changing demand leads to excessive costs and inefficient resource use. Therefore, the modular system not only delivers superior service levels but also ensures financial sustainability over the long term.

TASK 12

Executive Summary for EasyNode

This report presents a comprehensive evaluation of EasyNode's locker bank configurations, focusing on the dynamic modular system developed in Task 10 and its comparison with the fixed-configuration setup from Task 7. The analysis spans a five-year horizon, highlighting critical insights, performance outcomes, and recommendations for optimal locker bank management.

The modular locker system, leveraging a dynamic monthly reconfiguration approach, achieved a service level of **98.8%**, slightly below the 99% target but with significantly lower costs. The system dynamically adjusted the number of locker towers based on real-time demand, maintaining high locker utilization throughout the year. Over the five-year period, the total cost for the modular configuration was **$49,893.63**, a stark contrast to the fixed configuration's cost, which averaged **$164,860.42** across different scenarios. This represents a cost saving of more than **70%**, making the modular system far more financially efficient.

In comparison, the fixed locker bank setup was designed to meet a 99% service level under peak demand scenarios. However, this configuration required a significant increase in lockers, from **454 to 674**, to accommodate peak demand, leading to excessive costs. The fixed configuration achieved its service level target but at the expense of underutilizationduring most of the year. As a result, the total cost ballooned due to constant rental and maintenance expenses, even when lockers were idle. This inflexibility highlights a major drawback of the fixed setup: its inability to adapt dynamically to changing demand, leading to inefficiencies and high operational costs.

The modular system demonstrated superior scalability and flexibility, seamlessly handling fluctuating demand and efficiently accommodating future growth. By allowing towers to be added or removed as needed, the system ensured that resources were neither over- nor under-utilized. This adaptability minimizes rejection rates, reduces customer dissatisfaction, and significantly lowers rejection-related costs. In contrast, the fixed configuration faced higher rejection rates in earlier iterations, only achieving optimal service levels after multiple adjustments, which increased costs and complexity.

Financially, the modular system excelled in controlling both rental and transition costs. The dynamic nature of the system reduced unnecessary expenses during off-peak periods by optimizing tower allocation. Transition costs, involving installation and removal of towers, were managed effectively, further contributing to the system's cost-efficiency. The fixed configuration, while avoiding transition costs, incurred disproportionately high rental and maintenance expenses, leading to a total cost more than **three times** higher than the modular system.

Operationally, the modular system ensured consistently high **locker utilization**, aligning resource availability with demand. The system dynamically reallocated resources, maintaining service levels even during peak demand periods. Conversely, the fixed system exhibited significant resource wastage during off-peak times due to its static nature. This inefficiency, combined with its higher costs, underscores the fixed configuration's limitations in managing dynamic demand environments.

Recommendations

Based on these findings, EasyNode should adopt the modular locker system as its primary configuration strategy. This system offers superior cost efficiency, operational flexibility, and scalability, ensuring sustained high performance and customer satisfaction. Additionally, EasyNode should implement **monthly performance reviews** to fine-tune locker configurations and proactively respond to demand fluctuations. This will further enhance the modular system's efficiency and ensure seamless adaptation to future demand growth.

In conclusion, the modular configuration aligns perfectly with EasyNode's goals of maximizing efficiency, minimizing costs, and maintaining high service standards. By adopting this approach, EasyNode can confidently meet current and future operational demands while optimizing financial performance.

TASK 13

Throughout this casework, we have been able to combine data from Python and Excel so as to reach our objectives. Indeed some people from the team are keener on Excel so we need to be able to use both tools so as to move forward.

Our work on heuristic optimization to minimize costs was particularly insightful. Tackling the challenge of yearly scheduling taught us to align production and resources with fluctuating demand efficiently.

Data visualization also became one of our strong suits. We already had some good knowledge of Plotly, Matplotlib, or Excel. But we have decided to improve dynamic visualization using the Tkinter library on Python to create nicer interfaces.

On the organizational and financial side, our approach to decision-making was heavily guided by key performance indicators. We explored a variety of scenarios with Monte Carlo simulations to understand how different variables could impact our outcomes, giving us a solid foundation for decision-making even in the face of uncertainty.

Overall, one of the most significant learning from the casework was understanding how to buffer against volatile demand through strategic and adaptable solutions. We recognized the critical importance of accounting for demand seasonality, which played a pivotal role in shaping our approach. A key insight was the value of implementing dynamic locker systems to minimize costs. By designing lockers that could adapt to varying demand patterns, we explored configurations that could flexibly accommodate fluctuations, ensuring efficient use of space and resources. Nevertheless, an efficient dynamic locker bank requires to know pretty precisely the demand since we can't add more lockers within an hour. To compensate this, we might need to still have a safety margin so as not to reject too many parcels.