Name: Vansh Dalal Reg. No: 21BCE0635

Elevator Tracker

A problem faced in everyday life is when someone goes to use an elevator (lift) one must spend quite a bit of time waiting. In fact it can even cause considerable delays depending on the situation, making a student late to an exam, a businessman late to a meeting or an interviewee late to an interview. Even if it is not a situation like those mentioned previously it can often be an inconvenience, reaching the lift and seeing you just missed it, or even if you do manage to get to the lift on time, it's too full. The goal of this website is to provide a way for its users to enter their details and accordingly the page will display the current floor the elevator is on and whether it is going up or down so the user can plan when to leave accordingly. Another feature would be that the system will also take data from the elevator of the weight it is carrying and transmit that data so the user of the website will have access to additional information of how full or empty the elevator is. As of now the scope of this project is limited to the VIT Vellore Mens Hostel blocks for its initial implementation on a small scale for the prototype development and testing.

Currently after extensively researching the current market there are various applications which act as an elevator management system, from leading companies such as Schindler which uses the Ahead ActionBoard tool, it helps keep track of maintenance schedules, it acts as a direct point of contact and helpline to the manufacturers or the parent company. It also acts as a reminder service using notifications and alerts to remind property managers or owners to contact their service providers for the obligatory maintenance and safety standards that should be held. Some other services also include a remote to power down an elevator in case of a malfunction or an emergency. Another service provided by Hyundai allows for various services mentioned above such as informing security in case of a malfunction, logging data for future reference in case of any emergency and auto contacting a technician in case of passengers being locked in the elevator. Hyundai also provides clients to track elevators remotely which is the purpose of this project and the solution to the problem statement. This project however will differ from Hyundai's system as their system has a pre-requisite. This pre-requisite is that a Hyundai Smart Elevator System must be installed. This project will allow remote elevator tracking for any elevator with a digital screen showing which floor the elevator is on. Further this system will add the additional feature of displaying how full the elevator is allowing the user to know if another person can fit and it is much more open to being adapted in the current elevator systems as it requires minimal modification to the existing equipment as explained later.

The system will consist of 2 major components, the hardware component and the software component. The hardware component will focus on using signals from the microcontroller in the seven segment LED display. This interfacing is commonly done using Arduino and it depends on current flow using a PN Junction diode based on forward and reverse bias principles. Using a degree of interfacing between the seven-segment display and microcontroller we can accordingly obtain the binary codes for each number the display is programmed to show. Assuming the hardware used is an Arduino Uno microcontroller, this binary code can be directly transmitted to an 'ESP8266 Wi-Fi module'. All modern elevators already contain a load sensor which provides feedback on if the elevator is overloaded. It consists of 3 states, empty, full and overload. This is the end of the hardware aspect. From here we can start with the software component, the data transmitted can be converted from binary to integer format and displayed to the user who can understand which floor the elevator is on. Further we can also see if an elevator is moving up, down or idle based on the details obtained from the binary code of the arrow from the display. For sake of this project and due to lack of access to appropriate hardware/permission to use hardware, the software component will be implemented and demonstrated by using a random number generator to provide a start floor, initial direction of motion and a probability check will be implemented to simulate the elevator moving to simulate the system's working in a day-to-day scenario. To implement the weight aspect, the data from the load sensor will transmit to the app to show if the lift is full or not. Another probability check can be implemented to check if the lift does experience an overload situation. At the lowest and highest floors, the elevator will automatically be emptied. A prompt will take the current floor of the person and calculate the time it will take for the lift to arrive to the floor. Then based on arrival, the user can select if the lift arrived on time, before time or after time and accordingly modify the algorithm.

Limitations of the system

1. No way to install on an existing elevator system without calling upon a serviceman or elevator maintenance to install the hardware necessary to transmit the required data from the microcontroller to the website.
2. A server might crash which could cause the website to go offline.
3. Relaying information from the hardware to a website will also require a website to be hosted on a server and have additional overhead costs of maintaining the website and private ownership of the domain name among others.

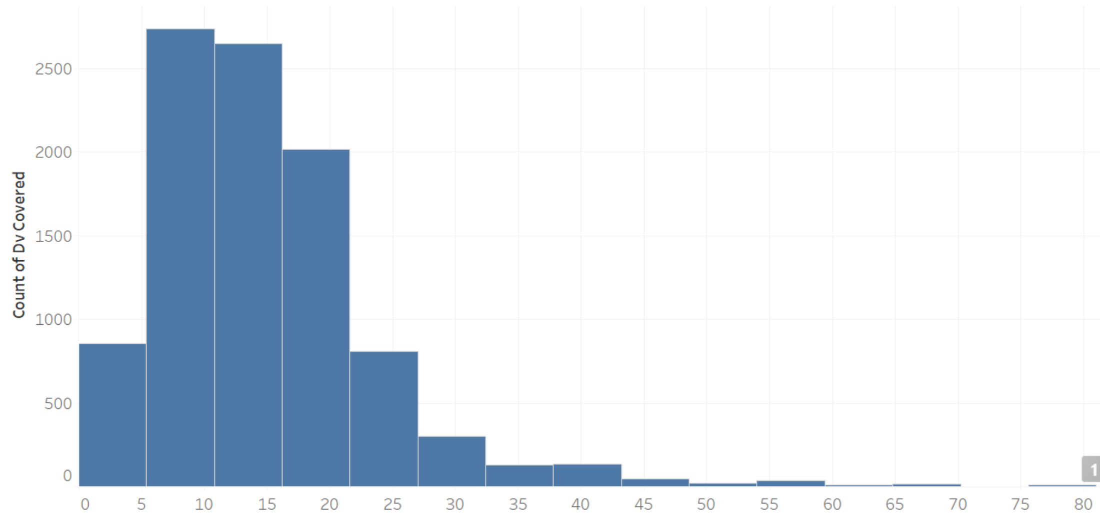
4. It is also to be noted that it is incredibly tedious having to open a website just to find the status of your elevator.

In the future various improvements may be implemented which include, but is not limited to,

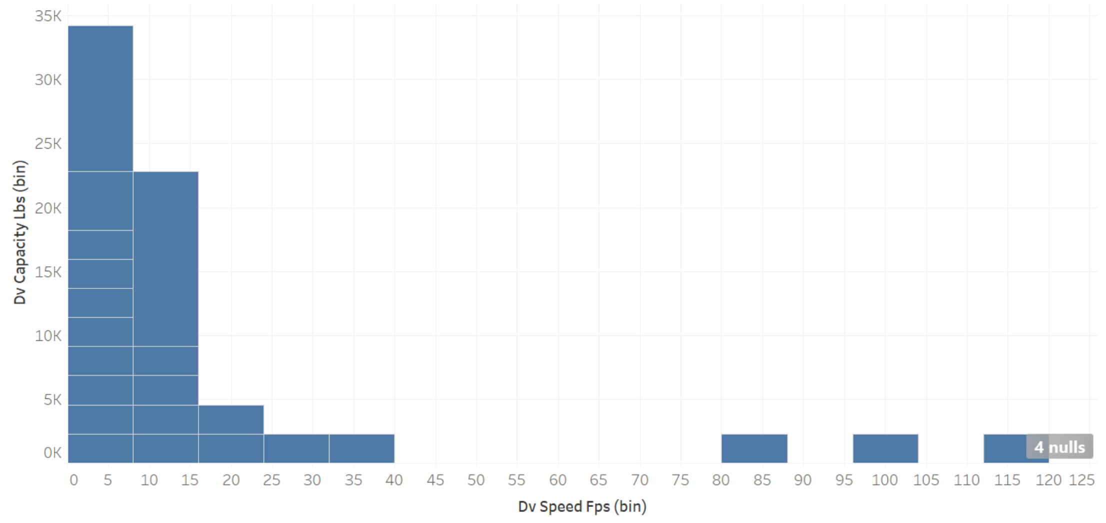
1. Using an AI or ML model along with datasets to customize time taken depending on time of day, rush hours will take more time.
2. A better system of possibly wirelessly tracking the elevator position using photoreceptors/photosensors.
3. Developing an app so as to improve upon convenience and hence the overall user experience.

Tableau

Total Distances Considered

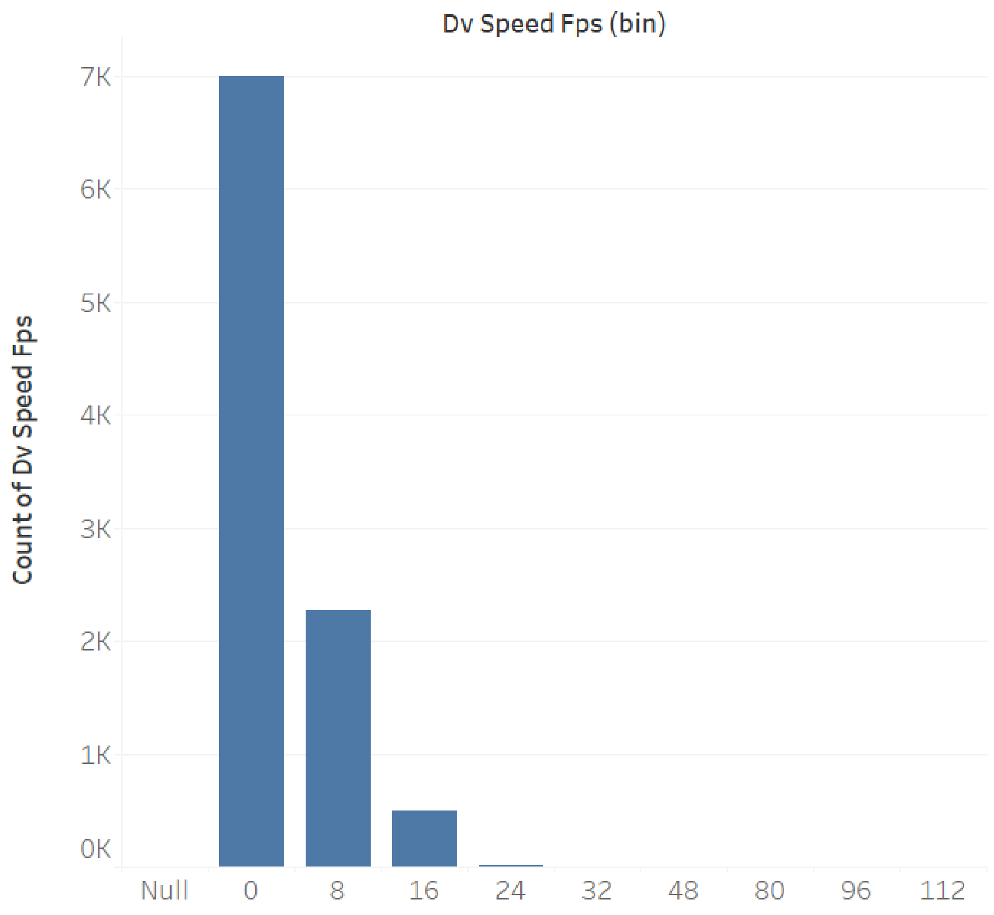


Capacity vs Speed

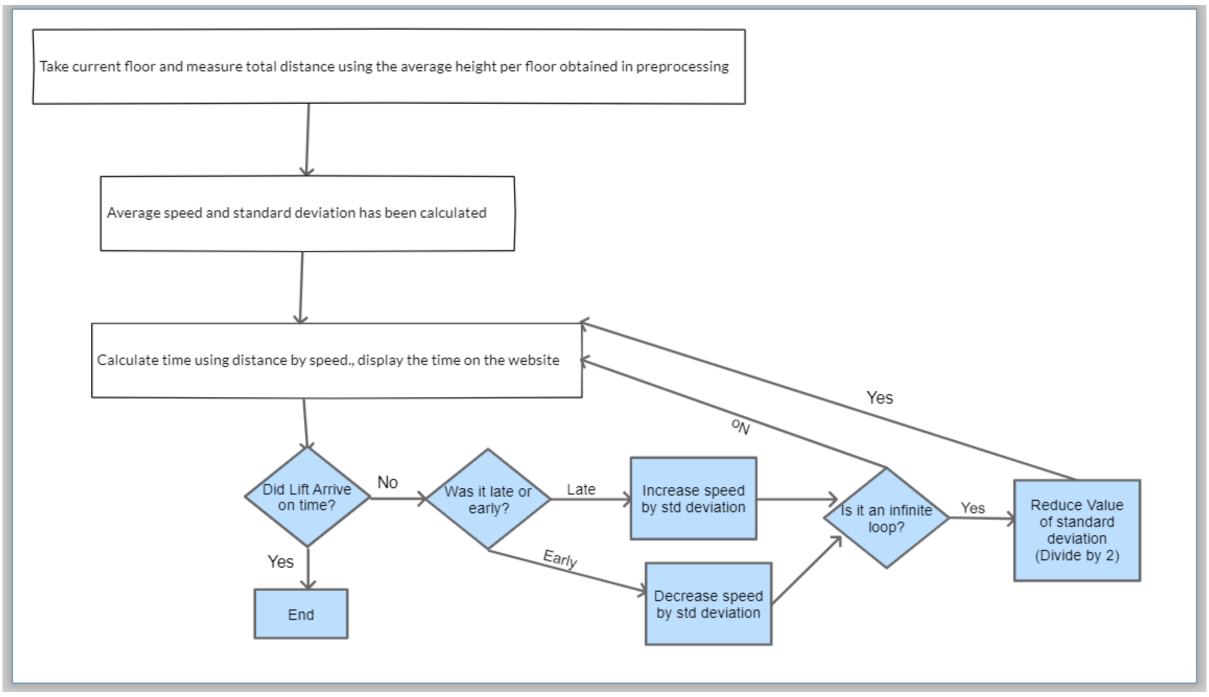


Name: Vansh Dalal Reg. No: 21BCE0635

Average Speed



Workflow:



Take current floor and measure total distance using the average height per floor obtained in preprocessing
Average speed and standard deviation has been calculated
Calculate time using distance by speed., display the time on the website
Did Lift Arrive on time?
No
Yes
End
Was it late or early?
Late
Early
Increase speed by std deviation
Decrease speed by std deviation
Is it an infinite loop?
No
Yes
Reduce Value of standard deviation (Divide by 2)
Yes

Name: Vansh Dalal Reg. No: 21BCE0635

Source Code

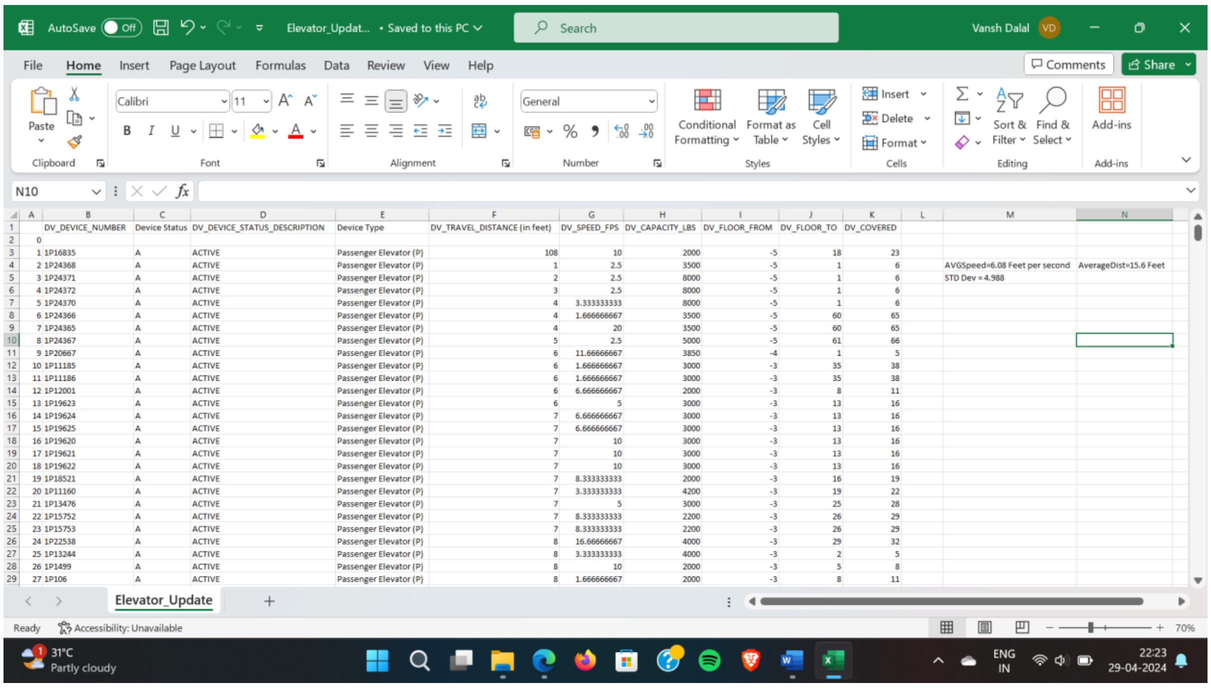
a. Copy main part of your functionality code and sample screenshot.

```
var ML = [];

$('#butt').click(() => {
    $(".Time-popup").addClass("open-popup");
    $(".list-group-item").css({"pointer-events":"none"});
    $(".Time-button-less, .Time-button-more").one("click", function(){
        $(".Time-popup").removeClass("open-popup");
        $(".list-group-item").css({"pointer-events":""});
        const value = $(this).hasClass("Time-button-less") ? 'L' : 'M';
        MLUpdate(value);
        // Unbind click event handlers for both buttons
        $(".Time-button-less, .Time-button-more").off("click");
    });
});

var speed=6.08;stddev=4.988;
var flag=0;

function MLUpdate(value)
{
    ML.push(value);
    if(value=='M')
    {
        speed+=stddev;
    }
    else if(value=='L')
    {
        speed-=stddev;
    }
    if(ML.length==10)
    {
        for(var i=1;i<10;i++)
        {
            if(ML[i-1]==ML[i])
            {
                flag=1;
            }
        }

        if(flag==0)
        {
            stddev=stddev/2;
            console.log(stddev);
            ML=[];
        }
        else
        {
            flag=0;
            console.log(stddev);
            ML=[];
        }
    }
}
```

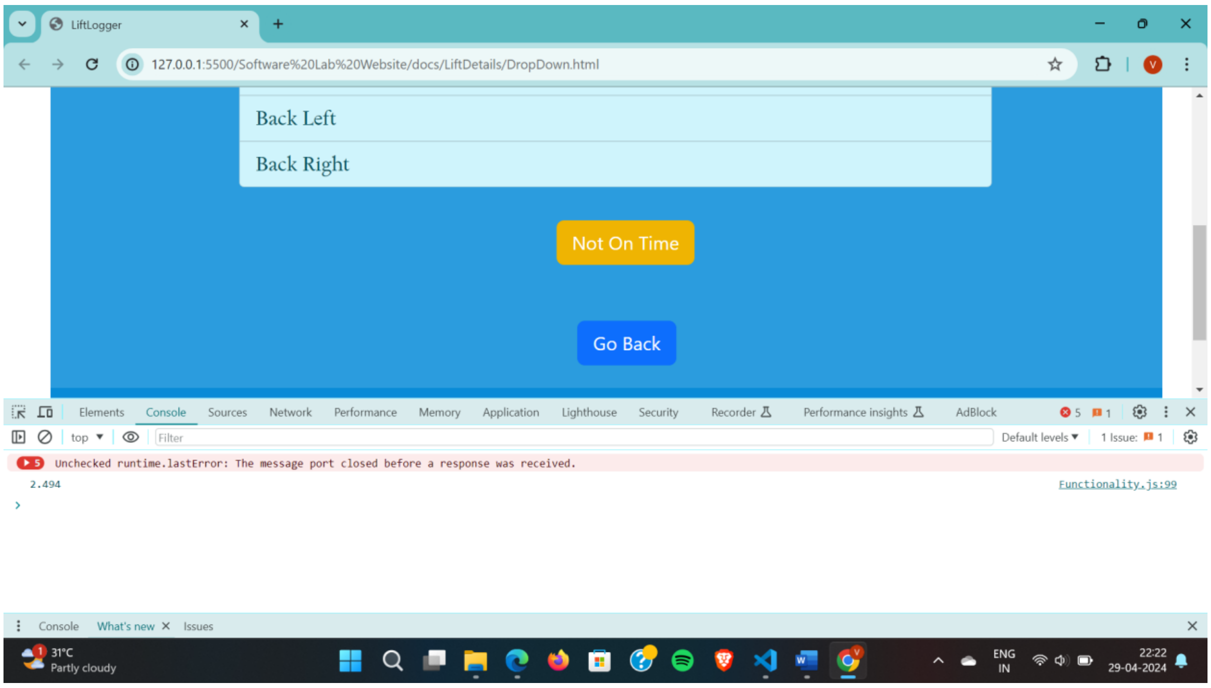
```
Preprocessing

from google.colab import files

uploaded = files.upload()

# The uploaded file is available in the `uploaded` dictionary.
# You can access the file contents using the keys of the dictionary.
import pandas as pd

import io

df=pd.read_csv(io.BytesIO(uploaded['_Test.csv']))
# Filter the DataFrame to keep only rows where the 'Device Type' column
has the value 'Passenger Elevator (P)'
df_filtered = df[df['Device Type'] == 'Passenger Elevator (P)']
# Drop rows where the 'DV_TRAVEL_DISTANCE' column is null
df_filtered = df_filtered.dropna(subset=['DV_TRAVEL_DISTANCE'])
# Calculate the mean of the previous and next 5 values for null values
in 'DV_Speed'
for i in range(len(df_filtered)):
    if pd.isna(df_filtered.loc[i, 'DV_Speed']):
        start_index = max(0, i - 5)
        end_index = min(len(df_filtered), i + 6)
        df_filtered.loc[i, 'DV_Speed'] =
df_filtered.iloc[start_index:end_index]['DV_Speed'].mean()

# Sort the DataFrame by 'DV_Speed'
df_sorted = df_filtered.sort_values(by='DV_Speed')
df_sorted['DV_FLOOR_FROM'] = df_sorted['DV_FLOOR_FROM'].replace(['L',
'M', 'B'], [0, 0, -1])
# Sort the DataFrame by 'DV_FLOOR_FROM'
df_sorted = df_sorted.sort_values(by='DV_FLOOR_FROM')
```

```
from google.colab import files

uploaded = files.upload()

# The uploaded file is available in the `uploaded` dictionary.
# You can access the file contents using the keys of the dictionary.
import pandas as pd
import math
import io
uploaded_file_key = list(uploaded.keys())[0]
df=pd.read_csv(io.BytesIO(uploaded[uploaded_file_key]))


# 6. Take Feedback on if lift arrived on time.
# 7. Implement a history to check if standard deviation must be
reduced.

# Floor Distance
df['DV_COVERED'] = df['DV_FLOOR_TO'] - df['DV_FLOOR_FROM']

# Avg height per floor
# df['AVG_HEIGHT'] = df['DV_COVERED']/df['DV_TRAVEL_DISTANCE']
# average=(df['AVG_HEIGHT'].mean())*100
# average="{:.1f}".format(average)
average=15.6

df.rename(columns={'DV_SPEED_FPM':'DV_SPEED_FPS'}, inplace=True)
averagespeed=df['DV_SPEED_FPS'].mean()
avgspeed="{:.1f}".format(averagespeed)
print(avgspeed)
```

Name: Vansh Dalal Reg. No: 21BCE0635



	DV_DEVICE_NUMBER	Device Status	DV_DEVICE_STATUS_DESCRIPTION	Device Type	DV_TRAVEL_DISTANCE (in feet)	DV_SPEED_FPS	DV_CAPACITY_LBS	DV_FLOOR_FROM	DV_FLOOR_TO	DV_COVERED		
0												
1	1P16835	A	ACTIVE	Passenger Elevator (P)	108	10	2000	-5	18	23		
2	1P24368	A	ACTIVE	Passenger Elevator (P)	1	2.5	3500	-5	1	6	AVGSpeed=6.08 Feet per second	AverageDist=15.6 Feet
3	1P24371	A	ACTIVE	Passenger Elevator (P)	2	2.5	8000	-5	1	6	STD Dev = 4.988	
4	1P24372	A	ACTIVE	Passenger Elevator (P)	3	2.5	8000	-5	1	6		
5	1P24370	A	ACTIVE	Passenger Elevator (P)	4	3.333333333	8000	-5	1	6		
6	1P24366	A	ACTIVE	Passenger Elevator (P)	4	1.666666667	3500	-5	60	65		
7	1P24365	A	ACTIVE	Passenger Elevator (P)	4	20	3500	-5	60	65		
8	1P24367	A	ACTIVE	Passenger Elevator (P)	5	2.5	5000	-5	61	66		
9	1P20667	A	ACTIVE	Passenger Elevator (P)	6	11.66666667	3850	-4	1	5		
10	1P11185	A	ACTIVE	Passenger Elevator (P)	6	1.666666667	3000	-3	35	38		
11	1P11186	A	ACTIVE	Passenger Elevator (P)	6	1.666666667	3000	-3	35	38		
12	1P12001	A	ACTIVE	Passenger Elevator (P)	6	6.666666667	2000	-3	8	11		
13	1P19623	A	ACTIVE	Passenger Elevator (P)	6	5	3000	-3	13	16		
14	1P19624	A	ACTIVE	Passenger Elevator (P)	7	6.666666667	3000	-3	13	16		
15	1P19625	A	ACTIVE	Passenger Elevator (P)	7	6.666666667	3000	-3	13	16		
16	1P19620	A	ACTIVE	Passenger Elevator (P)	7	10	3000	-3	13	16		
17	1P19621	A	ACTIVE	Passenger Elevator (P)	7	10	3000	-3	13	16		
18	1P19622	A	ACTIVE	Passenger Elevator (P)	7	10	3000	-3	13	16		
19	1P18521	A	ACTIVE	Passenger Elevator (P)	7	8.333333333	2000	-3	16	19		
20	1P11160	A	ACTIVE	Passenger Elevator (P)	7	3.333333333	4200	-3	19	22		
21	1P13476	A	ACTIVE	Passenger Elevator (P)	7	5	3000	-3	25	28		
22	1P15752	A	ACTIVE	Passenger Elevator (P)	7	8.333333333	2200	-3	26	29		
23	1P15753	A	ACTIVE	Passenger Elevator (P)	7	8.333333333	2200	-3	26	29		
24	1P22538	A	ACTIVE	Passenger Elevator (P)	8	16.66666667	4000	-3	29	32		
25	1P13244	A	ACTIVE	Passenger Elevator (P)	8	3.333333333	4000	-3	2	5		
26	1P1499	A	ACTIVE	Passenger Elevator (P)	8	10	2000	-3	5	8		
27	1P106	A	ACTIVE	Passenger Elevator (P)	8	1.666666667	2000	-3	8	11		


LiftLogger
127.0.0.1:5500/Software%20Lab%20Website/docs/LiftDetails/DropDown.html
Back Left
Back Right
Not On Time
Go Back

```
Unchecked runtime.lastError: The message port closed before a response was received.
2.494                                                          Functionality.js:99
```

Repository Link:

Name: Vansh Dalal Reg. No: 21BCE0635

https://github.com/Terry52139/LiftLogger

Published Website:

https://terry52139.github.io/LiftLogger/