ASSIGNMENT 2 – PANDEMIC MODE SYSTEM

Rajiv Mehta - 45433062

Macquarie University
COMP2050

1 TABLE OF CONTENTS

2 INTRODUCTION

2.1 ROLES

Table 1. Roles

Name	*Project A*	*Project B*
Rajiv Mehta - 45433062	Client	Software Engineer
Thomas Cowan - 45321760	Software Engineer	Client

2.2 VISION STATEMENT

The safety of passengers has always been the main focus when designing and implementing elevators into society. Due to the outbreaks of COVID-19 across the globe and there potentially fatal impacts on certain members of society. New laws and regulations have been introduced by governments introducing the idea of social distancing to stop the spread of this outbreak. Due to these changes, aspects such as maximum occupancy in an elevator at any point in time has changed and in turn, has affected how elevators in high demanded areas such as office buildings are highly affected. As a result, we have decided to design and implement a new safety system known as 'pandemic mode' that will introduce frequent cleaning and detailed airflow control systems in order to ensure the safety of all passengers on elevators around the world.

2.3 DEFINITIONS

Table 2. List of Definitions

Abbreviation	*Definition*
PM	Pandemic Mode
SDS	System Design Document
SRS	System Requirements Specification

3 SYSTEM DESIGN DOCUMENT

3.1 PURPOSE AND SCOPE

The purpose of the System Design Document (SDS) is to outline the architecture for the pandemic mode (PM) system being integrated into elevators around the world in order to help stop the spread of COVID-19. This document will outline the system architecture in terms of both the software and hardware components, the methods of data storage that will be used the PM, trade-offs and changes to the original design and any concurrent processes and the strategies used to coordinate them. The document will also have a package diagram for the system to showcase the transfer of information between subsystems within the PM.

3.2 SYSTEM ARCHITECTURE

3.2.1 System Hardware Architecture

With the implementation of this 'pandemic mode' system into elevator systems across the globe, a large variety of components need to be implemented into an already existing elevator in order for the new system to be operational. A list of the hardware components can be seen in table 3.

Table 3. Hardware components

Components	***Description***
Motion Sensors	Various motion sensors will need to be installed in order to detect the number of passengers entering/leaving the train and allow passengers to select which floor they would like to travel to without having to physically press a button.
Weight Sensors	Weight sensors on the elevator will also need to be implemented on the elevators to also help social distance on an elevator. For example, if a guy is very heavy it can be assumed that they would take more space than a person of average weight, meaning that it could become harder for other passengers to social distance.
Microphones	Multiple microphones will need to be placed within each elevator carriage in order to detect the sounds that passengers make. If a passenger sneezes or coughs, the microphones will pick up this sound and deploy the emergency systems.
Disinfectant spray machine	The spray machine will act as a disinfectant for the elevator as when the elevator is empty the machine will spray the elevator to periodically sanitise the elevator.
Hardware to operate the pandemic mode system	Hardware such as new control boards, ICs, and storage/database units will be needed to be implemented in order for the software to be integrated into the pre-existing elevator software.
Operational Elevator	This is the elevators that the pandemic mode system will be implemented onto.
Alert Siren	This siren will be needed to be installed to let both staff and passengers to leave the elevator if a passenger has coughed/sneezed on a particular elevator.
Communication Gear	Communication gear needs to be also implemented between staff and the cleaners so that staff can inform which elevators need to be cleaned when something has occurred out of schedule (reactive approach)
Video Camera	A video camera will also need to be placed on elevators, so that video can be used to identify if someone looks ill and help identify the passenger for contact tracing.

3.2.2 System Software Architecture

The ‘pandemic mode’ (PM) system on software level will be designed and implemented through the use of the Java. The reason behind this is that Java uses a class-based object oriented language, which would be ideal in the case of the PM as it has many different sub-systems, not all of which have a reliance on another subsystem. A list of the classes and there descriptions can be seen in table 4.

Table 4. List of Software Classes for Pandemic Mode

Class	*Description*
Elevator	Holds the normal systems on an elevator
PandemicMode	Main class that holds all the information of the
PassengerCounter	This class will be used to count the number of passengers entering and exiting each elevator and compare to the maximum number allowed.
MotionFloor	This class will be used to allow passengers to select which floor they would like to travel to via a motion sensor.
Spray	This class will activate the spray dispenser if the passenger count from PassengerCounter is equal to 0.
Siren	This class will activate the siren if emergency mode is activated.
Emergency	This class moves the elevator into an emergency state if the Microphone detects a sneeze or cough within the elevator.
WeightCalc	This class calculates the current weight of the elevator and compares it to the max weight set in each elevator
Staff	This class sets up the basic attributes of a staff with attributes such as access and role.
Cleaners	Extends the Staff class to make a specific cleaner
Admin	Extends the Staff class to make a specific admin
Camera	Turns the camera on/off so that Admin can see the progress of the cleaning.
Speaker	Turns the speaker on/off so that Admin can communicate with the cleaning staff when an emergency occurs
Microphone	Turns the non-filtered microphone in the elevator on/off so that cleaners can communicate with admin staff.
DoorOpen	This class forces the doors to open/close depending on the state of the elevator.
FilterAudio	Searches for sneezes and coughs and filters out other sounds by the use of speech recognition libraries and software.
NumberOfCleans	This counts both the number of sprays and the number of times a deep clean of the elevator has occurred and displays a time-stamp.

3.3 STORAGE AND PERSISTENT DATA STRATEGY

The ‘Pandemic Mode’ is a system that is continuously running and updating, resulting in the system requiring a sustainable data storage system in order for the system to not fail during its operational life time. Subsystems of the PM such as the ‘passenger counter system’ and ‘FilterAudio’ are receiving constant inputs as the number of passengers entering or exiting the elevator is constantly changing, while the number of sounds coming from an elevator at any given time is constant and always varying. Though, not all subsystems require to be run continuously. For example, the NumberOfCleans is something that will update periodically but will need to be pulled up by higher level personal in order to order replacements of spray cans and track costs of deep cleanings. Therefore, the PM requires both local storage and off-site databases in order for the system to operate effectively. The local storage will be constantly updated and refreshed is useful as the system does not require a large amount of storage to operate these subsystems. By having it local as well, there will be a reduction in latency across the subsystems that are being operated continuously. Then by having a separate database offsite for storage of important information, many different personal will be able to access the same information and in turn make business decisions based on the information stored. Therefore, having two separate storage systems, one local and one off-site is how the PM will operate as the subsystems vary on the type of information stored and how long it needs to be retained.

3.4 TRADE-OFFS

1. Full Implementation into all elevators: The first trade-off is that the pandemic mode system will not be able to implemented on every type of elevator. This is because of the nature of pandemic mode system. For instance, elevators in areas that are not close to a cleaning service can only utilise half of the PM system, as the ability to have a constant cleaning service is unlikely meaning that the only constant cleaning would be through the spray sub-system.
2. Cost: Due to the many hardware components required to make this system, the cost to implement this system can be quite high in terms of both time and money. Therefore, a big trade-off is that this system will not be feasible to implement into every elevator, just those in high-foot traffic areas.
3. Designed for high foot traffic areas: With the system being designed with the focus on implementation in high-foot traffic areas. The system will not be able to be optimized in out of reach areas and in turn, the full implementation will be limited especially in low-foot traffic regions. Though, in high-foot traffic areas, where the spread of COVID-19 is most likely to occur, the system will greatly stop the spread of this disease.
4. Reliability: Due to the system also having requiring an off-site database for storage of number of deep cleans and number of sprays, the reliability of the system is dropped as the potential for data loss is higher. This can cause many issues such as if the number of sprays is not recorded correctly, then the elevator could be spraying an empty canister without any of the staff noticing causing the elevators to not be clean.

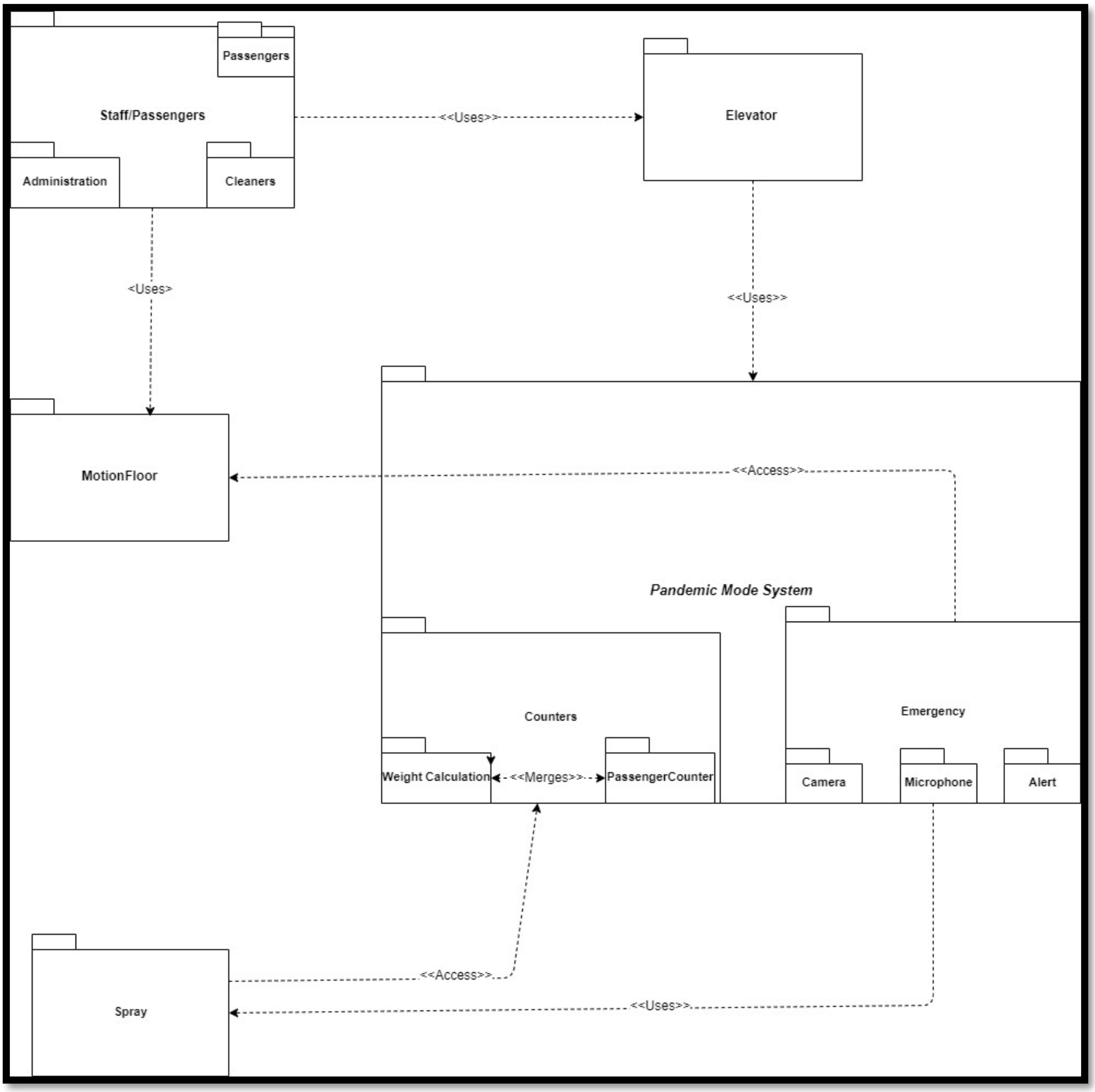
3.5 CONCURRENT PROCESSES

All the subsystems of the PM are all concurrent with the operations of the standard elevator that the system is being applied to. In the PM system, there are many processes that operate at the same time in order to reduce the spread of COVID-19. Depending on the state of the elevator, whether it is in the normal pandemic mode or the emergency state, different subsystems will operate at the same time.

During the normal state for the pandemic mode, the main concurrent processes are between the door motion sensors and the weight sensors. Both subsystems are used to enforce social distancing as the motion sensors on the door are used as a counter to record the number of people on an elevator at any given time. If the number of people on the elevator exceeds the maximum number allowed on the elevator then the elevator will not move until someone exits the elevator and waits for the next. The weight sensor also performs the same purpose as the passenger counter but instead of locking the elevator from moving if the elevator has too many people it will lock if the maximum weight exceeds a certain limit. For instance, if the number of passengers is still below the limit, but a passenger brings along a heavy item, which can assume will take a large amount of space, then the elevator won't start as the ability for others to social distance is limited. This in turn, creates a double checking system so that the system isn't reliant on one measure and ensures that the spread of COVID-19 is limited. Also during the entire time the elevator is active, microphones are record audio and look out for people who sneeze or cough in the elevator. So this process is concurrent with the normal elevator operations. When the emergency state is in action, both microphones and speakers are activated in order to allow communication between the passengers and staff. At the same time as this, staff are also notified and video footage is started to be recorded and transmitted to the off-site database, which can then be used for contact tracing.

3.6 PACKAGE DIAGRAM

Figure 1. Package Diagram



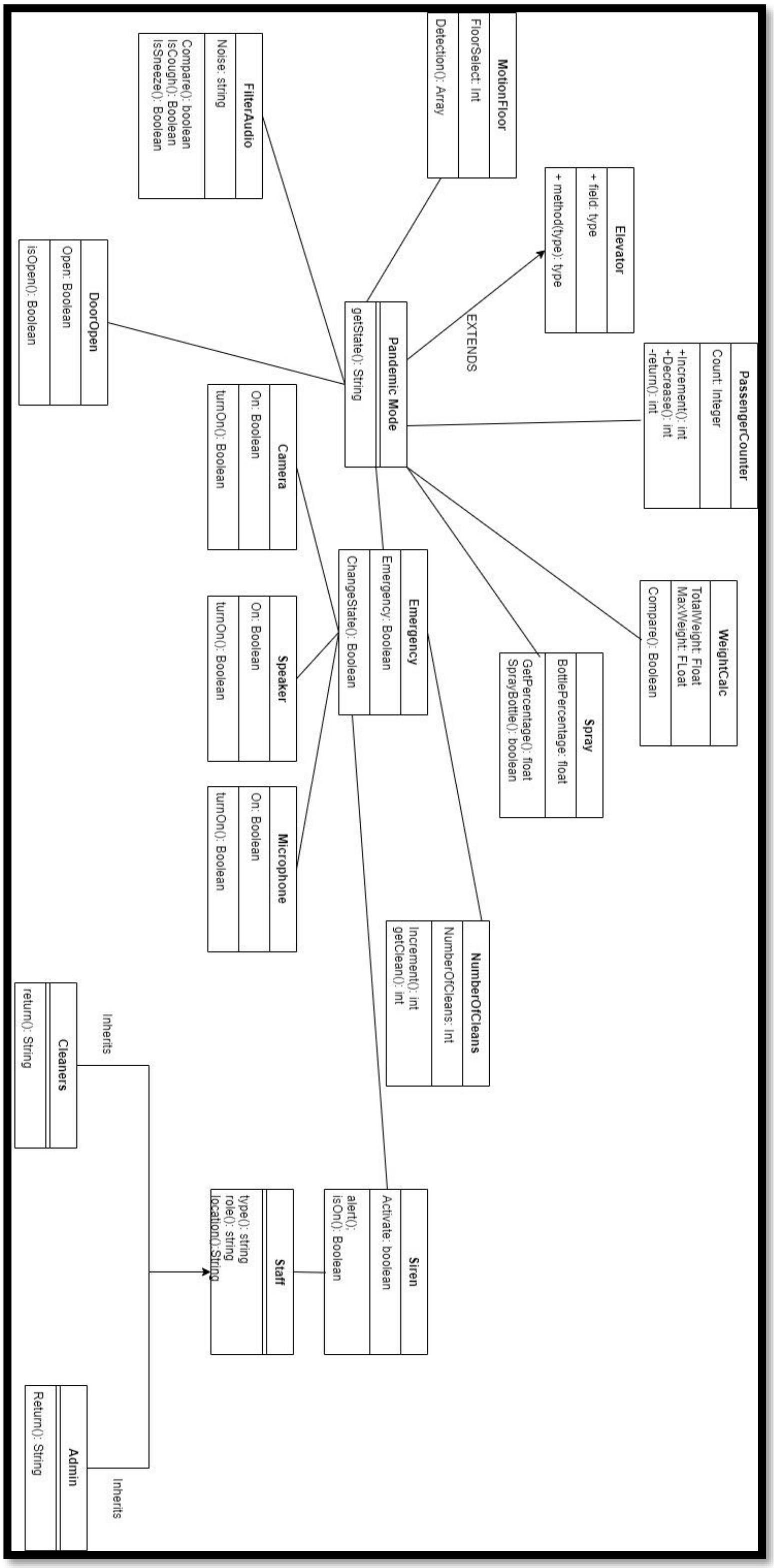
4 DATA DEFINITIONS

Table 5. Data Definitions

Name	***Type***	***Definition***	***Example***
PassengerCount	Integer	Holds the number of passengers currently in the elevator.	3
AmountOfSpray	Float	Indicates the percentage of the disinfectant cans have left in the bottles.	43.6
NumberOfDeepCleans	Integer	Holds the number of times the elevator has been deep cleaned for the day.	5
DeepCleanTime	String	Holds the time that the deep cleaning occurred	"Deep Clean occurred at 6:30pm on the 23/10/2020"
MaxWeight	Float	Holds the maximum weight the elevator can handle.	320.0
MaxPassenger	Integer	Holds the maximum number of passengers that the elevator can hold.	4
DoorOpen	Boolean	Force changes the door to be open or closed	True
SirenControl	Boolean	Turns on/off	False
MicrophoneControl	Boolean	Turns on/off	True
SpeakerControl	Boolean	Turns on/off	False
VideoControl	Boolean	Turns on/off	True

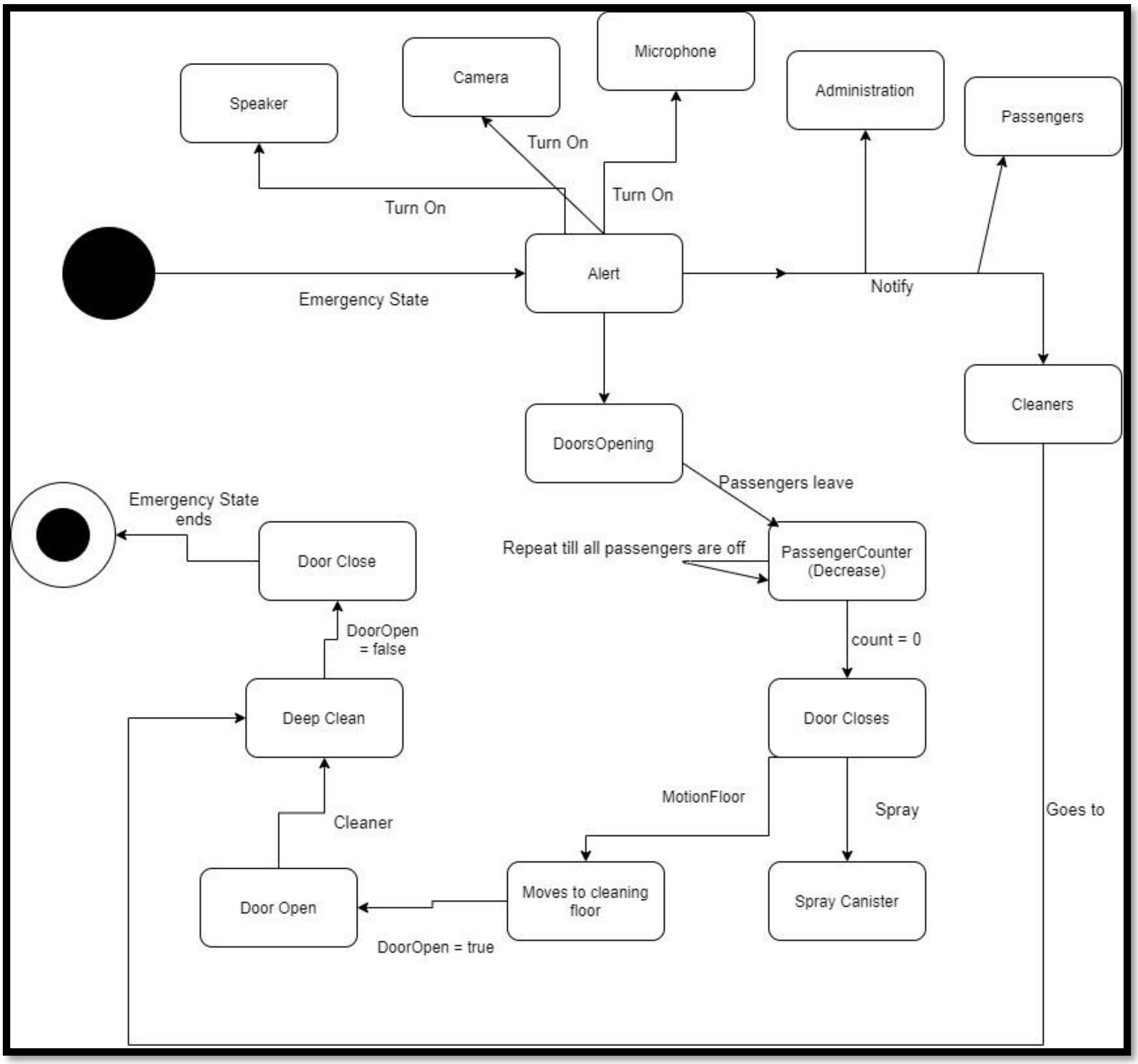
5 ANALYSIS AND CLASS DIAGRAM

Figure 2. Class Diagram



6 STATE DIAGRAM

Figure 3. State Diagram of an emergency state



7 REQUIREMENTS TRACEABILITY MATRIX

Table 6. Traceability Matrix

Requirement Identification	*Use Cases*	*Classes*	*Methods*	*Packages*	*Build Number*
R1	N/A	N/A	N/A		
R2	1	MotionFloor	Detection();	MotionFloor	
R3	N/A	N/A	N/A		
R4	3	Spray, DoorOpen	GetPerecentage(); SprayBottle(); isOpen();	Spray MotionFloor	
R5	3	PassengerCounter, WeightCalc	Increment(); Decrease(); Return(); Compare();	Counters	
R6	3	Elevator	N/A		
R7	N/A	N/A	N/A		
R8	N/A	N/A	N/A		
R9	2	Microphone	TurnOn();	Emergency	
R10	1,2	FilterAudio	TurnOn();	PandemicMode	
R11	1,2,3	PassengerCounter, WeightCalc, Microphone, Emergency, Speaker, Camera, Siren, DoorOpen, Staff	Increment(); Decrease(); Return(); Compare(); Alert(); TurnOn(); Type(); Role(); Location();	Emergency Counters Spray Staff/Passengers	
R12	1	MotionFloor	Detection();	MotionFloor	
R13	1	MotionFloor	Detection();	MotionFloor	
R14	N/A	N/A	N/A		
R15	N/A	N/A	N/A		
R16	2	Emergency	ChangeState();	Emergency	
R17	2	Emergency, PassengerCounter, WeightCounter	ChangeState(); Increment(); Decrease(); Return(); Compare();	Emergency Counters	
R18	1,2	Emergency, MotionFloor	ChangeState(); Detection();	Emergency MotionFloor	
R19	2	Emergency, MotionFloor, PassengerCounter, WeightCounter	ChangeState(); Detection(); Increment(); Decrease(); Return(); Compare();	Emergency MotionFloor Counters	
R20	N/A	N/A	N/A		
R21	1	WeightCounter	Compare();	Counters	
R22	N/A	N/A	N/A		
R23	2,3	Spray, PassengerCounter	GetPerecentage(); SprayBottle();	Spray Counters	

			Increment(); Decrease(); Return();		
R24	2,3	Spray, PassengerCounter	GetPerecentage(); SprayBottle(); Increment(); Decrease(); Return();	Spray Counters	
R25	N/A	N/A	N/A		
R26	N/A	N/A	N/A		
R27	2	Microphone	turnOn();	Emergency	
R28	2	Microphone	turnOn();	Emergency	
R29	N/A	N/A	N/A		
R30	N/A	N/A	N/A		
R31	2	WeightCounter, Emegency	Compare(); ChangeState();	Counters	
R32	2	MotionFloor, Door, Emergency	ChangeState(); Detection(); isOpen();	MotionFloor Emergency	
R33	2	Emergency, WeightCounter	ChangeState(); Compare();	Emergency Counters	

8 ASSUMPTIONS

1. The PM will integrate perfectly with the operating systems used by any elevator.
2. The PM will only be initially installed and tested in locations that can utilise the full implementation of the system i.e. high foot-traffic areas such as shopping centres and work buildings.
3. The PM will be able to send signals perfectly to the correct databases.
4. PM will always be active i.e. program does not need to be turned on/off.
5. The deep cleaning for the elevators has changed to a reactive approach.
6. Every time the emergency section is activated, the elevator goes down for a deep cleaning before it is available for use.

9 TESTING

9.1 OVERVIEW

Requirements that are being tested:

- R2: PM Transportation
  - Fit Criteria: PM must transport people from the floor that they are currently on to the floor that they have selected by using the user interface.
- R4: Diffuse Disinfectant
  - Fit Criteria: The diffuser will be able to release a disinfectant into the carriage when prompted to do so
- R5: Diffuser Availability
  - Fit Criteria: The diffuser may only be used when no passengers are currently in the elevator's carriage

- R9: Microphone
  - Fit Criteria: The microphone must listen to the audio from within the elevator's carriage
- R10: Recorded Audio Processing
  - Fit Criteria: The audio recorded by the microphone will be processed by the audio for any noises that could mean an individual riding the elevator is sick
- R11: Potential Sick Individual
  - Fit Criteria: Cleaning of the elevator's carriage will be scheduled immediately if the system determines a person is likely sick
- R13: Intended inputs via Motion Sensors
  - Fit Criteria: Motions sensors will only become triggered when the user's actions are considered deliberate
- R17: Emergency State Triggers
  - Fit Criteria: The emergency state must be triggered when too many people are in the lift
- R:18 Emergency State Actions
  - Fit Criteria: The emergency state must go to the nearest floor and open the doors of the elevator
- R:19 Emergency State Recovery
  - Fit Criteria: PM will exit out of the emergency state when there are less than the maximum amount of people in the elevator
- R20: Scheduled Cleaning
  - Fit Criteria: The Elevator Controller will send a message to a cleaner that the Elevator Carriage is scheduled for cleaning with an approximate timeframe
- R21: Elevator Carry Weight
  - Fit Criteria: The elevator must be able to carry a weight of 200 Kg per 2 m2 of floor space the elevator's carriage has
- R24: Disinfectant defusal time
  - Fit Criteria: The disinfectant will be defused for no longer than three seconds in a single cleaning session
- R27:Microphone Detection
  - The microphone must be able to pick up sounds of 30 decibels
- R28:Audible Symptom detection
  - The system must be able to detect 100% of audible symptoms
- R31: Emergency State Being Triggered
  - Fit Criteria: PM will enter into an emergency state when the weight inside the elevator is greater than 200 Kg per 2 m2 of floor space
- R32: Emergency State Reaction
  - Fit Criteria: When PM enters into an emergency state, the elevator will move to the nearest floor and open its doors in under 60 seconds
- R33: Emergency state Release
  - Fit Criteria: PM will exit out of the emergency state when it is at a floor and the weight inside the elevator is less than 200 Kg per 2 m2 of floor space

9.2 TEST CASE SPECIFICATIONS

Table 7. Test-Case 1 Emergency state

Test Case ID	***TC-1***
Requirements	R2,R9,R10,R11 R13, R17, R18, R19, R21, R24, R27, R31, R32, R33
Description	This test case will determine if the elevator will switch from pandemic mode to the emergency mode and if the emergency state will run as intended.
Input Specification	Various inputs will be required in order to test this specification. Inputs like number of customers, heavy objects that can break past the maximum weight of the lift.
Expected Output	The expected output is that all the requirements are met and that there is no bugs with the emergency state system i.e. operates as initially designed.

Table 8. Test-Case 2 Dispenser

Test Case ID	***TC - 2***
Requirements	R4, R5, R20,R28
Description	This test case will focus on testing the dispenser when the system is in the normal pandemic mode.
Input Specification	1. No passengers in the elevator and full canisters of disinfectant. 2. Passengers in the elevator and full canisters of disinfectant.
Expected Output	The expected output is that the dispenser should be operational and only activate when passengers are not in the elevator.

9.3 TEST PLAN

In order to implement these test plans the first step would be to set up an environment suitable for testing. In order to do this, testing shall be done after work hours in an office building with at least 20 floors. This is so that the full implementation of the PM can be tested within a building with high foot traffic. If the tests are passed under a simulated environment, then it can be further tested in a less heavy foot traffic area so that results can be monitored, but where minimum casualties can occur if the system fails. If the tests are all passed then further testing can be done in high foot traffic areas where results can be monitored.

The second step is to alter the mode of testing will be through administration at the start with participants. Then as the tests continue and locations are moved the mode of testing becomes more passive to see how the system handles everyday life and to monitor results through graphs rather than physically watching.

The last step is to move back to an administration form of testing, but to test with extreme circumstances. For instance stress testing the system by having lots of participants mess around in the elevators to try and cause unlikely circumstances to check the robustness.

10 REFLECTION

There were many difficulties that I encountered when working with the SRS provided. The difficulties mainly stemmed from not having the insight of the SRS and the thought process when it was being made. This made understanding each decision and trying to link how the system was designed to function. For instance, when I was trying to understand how the emergency system was supposed to function in his design, it took a long time to realise that it did not just rely on a counter system for passengers entering and the microphone for detecting sneezes or coughs but also on a weight system where if the elevator weighed over a set capacity then the elevator would also go into its emergency state.

In order to combat this, I had two key solutions. The first was to talk to my partner Thomas (Details on page 2. Table.1) and try to get information on his thought process when he was designing his system the first time around. This led to me getting a better understanding on how he thought of storing data and how the subsystems connected together. Then with that knowledge I implemented my second solution which was to make minor tweaks to his original design to better fit the profile for testing and designing the system. For instance, I added that if the elevator went into an emergency state, then it had to be deep cleaned before the elevator could go back to the standard pandemic state.

By doing this, the document became easier to write as clarification on areas of his SRS and making small tweaks to the original design made it easier to design a system that was logical to me. What I learnt from this document is that the hardest part of working of another software engineers SRS is understanding the original design and thought process that went into creating the project.