UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO

INTERNET OF THINGS

AMBIENT ASSISTED LIVING

Intelligent Car with Safety and Garage Automation (ICSGA)

Professor: Prof. Giuseppe Desolda

GROUP FOUR

ALEX PRAVEEN DASS

(MAT NO: 725190)

SHANMUGAPRIYA BASKARAN

(MAT NO: 725189)

Contents

ACKNOWLEDGEMENT

We would like to thank our beloved professor Prof. Giuseppe Desolda, PhD in Computer Science, IVU Laboratory Department of Computer Science, University of Bari Aldo Moro (UNIBA) for introducing us to the IOT concepts and other powerful tools and encouraging us to learn more and more to improve our skills in one of the most crucial and important subject of the era.

PROJECT SUMMARY

The Internet of Things (IoT) describes the network of physical objects— "**things**"—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet. These devices range from ordinary household objects to sophisticated industrial tools.

Over the past few years, IoT has become one of the most important technologies of the 21st century. Now that we can connect everyday objects—kitchen appliances, cars, thermostats, baby monitors—to the internet via embedded devices, seamless communication is possible between people, processes, and things.

Here we have developed a project based on the term '**AMBIENT ASSISTED LIVING**' **(AAL)** can be defined as the use of information and communication technologies (ICT) in a person's daily living and working environment to enable them to stay active longer, remain socially connected and live independently into old age and one of the largest is human activity recognition and behaviour understanding, with the objectives of detecting and recognizing actions, activity, and situations within an environment.

Here we have developed an innovative concept of an '**Intelligent Car with Safety and Garage Automation (ICSGA)**'. Basically, nowadays all the cars are built with all kinds of inbuilt safety features and durability based on the price he/she pays to the manufacturer. Depending on the price of the car the features are packed up within the hood and the roof. But they're all built to sustain or to restrain the passenger from getting injured. Here we are proposing a completely different concept to a multiple decade old car to save it for the user's future generations.

The following assessments are discussed in brief for (ICSGA):

1. A scenario and personas for AAL (Ambient Assistant Living);

2. The refined scenario and personas after the use of the Tile Cardtechnique, detailing and highlighting how the initial scenario has been improved after the adoption of the Tile Cardtechnique;

3. Smart objects creation in the Tinker cad relating to the scenario;

4. NIST recommendations to prevent cyber-attacks by discussing the mitigation strategies against the smart environment;

5. Design of orchestration proposed in the scenario as automation rules using Node-Red;

INTELLIGENT CAR WITH SAFTEY AND GARAGE AUTOMATION (ICSGA)

SCENARIOS AND PERSONAS

Definition:

Scenario*:* A persona is typically combined with one or more scenarios. A scenario is a brief story that describes how and why the persona would use the product to complete a specific task in a specific context. Different scenarios could be written for different tasks or different contexts.

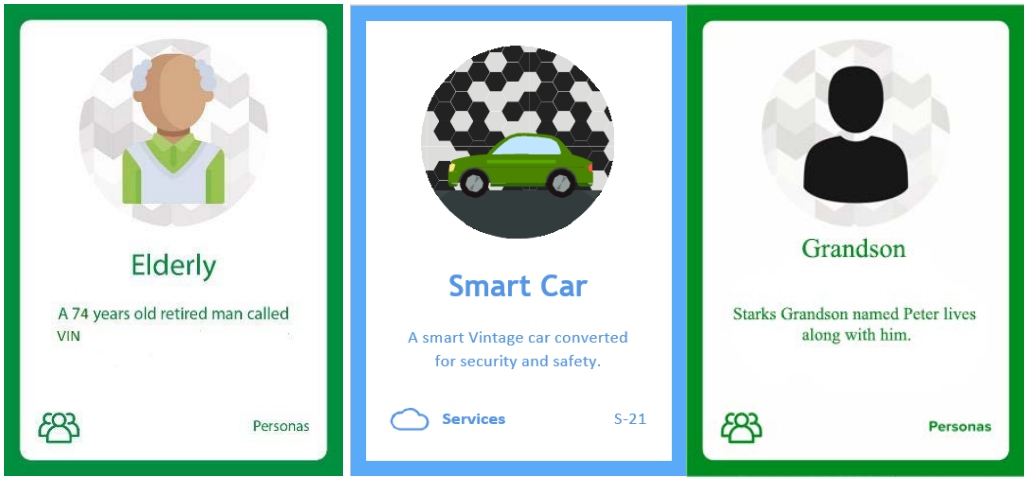
The purpose of a scenario is to help the design team visualize how a target user would interact with the product in real-life. This can help with determining the functions and features of the product for design requirements, as well as with developing tasks for usability testing

Personas: A persona is a description of a target user for a product or service. A persona summarizes the target user's background, goals, and needs related to the product. The persona helps you better understand your target users, so you can design a solution to meet their expectations.

It is important that a persona is fictional (not an actual individual) yet realistic (based on real data from actual users). Since one purpose of a persona is to develop empathy for the target users, a persona should NOT be humorous or demeaning.

1. Explore:

Mr. Stark is 86 years old who lives with his grandson Peter. Stark owns an old 1965 Ford Mustang which he loves more than anything. Stark Drives his car every day to meet his friends and to the golf course. In spite of his old age Stark drives on his own and never lets anyone to touch his car as it is one of the rarest car in the country. So there are many occasions in which his car was attempted to be stolen numerous times. Peter is very much concerned of his grandfather's wellbeing and he wanted to secure his grandfather's favourite and rarest car. So, he decided to upgrade the car without spoiling its originality to satisfy his needs of concerns. So he hired some professionals who is also an expert in Ambient Assisted living to install an IOT device that is **"Intelligent Car with Safety and Garage Automation (ICSGA)"** to help him monitor his grandpa's car and help him drive safe back home every day.



2. Challenges:

Since Mr. Stark is very old he has certain health conditions which he struggle with, firstly he has a habit of forgetting things quite a lot. Then he is always prone to jump scares or panic attacks because of his old age anxiety. So considering all these things in practicality he has these following challenges.

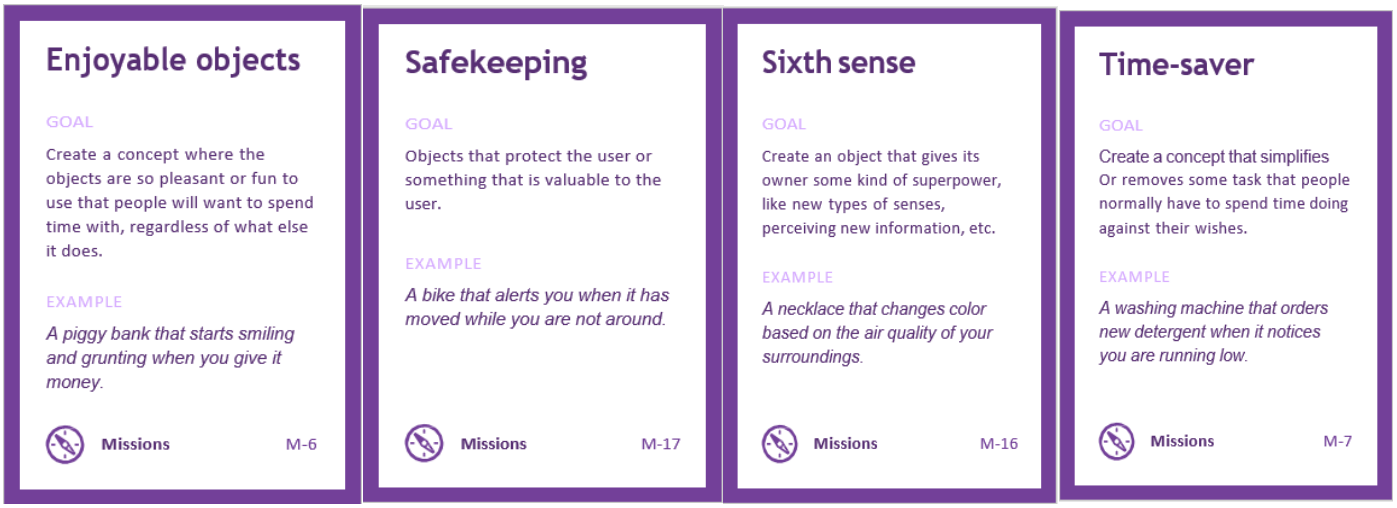
Stark always forgets his car keys, so he needs a password protected car door lock just in case he forgets his keys. So, Stark may forget his keys but not his birth year so he has planned to set that as a password.

As we saw earlier his car is one of the rarest in the country so while it's parked outside the golf course there is a chance for someone to steal it, so it requires a modern day tech alarm to notify both him and his grandpa stating that someone is trying to steal the car in anyway.

Initially in their house they had a manual garage door that is really narrow and requires a lot of care to bring the car outside and it does require quite a lot of effort to screw up/lift the garage door all the way up all by himself at his old age. So, Peter decided to make him an automatic remote accessed garage opener.

Sometimes when he parks his car in the golf ground parking he may find the other car parked on the both sides in which he has to wait till someone takes their car away so that he can get inside his driver's seat and start the car. So peter decided to provide him a remote car operator in which it allows him to turn on the headlights, move the car forward, reverse and halt with a touch of a button.

If in case Stark meets with an accident there should be a device that sends an instantaneous message to Peter, the police and the nearby authorities for instant action to help him because Stark and Peter live in the country side not in a Metropolitan City so emergency assistance is always not instantaneous.



3. Combine - Sketch

The AAL architecture is integrated by IOT devices, these devices are generally resource-constrained devices in terms of processing, memory and storage capacities available as a smart watch and application that can be downloaded in any smart devices like mobile, tab etc. These devices collect and transmit data to the Smart IOT Gateway which acts like a bridge between this device and internet or cloud.

- At first the first mode of security is the password protected car door which is always turned on while the engine is off. So the car is installed with the latest push button start along with the key hole. The car's door will open only if the password is entered correctly. If the password is entered wrong it will send a notification to Stark's cell phone and peter's cell phone at the same time stating that someone has tried to access the car door with a wrong password. If the password is entered correctly then it opens the lock and lets stark inside.

- Before stark sits on the driver's seat, he can simply press the open button from the wall to open his garage door. Once he moves the car away from his house then there will be a sensor outside to detect that the car has completely moved out and initiates the garage to close the garage door.

- But when he comes back home to his garage to park the car the external sensor outside the garage will sense the motion of the car and triggers the garage door motor to activate the open module to open the garage autonomously to let him get inside his garage, as soon as the car goes inside the door if the sensor fetches another motion it keeps the door in the idle state. Stark can safely close the garage door manually from inside pushing the button.

- When Stark goes out in his car there is a can be unfortunate moments in which he may or may not be indulged in an accident. So, there is a smart sensor placed in the front and back bumper of the car. In an unfortunate event if the button is triggered it will turn on the airbags and send the distress message to his grandson Peter and to the emergency care and to the Police station. If the impact is from the sides of the car then there is an inbuilt steering mounted button to activate the accident sensor that will also send out the distress messages.

- After Stark parks his car in any random place as it's a rare car anyone can try to steal it, so once the car is parked a manual security system can be activated which will monitor the car's doors. Since the car driver's door is password protected the person who tries to steal the car will try the other doors or he'll try to break in to the car through the window, the security system has the force sensors in all the door handles and it's also attached to the windows. If someone tries to break through the doors/windows the system will trigger a vigorous alarm and sends out a security breach message to both Peter and Stark to alert them about their car safety.

- If Stark has hard time to find his car during a dark evening time, he can turn on the car lights and turn on the car along with the heaters/air conditioners to prepare the car for a comfortable ride. If the car is stuck in between two other cars in both the side he can accelerate the car to back and forth to retrieve the car from the parking zone.

4. User Story

Stark just like every day prepares his Golf kit to go play some golf. Peter is working from home due to the covid-19 situations. Stark goes to the garage to his car and enters the password lock which is his date of birth. Once the password is entered the car door opens. Then he pushes the garage open button to open the garage to takes his car out. Once the car is out the garage closes the garage door automatically.

Then he starts driving to the freeway and parks his car near the Mc Donald's to grab some breakfast. He puts the car in manual lock and goes into the restaurant with any concern, well thanks to his cars security system. After finishing his breakfast he comes to the parking lot to start his journey again to the golf course, but there is a problem.

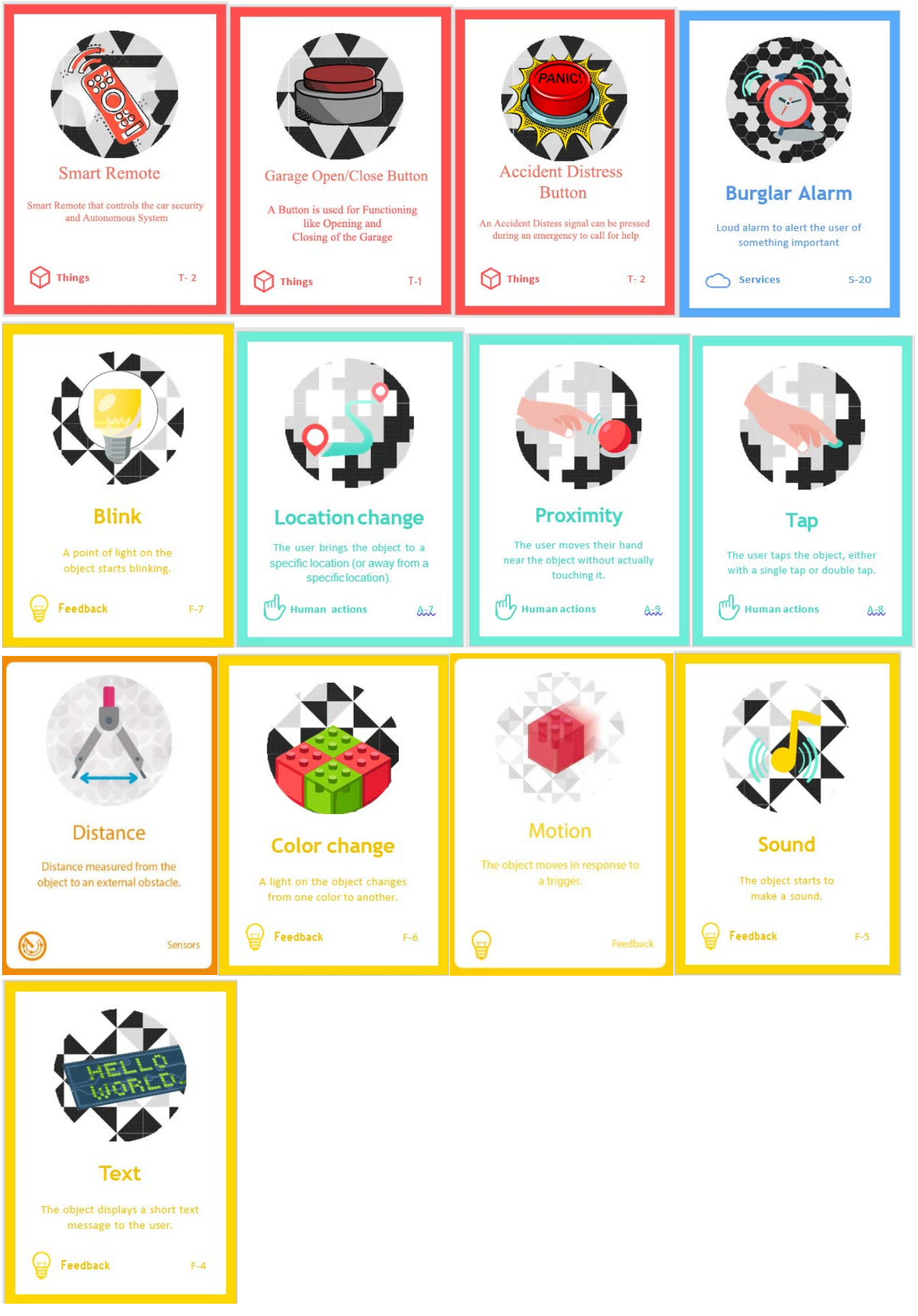
Cars are parked very close in both the sides and he cannot access neither the driver seat nor the password keypad. Thanks to the Autonomous driving Stark uses his car remote and turns off the car security then he uses the forward accelerate button to slowly move the car to the front. As soon as the car is free on the sides he enters his car password and gets inside the car and starts driving.

Then he parks the car turns on the manual car security and goes off cool to play some golf with his buddies. A guy sees the car and was astonished as it was his dream car. So this guy pulls the door handle to see if the car's door is open so that he could take a look inside, the car security triggers the alarm and sends out notification to both Peter and Stark and also puts out a really loud alarm. The securities of the golf court rushes to the parking lot to see what's wrong. The mysterious person flees away realizing that he tripped the alarm out of fear. Stark was informed that his car was safe by the golf court securities and everything was okay. Peter called Stark seeing the notification to check with his grandpa to see what's happening. Stark informed Peter that everything is okay and safe.

On his way back home near his house stark pulled over his car for a traffic stop, a pizza delivery guy came really fast in his scooter as he was late for a delivery, but he crashed into Stark's car's rear bumper. The accident sensor triggered a distress alarm to peter, nearby police station and to the emergency health care centre to send out an ambulance to the triggered location. Luckily Stark was alright but the delivery guy was severely injured. Thanks to the accident sensor the police and ambulance arrived in right time. The police recorded the accident for Stark's insurance claim to fix his bumper and the ambulance was useful for the delivery guy to help him with his injuries. Peter arrived to the scene to check the wellbeing of his grandpa and found him safe and secured.

Then Peter and Stark drove back home and drove his car inside his compound and approached near the garage, thanks to the autonomous garage opener as soon as the car's front reached the desired distance the garage door automatically opened. Stark parked his car inside the garage and the close button was triggered to close the garage door.

IMPROVISATION OF SCENARIOS USING TILE-CARD TECHNIQUES.



1. Key Findings of Tile Card Technique:

Tile card technique is a methodology used to analysis, to identify, clarify and organize system requirements. It is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. The method creates a document that describes all the steps taken by a user to complete an activity.

Tile Card when added to our scenario make it easy in planning system requirements, validating design, and creating an outline for understanding device manuals. A Tile Card document can help the development team identify and understand where errors may occur during a transaction so they can resolve them.

2. Improvisation Outline:

- Tile Card technique captures the functional requirements of a system.
- Tile Card technique is traceable.
- Tile Card technique can serve as the basis for the estimating, scheduling, and validating effort.
- Tile Card technique can evolve at each iteration from a method of capturing requirements, to development guidelines to programmers, to a test case and finally into user documentation.
- Tile Card technique captures additional behaviour that can improve system robustness.
- Tile Card technique have proved to be easily understandable by business users, and so have proven an excellent bridge between software developers and end users

3. Refined Scenario:

The entire Scenario with Tile Card can benefit by revealing how a system should behave while also helping identify any errors that could arise in the process.

- The list of goals created in the scenario process can be used to establish the complexity
- By focusing both on the user and the system, real system needs can be identified earlier in the design process.
- The refined scenario with title cards is easily understood by stakeholders, including customers, users and executives -- not just by developers and testers.
- The creation of title cards helps in identification of exceptions to successful scenarios saves time by making it easier to define subtle system requirements.
- By identifying system boundaries in the design scope of the scenario, developers can avoid scope creep.
- Premature design can be avoided by focusing on what the system should do rather than how it should do it.

In addition, this refined scenario with title cards can be easily transformed into test cases by mapping the common course and alternate courses and gathering data for each of the scenarios.
Furthermore, scenario with title cards can be used in various other areas of software development, including project planning, user documentation and test case definitions. Below we have title cards that describe the features and services that the smart objects hold making it easier to understand how Tile Card have improvised our whole scenario.

4. Key findings after refining the scenario by applying the Tile Card technique

The scenario defined here helps to withdraw a criterion for the activities supported by Ambient Assisted living to help the elderly living persons. Below are some of those:

Innovation:

Always new set of Ideas has to be experimented, new ideas leads to new set of experiences, new set of experiences will lead to new set of problems and new set of problems will lead to new innovative solutions.

User Friendly:

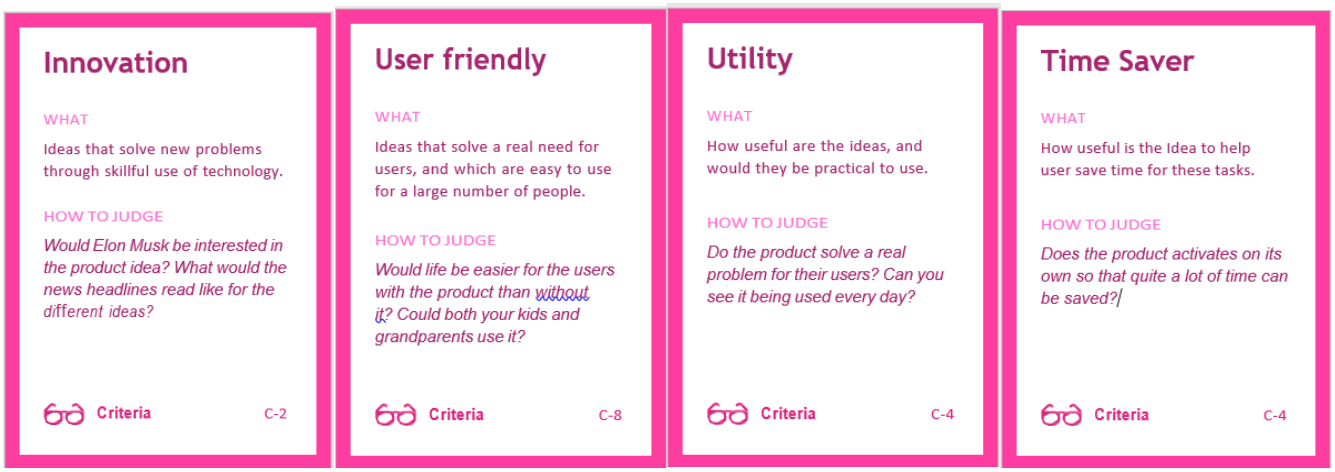
Every Design has to be made with lot of understandable, practical but yet easy/simple to use so that anyone can get access to technology no matter whether they are educated or not.

Utility:

The system is simple and easy to utilize by elderly people with great ease and enjoyment.

Time Saver:

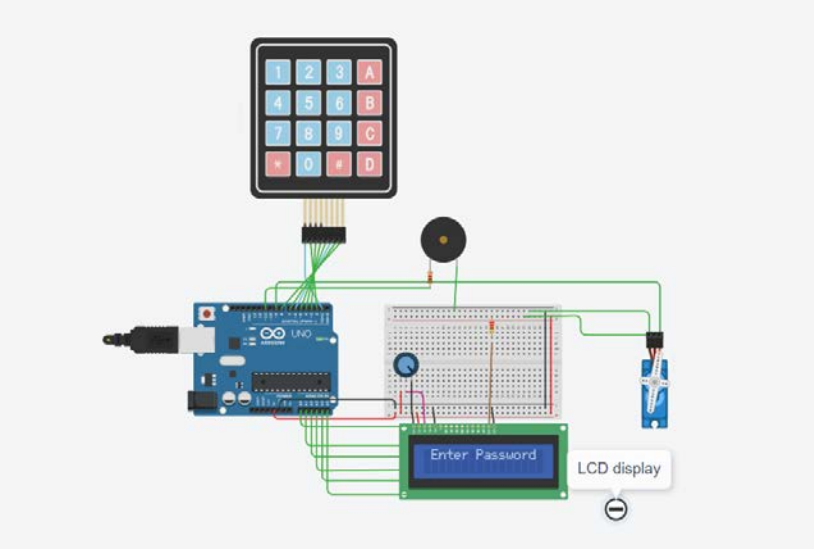
Any type of technology should be designed to reduce the work of a human, on top of that each and every system has to be a time saving mechanism because less time more work and more work more time can be spent useful.



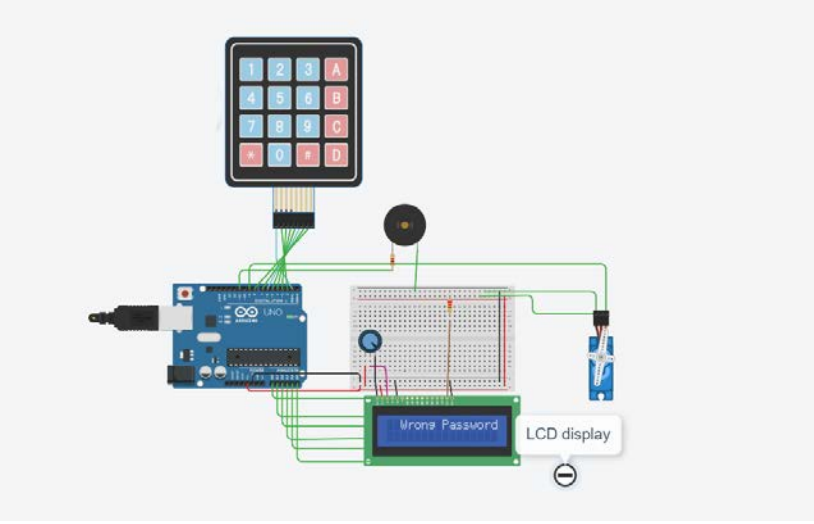
SMART OBJECTS- TINKER CAD

1. Keypad Password Lock for car:

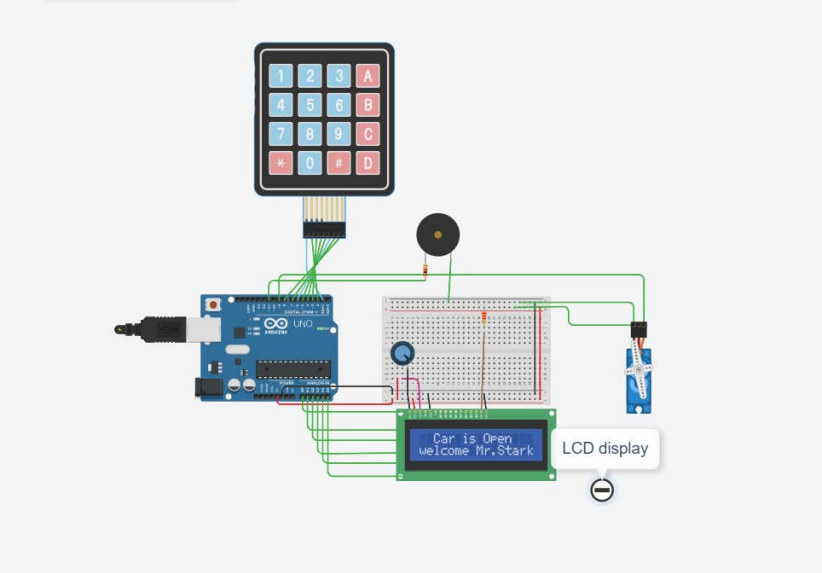
To enhance the security of the car we have installed a password protected access door to the driver's seat. This Smart device comprises of the components such as Keypad 4x4, Positional Micro Servo, Piezo, Arduino Uno R3, LCD 16, 220 Ω Resistor, 1 kΩ Resistor, 250 kΩ Potentiometer. When the password is entered into the keypad eventually the password was set to the date of birth of Mr. Stark so that he remembers them every time when he open the car. If the wrong password is entered into the keypad then the device will send in an alert notification to both Mr. Stark and his grandson Peter that there was a wrong password attempt happened in the car door.



The LCD display indicates that the password has to be inserted in order to open the door.



If the wrong password is entered then an alert message will be sent to the phone and the alarm is triggered that produces a noise.



If the given password is correct then the car door opens up with a greeting message on the LCD display.

Keypad Password Lock Sketch:

```
#include <Keypad.h>

#include <LiquidCrystal.h>

#include <Servo.h>

Servo myservo;

LiquidCrystal lcd(A0, A1, A2, A3, A4, A5);

#define Password_Lenght 7

int pos = 0;

char Data[Password_Lenght];

char Master[Password_Lenght] = "123456";

byte data_count = 0, master_count = 0;

bool Pass_is_good;

char customKey;

const byte ROWS = 4;

const byte COLS = 4;

char keys[ROWS][COLS] = {

 {'1', '2', '3', 'A'},

 {'4', '5', '6', 'B'},

 {'7', '8', '9', 'C'},

 {'*', '0', '#', 'D'}

};

bool door = true;

int buzzer=11;

byte rowPins[ROWS] = {1, 2, 3, 4}; //connect to the row pinouts of the keypad
```

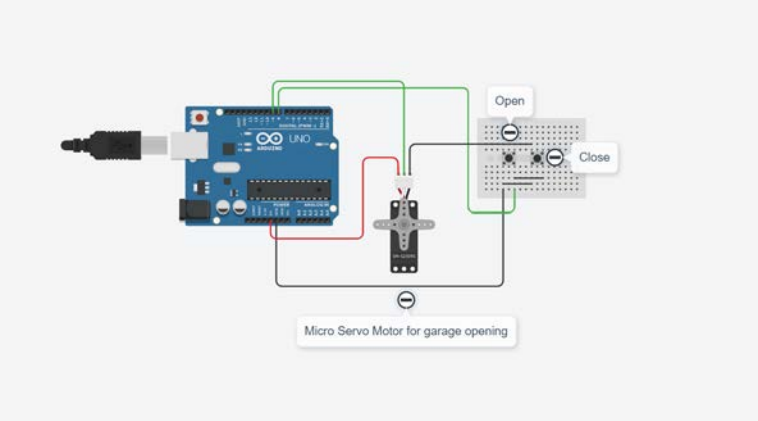
```
byte colPins[COLS] = {5, 6, 7, 8}; //connect to the column pinouts of the keypad

Keypad customKeypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS); //initialize an instance of class NewKeypad

void setup()

{

  myservo.attach(9);

  pinMode(buzzer,OUTPUT);

  ServoClose();

  lcd.begin(16, 2);

  lcd.print(" KEYPAD SECURITY");

  lcd.setCursor(0, 1);

  lcd.print("**HELLO**");

  delay(1000);

  lcd.clear();

}

void loop()

{

  if (door == 0)

  {

    customKey = customKeypad.getKey();

    if (customKey == '#')

    {

      lcd.clear();

      ServoClose();

      lcd.print("  Door is close");

      delay(2000);

      door = 1;

    }

  }

  else Open();

}

void clearData()

{

  while (data_count != 0)
```

```
  {
    Data[data_count--] = 0;
  }
  return;
}
void ServoOpen()
{
  for (pos = 180; pos >= 0; pos -= 5) { // goes from 0 degrees to 180 degrees
    // in steps of 1 degree
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the position
  }
}
void ServoClose()
{
  for (pos = 0; pos <= 180; pos += 5) { // goes from 180 degrees to 0 degrees
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the position
  }
}
void Open()
{
  lcd.setCursor(0, 0);
  lcd.print(" Enter Password");
  customKey = customKeypad.getKey();
  if (customKey) // makes sure a key is actually pressed, equal to (customKey != NO_KEY)
  {
    Data[data_count] = customKey; // store char into data array
    lcd.setCursor(data_count, 1); // move cursor to show each new char
    lcd.print(Data[data_count]); // print char at said cursor
    data_count++; // increment data array by 1 to store new char, also keep track of the number of chars entered
  }
```

```
  if (data_count == Password_Lenght - 1) // if the array index is equal to the number of expected chars, compare data to master
  {
   if (!strcmp(Data, Master)) // equal to (strcmp(Data, Master) == 0)
   {
    lcd.clear();
    ServoOpen();
    lcd.setCursor(0,0);
    lcd.print("  Door is Open");
    lcd.setCursor(0,1);
    lcd.print("welcome home");
    door = 0;
   }
   else
   {
    lcd.clear();
    lcd.print("  Wrong Password");
    digitalWrite(buzzer,HIGH);
    delay(1000);
    digitalWrite(buzzer,LOW);
    door = 1;
   }
   clearData();
  }
}
```

2. Garage Opener (Manual):

Technological mishaps can happen any time. So, going manual in certain cases is always fine. In this we have decided to give Mr. Stark a manual door button to do functions like opening and closing of the garage door. But the real smart device is in the other side of the door which we'll seeing in the upcoming explanations. This Simple device just consists of open close buttons, Micro Servo, an Arduino board and a breadboard for connection.



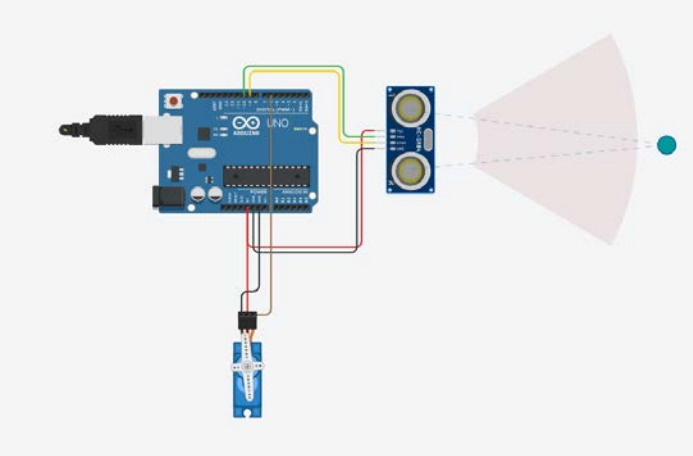
These set of buttons are placed inside the garage for the operations. When open is pressed the garage door lifts up and when close vice versa.

Garage Opener (Manual) Sketch

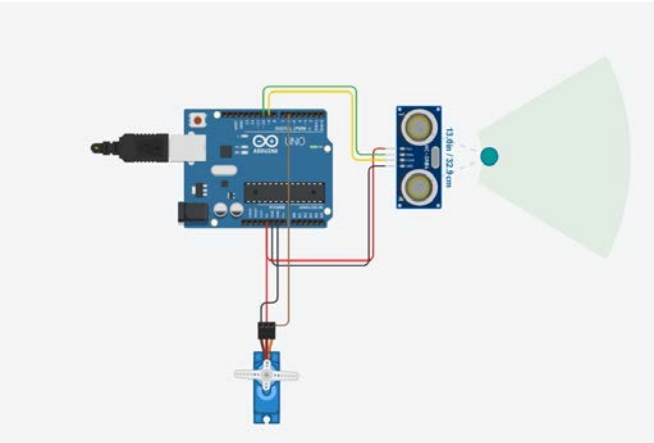
```
#include  "Servo.h"
  Servo monservo;
int PIN_PUSH_VERSE1 = 8;
int PIN_MONSERVO = 9;
int POUSOIR_VERSE1 = 0;
void setup() {
  pinMode(PIN_PUSH_VERSE1, INPUT);
  digitalWrite(PIN_PUSH_VERSE1, HIGH);
  pinMode(PIN_MONSERVO, OUTPUT);
monservo.attach(PIN_MONSERVO);
  monservo.write(0);
}
void loop() {
  POUSOIR_VERSE1 = digitalRead(PIN_PUSH_VERSE1);
  if (POUSOIR_VERSE1 == LOW)  {
    monservo.write(90);
    delay(4000);
    monservo.write(0);
  }
 }
```

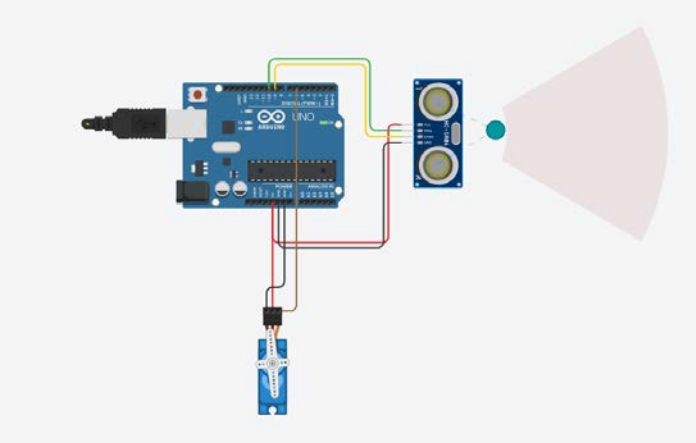
3. Smart Autonomous Garage Door.

When the user comes back home to the garage using of manual door buttons are time consuming. So as the garage is placed inside the house backyard it is useful to have an automated door system to save time and help him to park the car without getting out from the car. This circuit includes components such as Arduino Uno R3, Ultrasonic Distance Sensor and a Servo Motor. The circuit may look simple but still the way it's programmed will just do the job right.



Initially the Ultrasonic Distance Sensor stays in the Idle mode until the car stops by, once the car is detected it will start opening the door just like (below),





After the car drives past the door it can sense that the car has passed by and again will initiate to close the door.

Since the garage can be used for other purposes like for using the washer, dumping old books etc. using a Ultrasonic Distance Sensor to detect presense inside the garage will be not that usefull and be annoying that the door simply opens up even if the user's intention is not to take the car out. Hence we have provided manual operatons for the safer side for user's convienience.

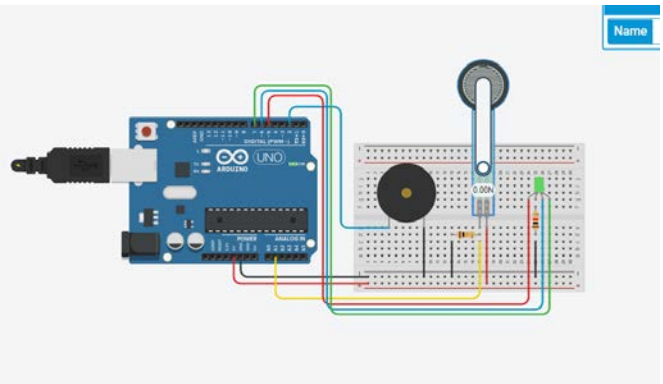
Smart Autonomous Garage Sketch

```
#include <Servo.h>

Servo a;

void setup()

{

 a.attach(6);

 pinMode(10, OUTPUT);
```

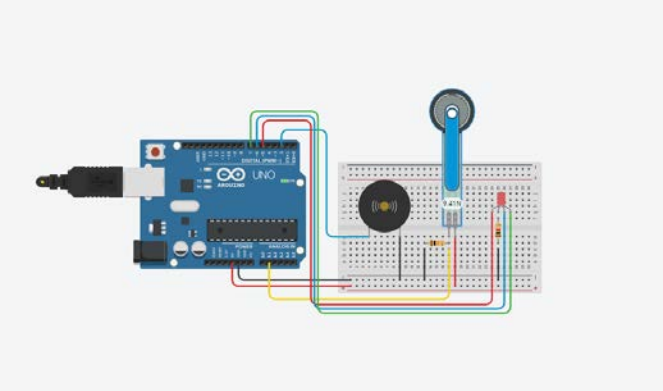
```
 pinMode(9, INPUT);
 Serial.begin(9600);
}
long duration;
int distance;
void loop()
{
 digitalWrite(10,0);
 delayMicroseconds(2);
 digitalWrite(10,1);
 delayMicroseconds(10);
 digitalWrite(10,0);
 duration = pulseIn(9,HIGH);
  distance = ( duration*0.034 )/2;
 Serial.println(distance);
if (distance>=2 && distance < 330)
   a.write(90);
 else a.write(0);
 delay(100);
}
```

4. Smart Car Security.

After Stark parks his car since it's a rare car anyone can try to steal it, so once the car is parked a manual security system can be activated which will monitor the car's doors. Since the cars driver's door is password protected the person who tries to steal the car will try the other doors or he'll try to break in to the car through the window, the security system has the force sensors in all the door handles and it's also attached to the windows. This security system has the ability to work on all the doors and can be turned on and off in a touch of a button. The components present in this circuits are, Arduino Uno R3, Force Sensor, LED, Piezo, Bread board, jumper wires resistors etc.



When an external force is applied on the door handles or on the windows the force sensor activates a vigorous alarm with lights flashing out from the head and tail lights and sends out a message to the users.



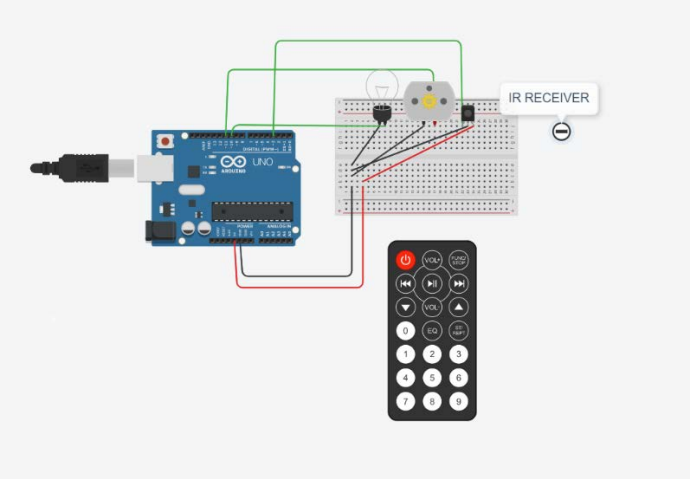
Smart Car Security Sketch

```
const int buzzerPin = 2;

const int forcePin = A1;

const int ledR = 5;

const int ledB = 6;

const int ledG = 7;

int forceValue = 0;

int val;

void setup()

{

  pinMode(forcePin, INPUT);

  pinMode(buzzerPin, OUTPUT);

    pinMode(ledR, OUTPUT);

  pinMode(ledB, OUTPUT);

  pinMode(ledG, OUTPUT);

    Serial.begin(9600);

}

void loop()

{

  forceValue = analogRead(forcePin);

  noTone(buzzerPin);

  tone(buzzerPin, forceValue);

    val = map(forceValue, 0, 914, 0, 100);

    if (val >= 95) {

    toggleLED(LOW, HIGH, LOW);

  } else if (val >= 80) {

toggleLED(HIGH, HIGH, LOW);

  } else {
```

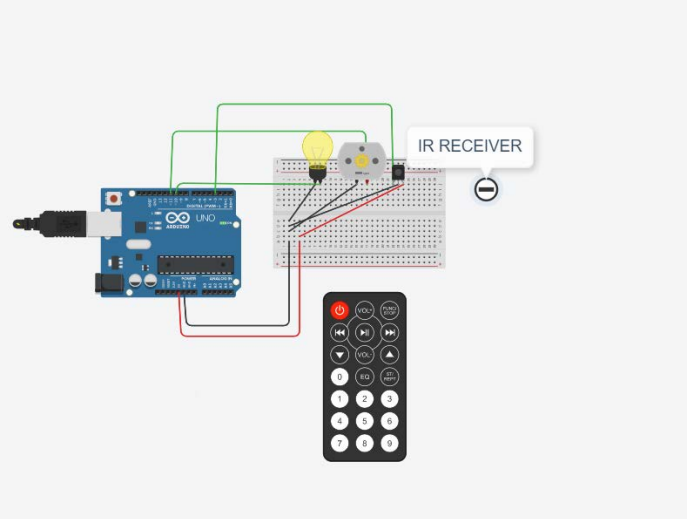
```
    toggleLED(HIGH, LOW, LOW);
  }
    delay(10);
}
void toggleLED(int red, int green, int blue)
{
  digitalWrite(ledR, green);
  digitalWrite(ledB, blue);
  digitalWrite(ledG, red);
}
```

5. Autonomous Driving

The idea of autonomous driving is to typically move the car front and back by turning on the engine and the lights at the same time all by using a click of a remote. This smart device circuit consists of components such as, IR receiver, IR remote, DC motor, Bulb, Bread board, Arduino Uno R3, Jumper wires etc.



When the button is pressed at first the lights are turned on, once the lights are turned on then the motor will be on, once the motor is on then forward and backward motions can be controlled accordingly as seen below.



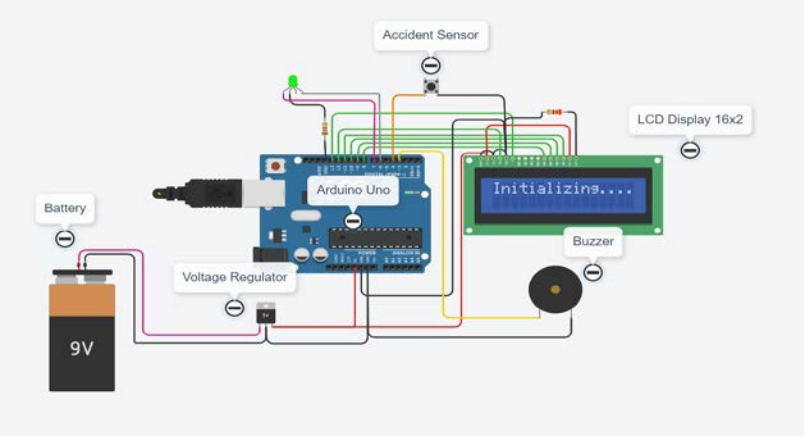
Autonomous Driving Sketch

```
#include <IRremote.h>
int RECV_PIN = 3;
int bulb = 10;
int motor = 11;
int itsONled[] = {0,0,0,0};
#define code1  0xFD08F7 // code received from button 1
#define code2  0xFD8877 // code received from button 2

IRrecv irrecv(RECV_PIN);
decode_results results;
void setup()
{
  irrecv.enableIRIn();  // Start the receiver
  pinMode(bulb, OUTPUT);
  pinMode(motor, OUTPUT);
}
void loop() {
  if (irrecv.decode(&results)) {
    unsigned int value = results.value;
    switch(value) {
      case code1:
        if(itsONled[1] == 1) {
          digitalWrite(bulb, LOW);
          itsONled[1] = 0;
        } else {
           digitalWrite(bulb, HIGH);
           itsONled[1] = 1;
        }
         break;
      case code2:
        if(itsONled[2] == 1) {
          digitalWrite(motor, LOW);
```

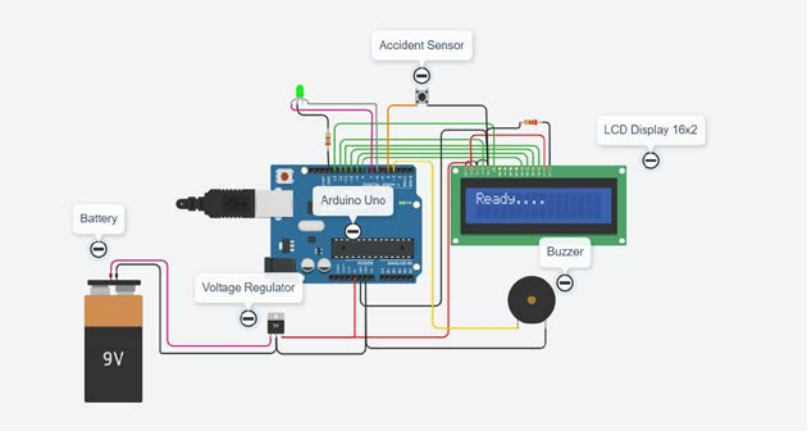
```
        itsONled[2] = 0;
      } else {
        digitalWrite(motor, HIGH);
        itsONled[2] = 1;
      }
      break;
  }
  irrecv.resume(); // Receive the next value
 }
}
```

6. Smart Accident detector with Emergency Distress

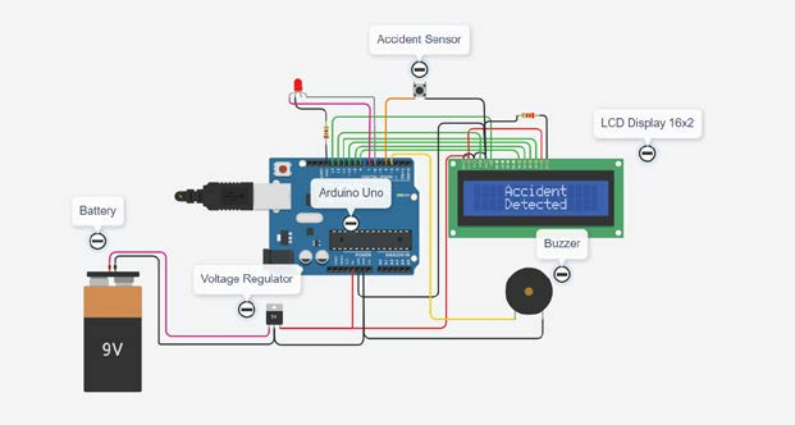
This smart device has the capacity to detect an impact and when it's triggered it identifies that an accident has occurred and immediately sends out notifications to the users whomsoever it concern. The circuit consists of the components such as a button, LED, Voltage Regulator, Buzzer, LCD display, Arduino Uno R3 and a battery.



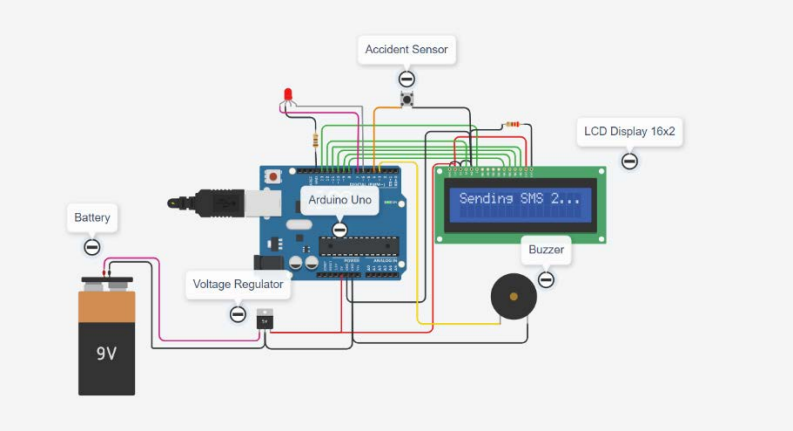
Once the car's engine is turned on this device is activated. It initializes the sensor and activated the system to the ready mode.



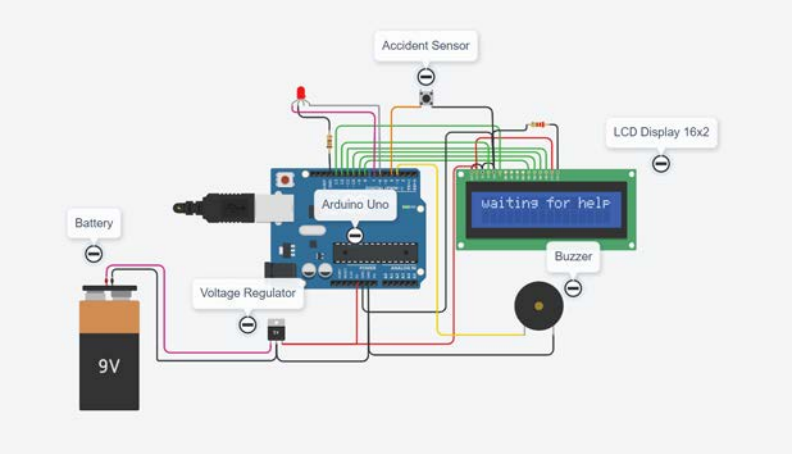
Once the system is ready it wait till the button is triggered which is an event of accident. If the button setup in the bumpers trigger then it automatically starts reacting like below



The accident is detected now it will send out the distress messages to the concern people for emergency help.



The SMS will be sent to 3 major accomplices, that is the Police, Emergency Care and Peter.



Once the messages are sent the system wait till the help arrive constantly giving out sound alarm to notify someone nearby for instant help.

Smart Accident Detector with Emergency Distress Sketch

```
#include <LiquidCrystal.h>

const int rs = 12, en = 13, d4 = 8, d5 = 9, d6 = 10, d7 = 11;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

const int buzzer = 3;

const int RED = 7;

const int GREEN = 6;

const int peizo = 4;
```

```
unsigned int i = 0;
#define PEIZO_ON LOW
#define PEIZO_OFF HIGH

#define BUZZER_ON {for(i=0;i<50;i++) {digitalWrite(buzzer, HIGH); delay(10);digitalWrite(buzzer, LOW);delay(10);}}
#define BUZZER_OFF {digitalWrite(buzzer, LOW);}

#define RED_ON { digitalWrite(RED, HIGH);   digitalWrite(GREEN, LOW);  }
#define GREEN_ON { digitalWrite(RED, LOW);   digitalWrite(GREEN, HIGH);  }

const char senderNumber1[] = "9029145142";
const char senderNumber2[] = "9029145142";
const char sms_body_1[] = "Accident detected";
#define PINNUMBER ""
void enter()
{
  Serial.print("\r\n");
}
void initialise_modem()
{
  enter(); Serial.print("AT"); enter();  delay(500);
  enter(); Serial.print("ATE0");  enter(); delay(500);
  enter(); Serial.print("AT+CPIN?");  enter(); delay(500);
  enter(); Serial.print("AT+CMGF=1"); enter(); delay(500);
  enter(); Serial.print("AT+CNMI=2,1"); enter(); delay(500);
  enter(); Serial.print("AT&W"); enter(); delay(500);
}
void send_sms(unsigned char user)
{
  enter(); Serial.print("AT+CMGS=\"");

  if(user==1) { Serial.print(senderNumber1);}
  else  { Serial.print(senderNumber2);}
```

```
 Serial.println("");
 delay(500);
 Serial.println(sms_body_1);
 Serial.println("Vehicle Number: TN-09 AP 1997");
 Serial.print("Latitude= XXXXXX.XX");
 Serial.print(" Longitude= YYYYYY.YY");
 Serial.write(0x1a);
 enter();
 delay(1000);
}
void setup()
{
 // put your setup code here, to run once:
 pinMode(peizo, INPUT_PULLUP);
 digitalWrite(peizo, HIGH);
 pinMode(GREEN,OUTPUT);
 pinMode(RED,OUTPUT);
 GREEN_ON;
 pinMode(buzzer, OUTPUT);
 BUZZER_OFF;
 Serial.begin(9600);
 while (!Serial) {
  ; // wait for serial port to connect. Needed for native USB port only
 }
 lcd.begin(16, 2);
 // Print a message to the LCD.
 delay(100);
 lcd.setCursor(0, 0);
 lcd.print("    Accident    ");
 lcd.setCursor(0, 1);
 lcd.print("Detection System");
 delay(1000);

 lcd.setCursor(0, 0);
```

```
  lcd.print("Initializing....");
  lcd.setCursor(0, 1);
  lcd.print("                ");
  initialise_modem();
  delay(1000);
  lcd.setCursor(0, 0);
  lcd.print("Ready....       ");
}
unsigned int loop_count = 0 , len = 0;
void loop()
{
  // put your main code here, to run repeatedly:
  if (digitalRead(peizo) == PEIZO_ON)
  {
    delay(1);
    if (digitalRead(peizo) == PEIZO_ON)
    {
      RED_ON;
      lcd.setCursor(0, 0);
      lcd.print("    Accident    ");
      lcd.setCursor(0, 1);
      lcd.print("    Detected    ");
        BUZZER_ON;
      delay(5000);
         lcd.setCursor(0, 0);
      lcd.print("Tracking GPS....");
      lcd.setCursor(0, 1);
      lcd.print("                ");
    lcd.setCursor(0,0);
  lcd.print("                ");
  lcd.setCursor(0, 1);
  lcd.print("                ");
        lcd.setCursor(0, 0);
        lcd.print("Lat: 1234.4558");
```

```
      lcd.print(" N ");
      lcd.setCursor(0, 1);
      lcd.print("Lon: 2258.7589");
      lcd.print(" E ");
     delay(1000);
    lcd.setCursor(0, 0);
    lcd.print("Sending SMS 1....");
    lcd.setCursor(0, 1);
    lcd.print("               ");
    send_sms(1);
    delay(5000);
    lcd.setCursor(0, 0);
    lcd.print("Sending SMS 2....");
    send_sms(2);
    delay(5000);
   lcd.setCursor(0, 0);
    lcd.print("SMS Sent.........");
    lcd.setCursor(0, 1);
    lcd.print("               ");
  BUZZER_OFF
    lcd.setCursor(0, 0);
    lcd.print("waiting for help");
    lcd.setCursor(0, 1);
    lcd.print("               ");
 while (1);
   }
 }
else
 {
 BUZZER_OFF;
  }
 delay(5);
}
```

RISK, RECOMMENDATIONS AND MITIGATION STRATEGY

1. Activity 1: Identify Expected Customers and Define Expected Use Cases

1. **Which types of people are expected customers for this device?**
   The Intelligent Car with Safety and Garage Automation is specifically used for the people with disabilities, old age and also the people who needs extra security for their car.

2. **Which types of organizations are expected customers for this device?**
   Many organization has already adapted this device in their cars but as of now it is designed for single user usage only as it needs more upgrades for organizational standards.

3. **How will the device be used?**
   ICSGA is a multi-purpose project, because he does not only monitor the car security of the user, but also acts as an emergency beacon to ask for assistance.

4. **Where geographically will the device be used?**
   ICSGA is meant to be used on a global scale, given the nature of the project itself.

5. **What physical environments will the device be used in?**
   ICSGA is an environment that links different devices all together, therefore the expected physical environments are both inside and outside.

6. **What dependencies on other systems will the device likely have?**
   ICSGA needs the Wi-Fi adapters, ADXL -335 etc. for GPS functionality and internet connectivity.

7. **How might attackers misuse and compromise the device within the context of the use case?**
   An attacker may compromise the device by overwriting the values provided by the sensors, which will result in a series of unexpected reactions by **ICSGA.**

2. Activity 2: Research Customer Cybersecurity Goals

1. **How will the IoT device interact with the physical world?**
   ICSGA is designed to interact with the physical world in a seamless way, because it can have many new connected to it so typically a user can modify this device and can interact with it in many ways. This device interacts to the users through push notification screens, smartphones etc.

2. **How will the IoT device need to be accessed, managed, and monitored by authorized people, processes, and other devices?**

   Since all the devices are connected to a smartphone, a customer is expected to access the application and connect each single device he'd like to an environment. So the only requirement on the user part of the application is to register and login, in order to use **ICSGA**.

3. **How will the IoT device's use of device cybersecurity capabilities be affected in terms of the device's availability, efficiency, and effectiveness?**
   In the eventuality of upgrades of the cybersecurity capabilities, the monitoring part of **ICSGA** may be restricted or inaccessible during the upgrading, but this won't stop the single devices connected to stop functioning either. An upgrade on a physical component however, would require the purchase of said component.

4. **What will the nature of the IoT device's data be?**
   The nature of the data generated by **ICSGA** will be of 3 types: Location History, Accident History, How many time the car has been accessed.

5. **What are the known cybersecurity requirements for the IoT device?**
   There are no known cybersecurity requirements other than the standard policies of data collection, storage, processing and privacy.

6. **What complexities will be introduced by the IoT device interacting with other devices, systems, and environments?**
   The only complexity added by introducing new IoT devices will be the knowledge of how actually the device works and how will be interacting with the environment provided by **ICSGA**.

3. Activity 3: Determine How to Address Customer Goals

1. **Which one or more of the following is a suitable means (or combination of means) to achieve the goal?**
   The IoT device can provide the technical means through its device cybersecurity

2. **How robustly must each technical means be implemented in order to achieve the cybersecurity goal?**
   The cybersecurity goals must be implemented on a software level, with the only exception of the single device weakness to cyberattacks.
   The only type of data that require a high cybersecurity lever is the authentication part, the rest of the data needs only a normal level of protection, keeping all the data types confidential and with a good level of integrity. The authentication will be a 2fA type for each smartphone that require access to an environment. Eventual broken updates will be rolled back as soon as the error is acknowledged to be created by the software itself, and not by a misuse of the IoT devices connected to it.

4. Activity 4: Plan for Adequate Support of Customer Goals

1. **What potential future use needs to be taken into account?**
   At this time, there is no life-span for every software updates, however may influence the processing speed of the smartphone, if the update is on the encryption side. In order to speed it up again, a new device might be needed. The same situation applies if there are new updates for the physical components of the environment.

2. **Should an established IoT platform be used instead of acquiring and integrating individual hardware, firmware, and software components?**
   Using an already established platform might be a possible solution in order to quickly expand the type and number of devices connected to an environment, however it may results in a higher level of restriction in order to access the data, and some other data, might not be even collected in the first place.

3. **Should any of the device cybersecurity capabilities be hardware-based?**
   No. The security capabilities should always be software based. However there may be exceptions to that rule.

4. **Does the hardware, firmware, or software (including the operating system) include unneeded device capabilities with cybersecurity implications? If so, can they be disabled to prevent misuse and exploitation?**
   At the time of writing, there aren't any unneeded cybersecurity implications.

5. **How is IoT device code protected from unauthorized access and tampering?**
   Code signing

6. **How can customers verify software integrity for the IoT device?**
   The Code signature validation can be used to verify software integrity.

7. **What verification is done to confirm that the security of third-party software used within the IoT device meets the customers' needs?**

Test the necessary executable code

8. **What measures are taken to minimize the vulnerabilities in released IoT device software?**
   The software to be configured to have a secure setting as default and has testing phases with it (Developer mode)

9. **What measures are taken to accept reports of possible IoT device software vulnerabilities and respond to them?**
   An automatic Vulnerability response program to be developed for the device.

10. **What processes are in place to assess and prioritize the remediation of all vulnerabilities in IoT device software?**
    By estimating the potential impact of the exploitation.

5. Activity 5: Define Approaches for Communicating to Customers

1. **What terminology will the customer understand?**
   The terminology used will change based on the feature that the environment belongs.

2. **How much information will the customer need?**
   Only the basic information that the customer can understand.

3. **How/where will the information be provided?**
   The information will be always provided as soon they're available on the application.

4. **How can the integrity of the information be verified?**
   The information will be provided by sensors put on the IoT devices.

6. Activity 6: Decide What to Communicate to Customers and How to Communicate It

Cybersecurity Risk-Related Assumptions

1. **Who were the expected customers?**
   The expected customers are elderly people, first time car learners/users and people with disabilities.

2. **How was the device intended to be used?**
   The original intention of the device is to be used as an emergency alarm in an account of accident and further more as a smart assistant with security options and ease of use like autonomous driving and autonomous garage etc.

3. **What types of environment would the device be used in?**
   The environment in which **ICSGA** will work is a hybrid between indoor and outdoor type-of environment.

4. **How would responsibilities be shared among the manufacturer, the customer, and others?**
   The responsibility will be shared based on the type of problem taken into account. If a user give the access credential on another person, it may modify the account settings or the connected devices, so the responsibilities falls on the user.
   If the devices is attacked successfully, the responsibilities falls on the software developers, and if a device is malfunctioning, then the responsibilities falls on the manufacturer.

Support and Lifespan Expectations

1. **How long do you intend to support the device?**
   As of now the life- span of the device is not been projected yet.

2. **When do you intend for device end-of-life to occur?**
   When the car is scrapped or if the user thinks that it has no purpose at some point the device end-of-life may occur.

3. **What functionality, if any, will the device have after support ends and at end-of-life?**
   Most likely all functionalities will be kept, however, there will be no more support.

4. **How can customers report suspected problems with cybersecurity implications, such as software vulnerabilities, to the manufacturer? Will reports be accepted after support ends? Will reports be accepted after end-of-life?**
   The customers can report problem directly using a direct contact with the support team, or by using the application. There will be no more support after the end-of-life of the device.

Technical and Non-Technical Means

1. **Which technical means can be provided**
   Very few technical means are proposed such as modification of contact numbers, distress message, time delay etc.

2. **Which technical or non-technical means should the customer provide themselves or consider providing themselves?**
   Only the basic requirements for Internet security is required.

3. **How is each of the technical and non-technical means expected to affect cybersecurity risk?**
   A heavy encryption of data may results in a slow decryption that will affect the speed of the application.

Device Composition and Capabilities

1. **What information do customers need on general cybersecurity-related aspects of the device, including device installation, configuration (including hardening), usage, management, maintenance, and disposal?**
   A tutorial on how to connect a device will be provided by the application or by the service support itself during the installation. The usage of all the connected devices will be on the user's manual of each of these devices.

2. **What is the potential effect on the device if the cybersecurity configuration is made more restrictive than the secure default?**
   The access to the information will be more difficult as it may requires different levels of authentication.

3. **What information do customers need about the sources of the device's software, firmware, hardware, and services?**
   The customer may not be interested in those information and this totally depends on the user type.

4. **What information do customers need on the device's operational characteristics so they can adequately secure the device? How should this information be made available?**
   Only the basic knowledge on how to use the application and the connected device is needed.

5. **What functions can the device perform?**
   Frequent monitoring of the given tasks and perform the actions.

6. **What data types can the device collect? What are the identities of all parties (including the manufacturer) that can access that data?**
   The data types collected by the device are the data that is gathered by all devices connected to it, the user data like accident history, security breaches and geographical coordinates.

7. **What are the identities of all parties (including the manufacturer) who have access to or any degree of control over the device?**
   As for now, the only party that has access to the information is the manufacturer.

Software and Firmware Updates

1. **Will updates be made available? If so, when will they be released?**
   The updates will be made available as soon as possible after a possible vulnerability/improvement is verified. It will be released as soon the issue is resolved.

2. **Under what circumstances will updates be issued?**
   When a vulnerability is found, or if an improvement is needed either by the manufacturer or the users.

3. **Which entity (e.g., customer, manufacturer, and third party) is responsible for performing updates? Or can the customer designate which entity will be responsible?**
   As for now, the updates will be done by the manufacturer, and the customer only needs to update the application then and there.

4. **How can customers verify and authenticate updates?**
   By actually testing the malfunctioning of the device/application or by using the new added features.

5. **What information should be communicated with each individual update?**
   Added features and vulnerabilities fixed and future upgrades.

Device Retirement Options

1. **Will customers want to transfer ownership of their devices to another party? If so, what do customers need to do so their user and configuration data on the device and associated systems (e.g., cloud-based services used by the device) are not accessible by the party who assumes ownership?**
   Users may want to transfer ownership of the device to other users but, if they want to keep their information secret they need to disconnect all the devices, delete their environment and log out the account.

2. **Will customers want to render their devices inoperable? If so, how can customers do that?**
   In order to render their devices inoperable, users need to disconnect all the connected devices. After that, those devices are inoperable by the application until they're reconnected again.

NODE-RED AUTOMATION ORCHESTRATIONS

This project is idealized by using the system implementation using MQTT and Node-RED. Node Red acts as a client for the MQTT broker and the data from here is published and received through MQTT broker. Further Node-RED is connected to MQTT broker using MQTT node in Node-RED. An android app which is developed that triggers our **ICSGA.**

1. Node-RED as Tool

IBM Node-RED is a flow-based programming tool, for wiring together hardware devices, APIs and online services and is supported by the JS Foundation. It supports browser-based flow editor and it is built on Node.js. The Node-RED can be cloud supported or may be run locally. The flows created in Node-RED are stored using JSON format. Node-RED has three parts:

- The palette contains all of the nodes that are installed and available to use.
- The main workspace is where flows are developed by dragging nodes from the palette and wiring them together.
- The sidebar contains panels that provide a number of useful tools within the editor.

2. Editor

So, the browser-based Editor is separated in several sections. At the top, you have the Header, which has the Deploy button and the Menu.

On the left, you have the Palette, containing all the available Nodes that you have installed. In the middle, you have the Workspace where you can drag and drop nodes and wire them together to create a flow. On the right, you have the Sidebar which contains panels that provides useful tools.

For example, the information panel which shows details about a selected node. There is also the Debug panel which displays messages that are passed to a Debug node connected to a flow. We'll look at the Debug node shortly, and go over the benefits of using it when creating flows.

3. MQTT

Message Queue Telemetry Transport (MQTT) is simple publish/subscribe and lightweight messaging protocol built for M2M and IoT and is ideal for constrained networks. It is real time that is specifically what makes it perfect for IoT applications. Below figure illustrates the connectivity of MQTT.

How MQTT Works-

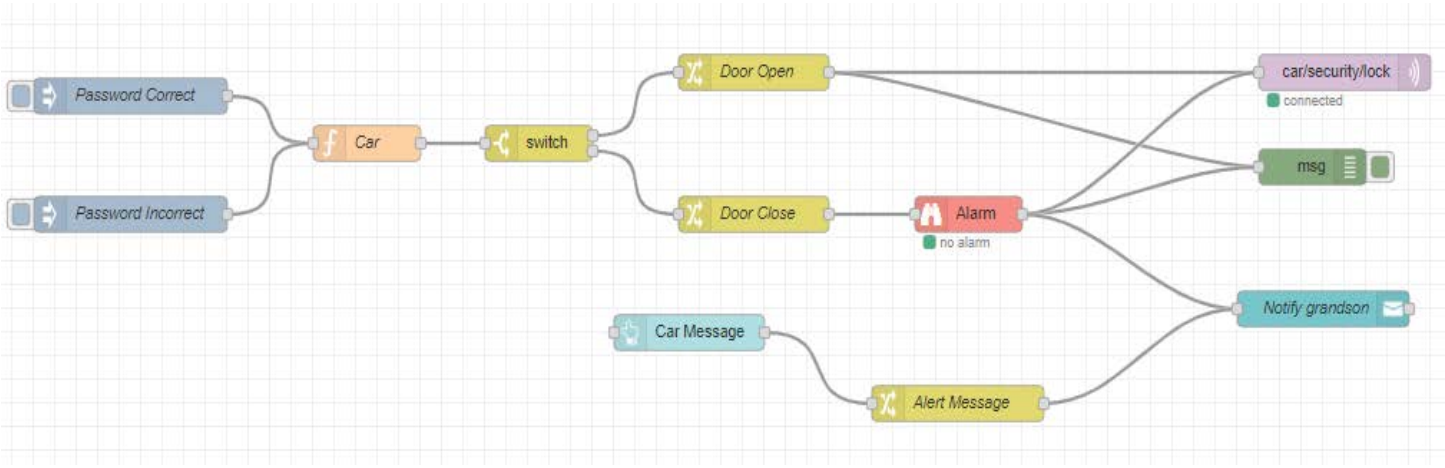
MQTT is client/server based protocol. MQTT server is called a broker and the clients are connected devices. So:

- Publish − When a device (a client) sends its data to the broker
- Subscribe − When a device (a client) subscribes for data from the broker.

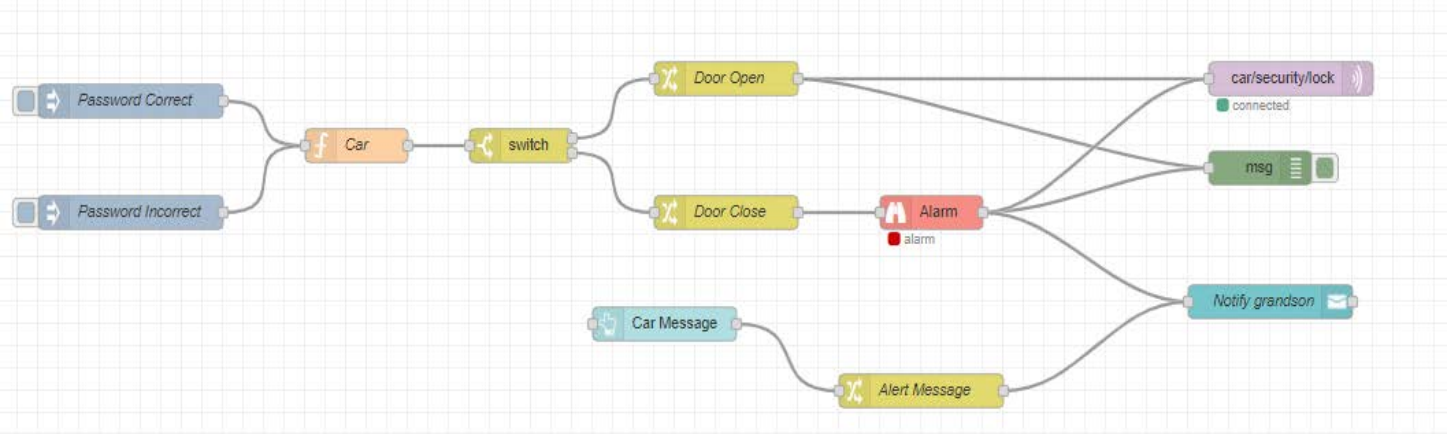
4. Node Red Flow Implementation

1. Keypad Password Lock for the Car

The circuit of the keypad lock may look simple but the way it is implemented in the software functionality is different. So this was achieved by using the node-red flows to clearly illustrated how the device work as we see in the below work flow,



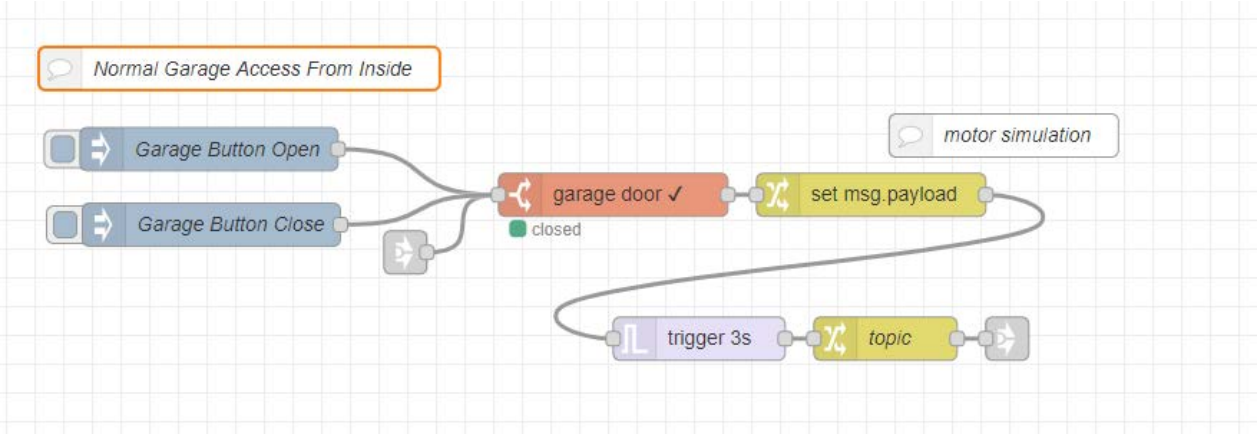
There are two instances where the password is correct and the password is wrong. If the correct password is injected without any issues the door will open without tripping out the alarm. If in case the wrong password is injected then at that time the alarm will be tripped and starts alerting the users same as we see in the below flow,



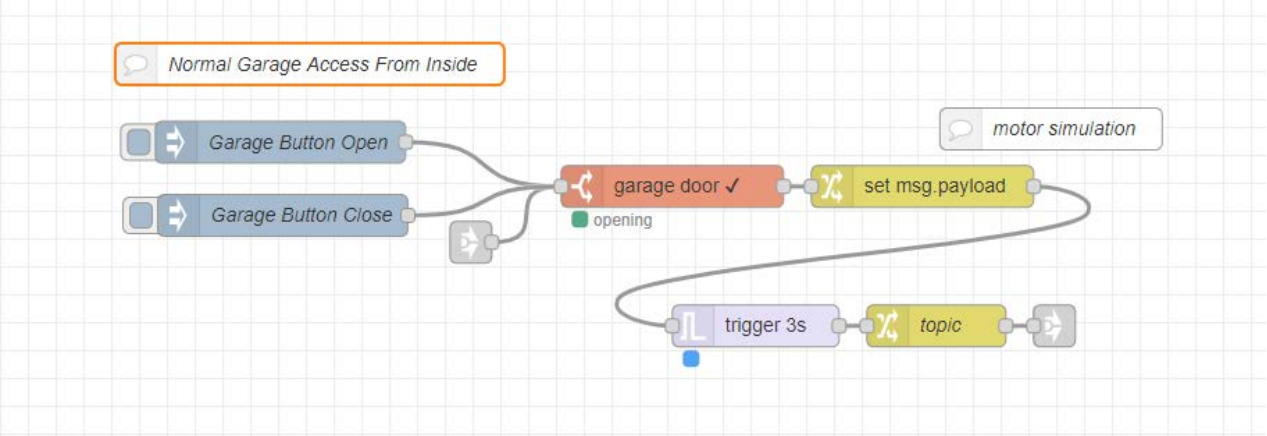
We can see the alarm being tripped and then the notification will be sent to the users which we will see in the dashboard of their smartphone shortly. So this can happen only if the wrong password being injected in any case. The user can also identify that they have entered the wrong password just by seeing the screen present in the circuit and by the sound of the alarm.

2. Garage Opener (Manual)

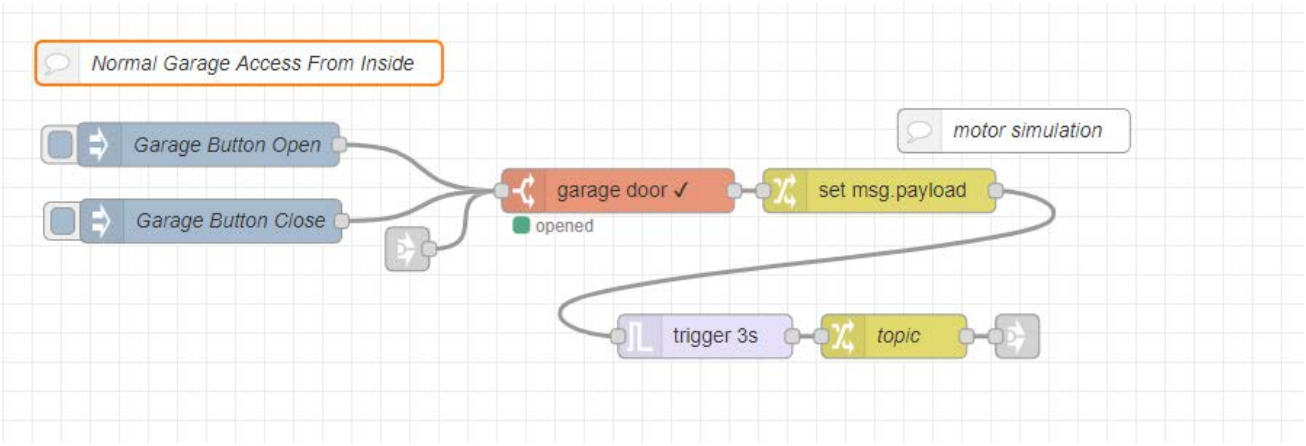
Always having a manual method to operate the garage from the inside is necessary, because anyone can use the garage for other purposes like laundry, storage etc. So as per the simple circuit proposed a clear flow of node-red garage opener is developed and as we see in the flow below it is said to be in closed state,



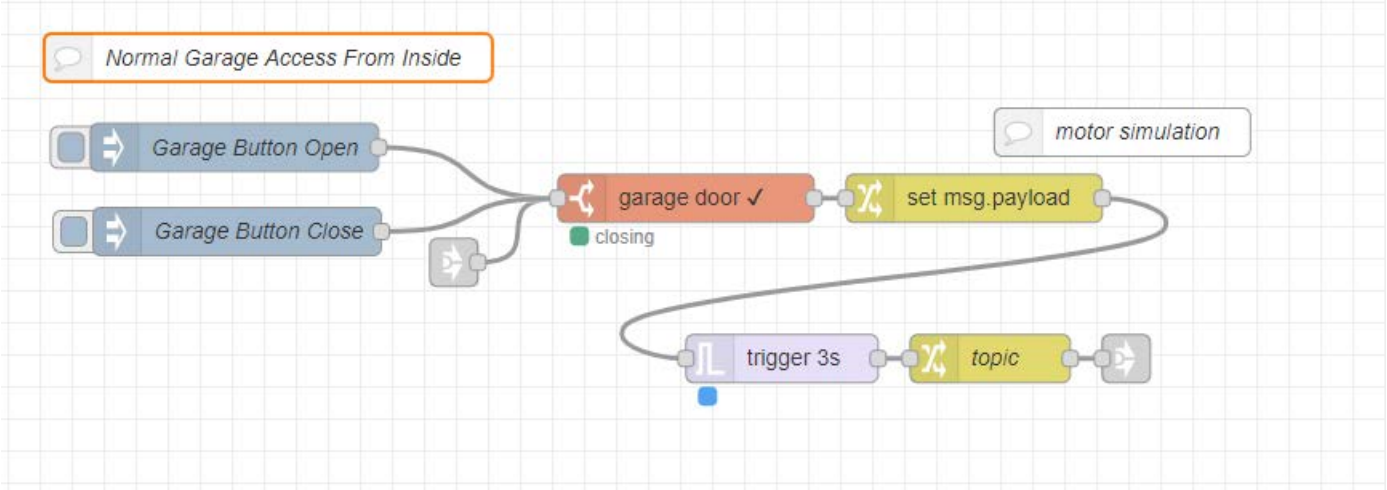
For this flow we can understand that the garage open is injected when the user needs to open it. When the button is injected then the device initiates the door to open like below,



As we can see in the above flow the garage door is currently opening, once it's fully opened,



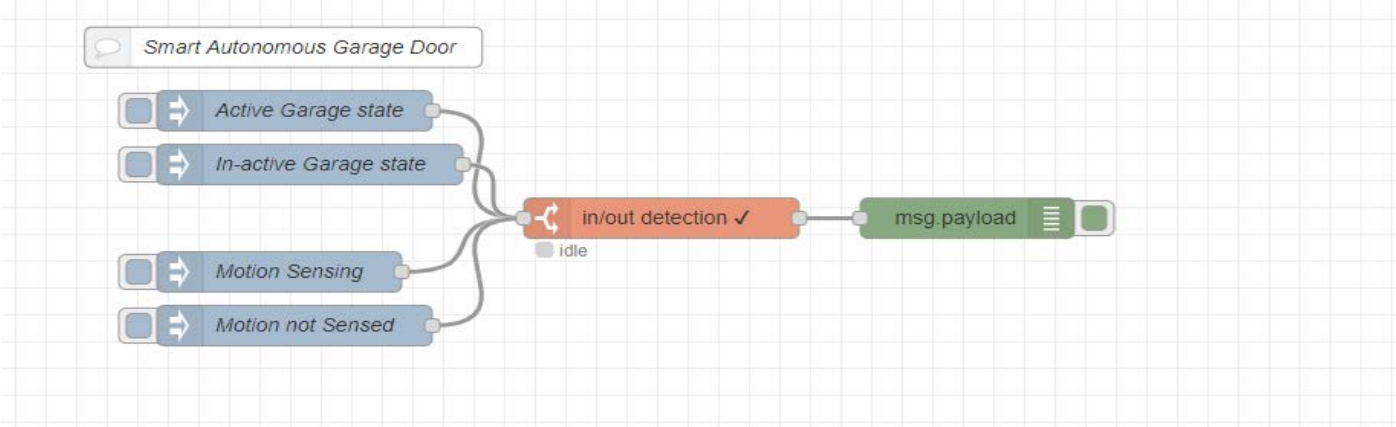
Again when the user injects the button to close then the flow goes to the closing phase like,



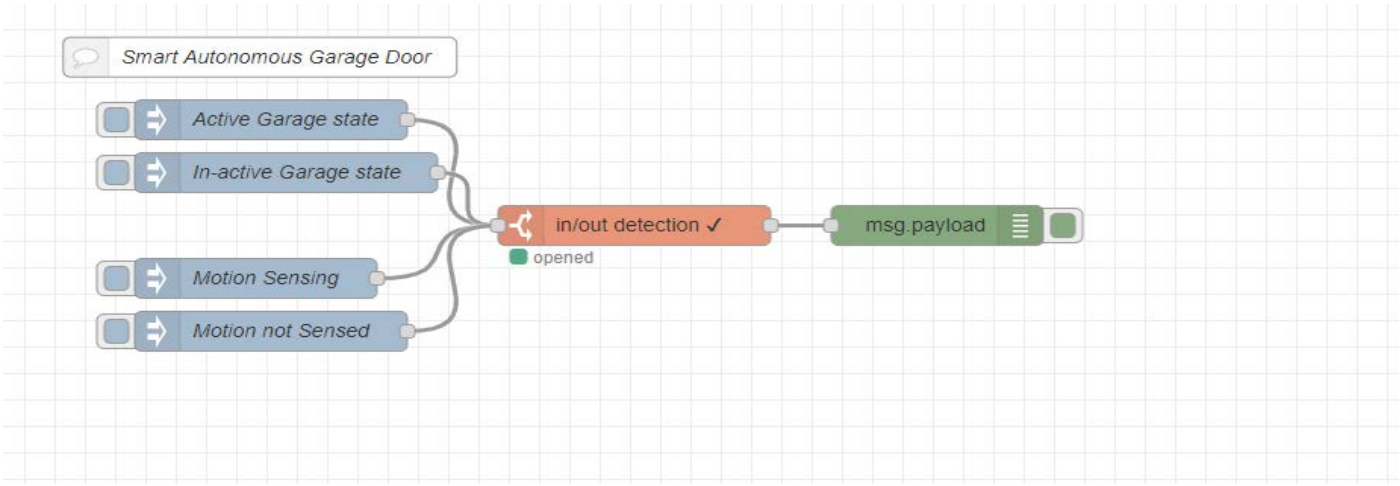
Then again it is turned back to the default closed state after the door is closed just like the first flow.

3. Smart Autonomous Garage Door:

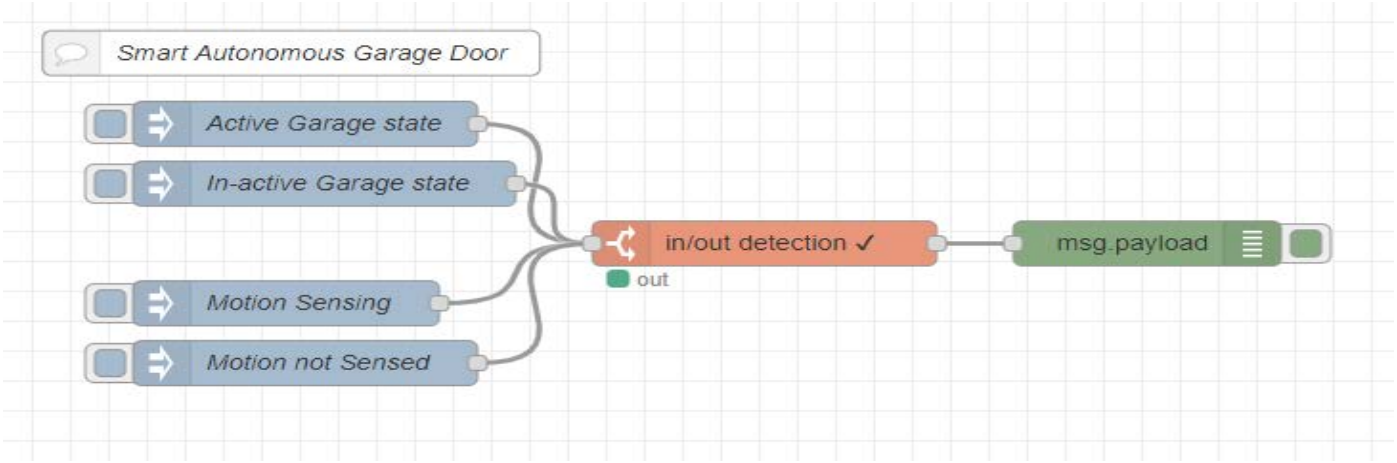
The autonomous garage opener is a smart device that is just enough to do the garage work as it automatically opens the garage door and closes it once the job is done. But as per the practicality we have implemented both the manual and the automated one. So this device is automated and a simple circuit. So, in order to understand its true capacity we should see how the flows are structured with right idea as we can see below it the door is in the 'idle' state which means the door is not opened by the other device too and no motion is detected so far.



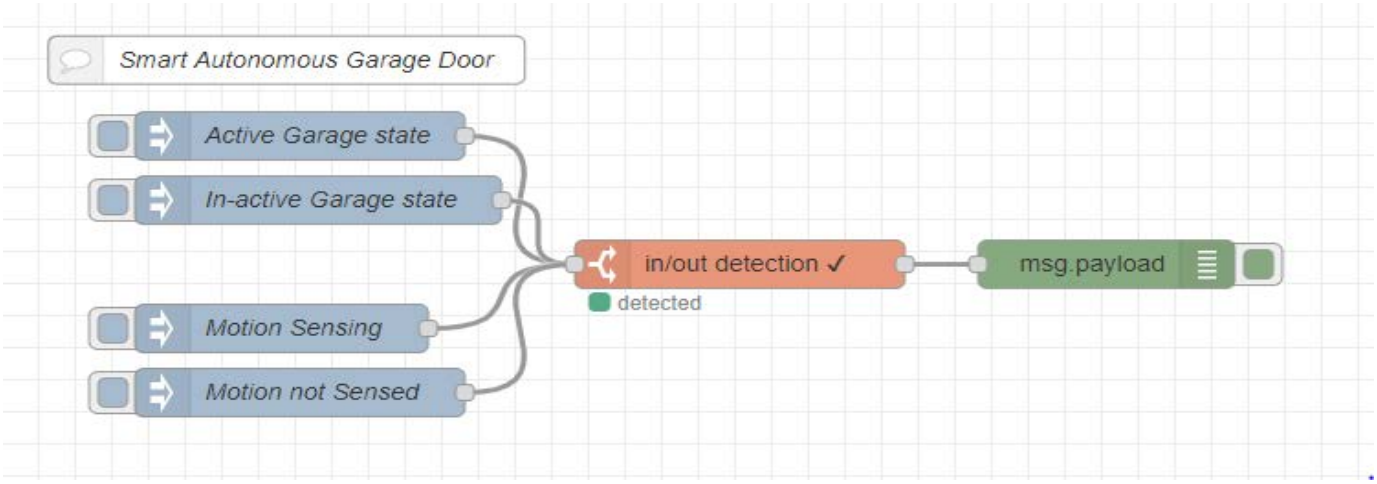
The first inject node indicated whether the garage door is active (Which means if its opening /opened by the other device). If the door is opened/opening this device understands it state like opened.



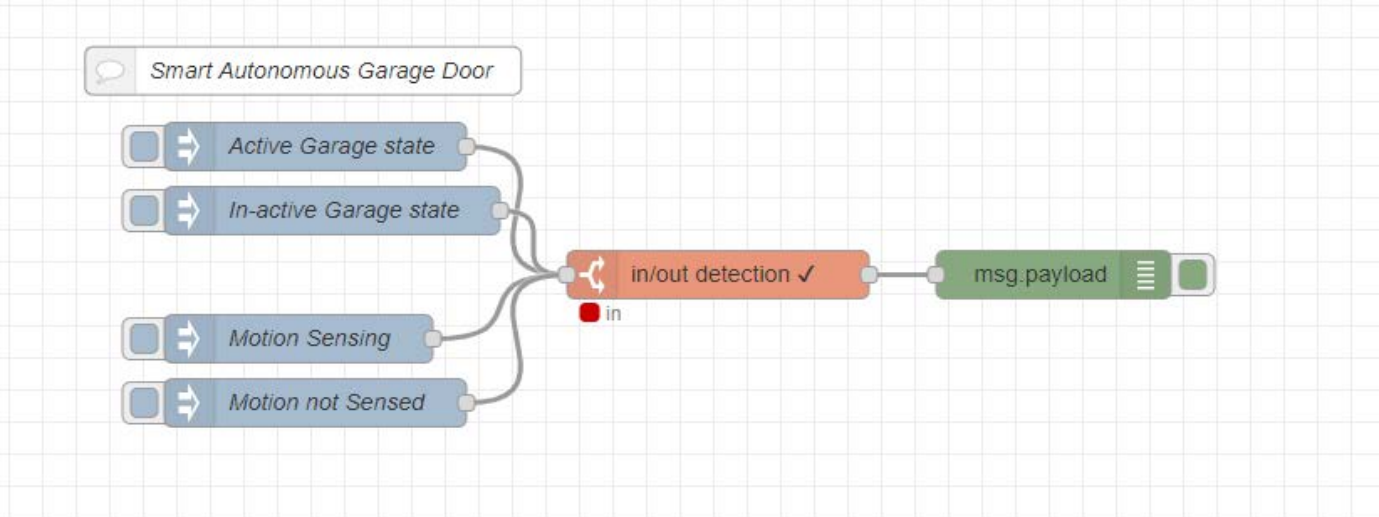
If the garage door is in active state(opened) and if the motion sensor picks up the motion from outside then the device understands that it should not close the door because the car is moving out, as we can see in the flow,



It will only close the door until the car reaches a safe distance outside the garage. When the car arrives back to the garage the motion sensor picks up the movement it understands the motion detection and goes to ‘detected’ phase and activated the garage open button like the following flow,



For Safety of the car until the car completely reaches into the garage it must not close the door and squeeze the car and damage it, so as we can see in the below flow it understands that the car is going inside the garage and goes to the ‘IN’ phase and then once the motion sensor goes to idle then the door will close automatically.

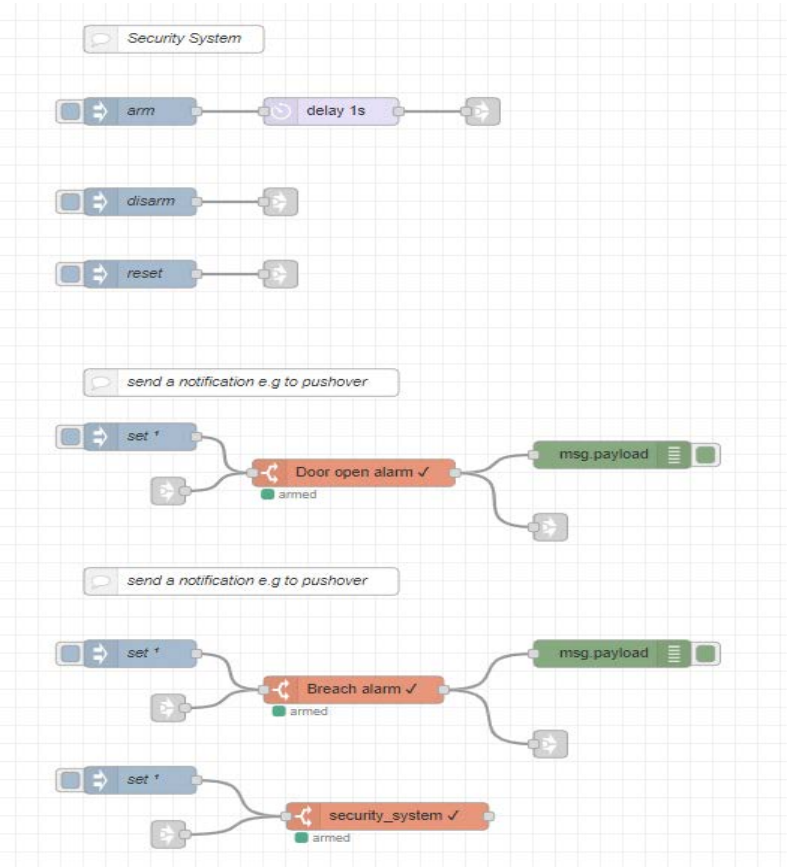


Once the car is parked inside the garage the door will be set back to the idle state by the device.

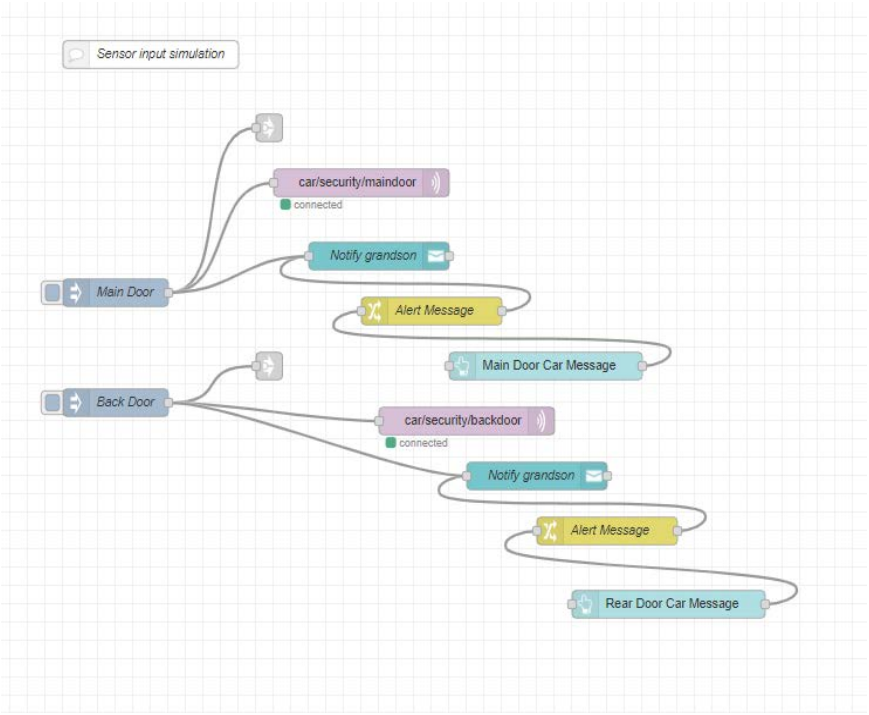
4. Smart Car Security:

This security device is an all-around security for the car parked while the user is not around. The circuit is pretty simple as it detects force and sounds the alarm, by seeing the node-red flows we can understand the real scenario behind the total security system. This is setup in each and every door of the car even in the back trunk door.

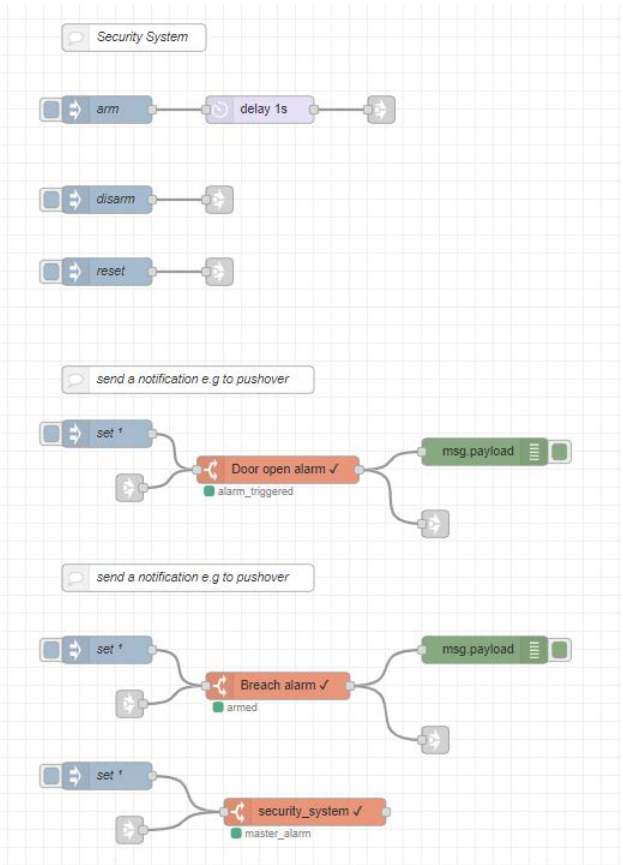
Now when the user leaves the car he arms the alarm for safety and the flow is injected and go to armed phase.



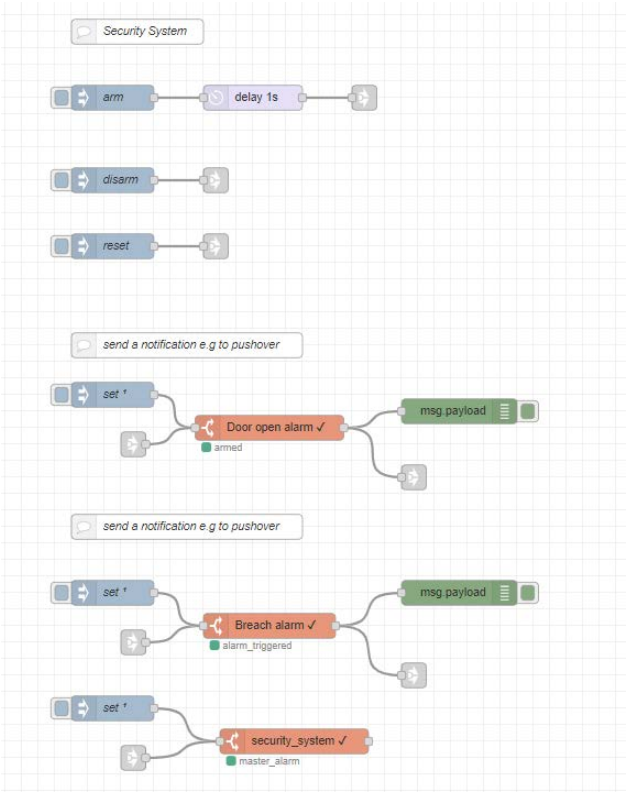
Once the device is armed the flow further shows the door handles and the windows all connected with multiple sensors wired together into two categories that is the Main Door (Front Door and Drivers Door) and then the Back Doors as we see in the below flows,



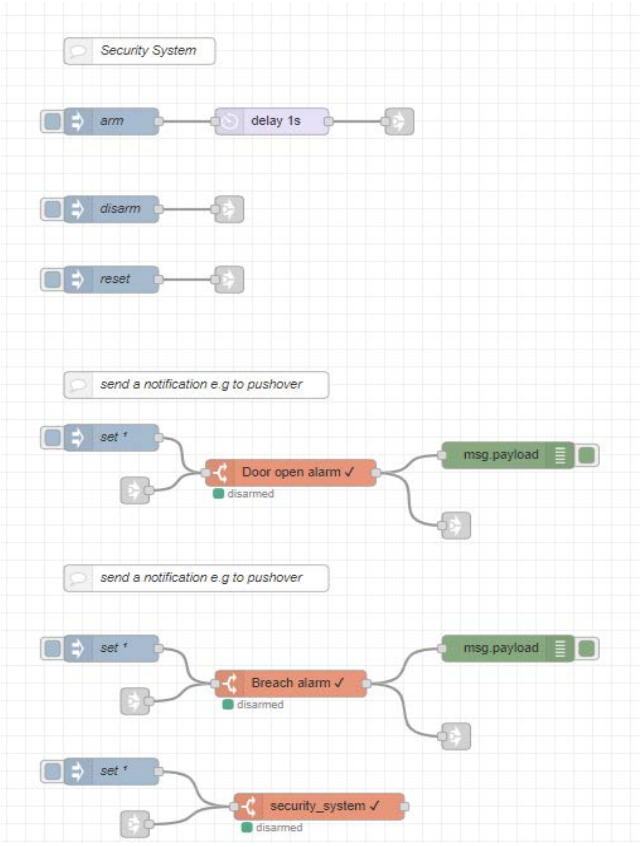
Now if any of the sensors detects any unnatural external force during an attempt to steal then the alarm is triggered. If someone tried to break-in through the Main doors (if the try or already broke in state) then the flow turns to alarm triggered phase as we see below and then sends out the message.



If in case it was the back doors then, the breach alarm is triggered and the message to the users is instantly sent.

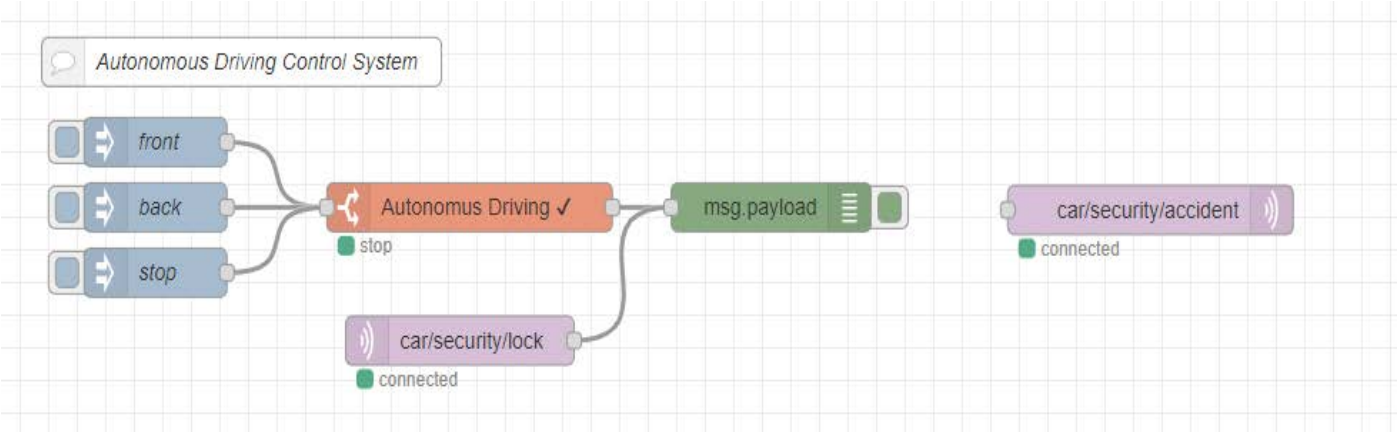


When there is no attempt for robbery of the car when the user comes to the car to take it away using the remote he can just simply disarm the security system, as we see the flow when disarm is injected the flow goes to the disarm state.

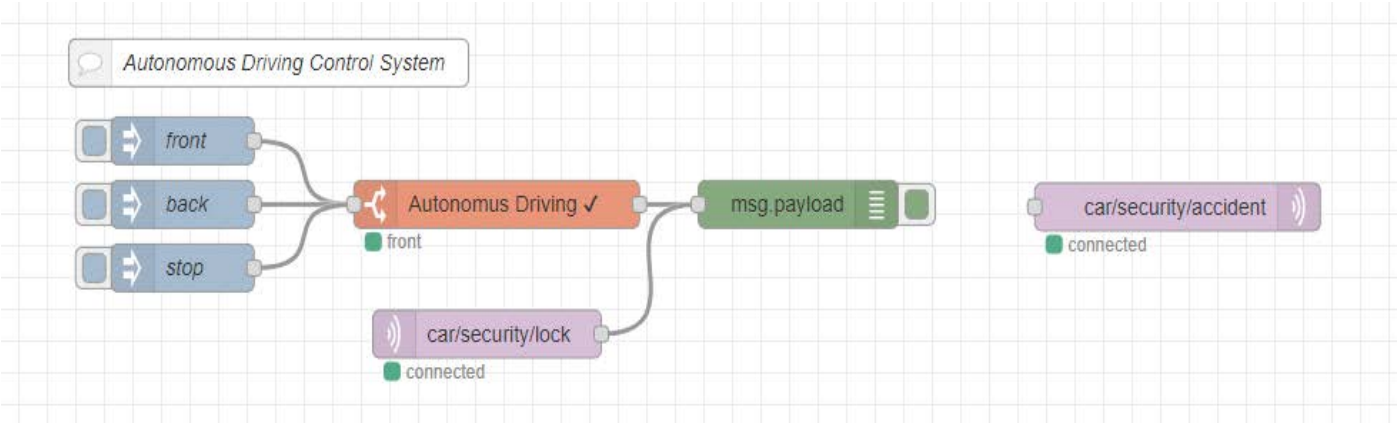


5. *Autonomous driving:*

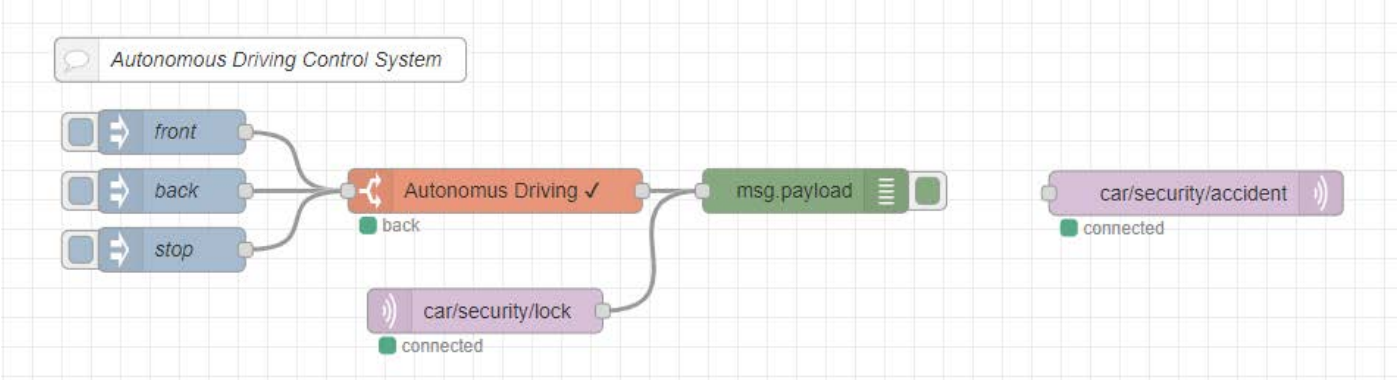
This smart device just been added to control the car's front (acceleration) and back (reverse) functionality. The cars headlights can be turned automatically through the car remote. Same way this flows will explain how the cars are controlled by the user for autonomous driving.



Initially the car is in the stopped state as it is not triggered by the user, once the user turns the car on using the remote and pushes the front button, then the car accelerated forward. As we can see in the below flow the flow phase will turn to front.



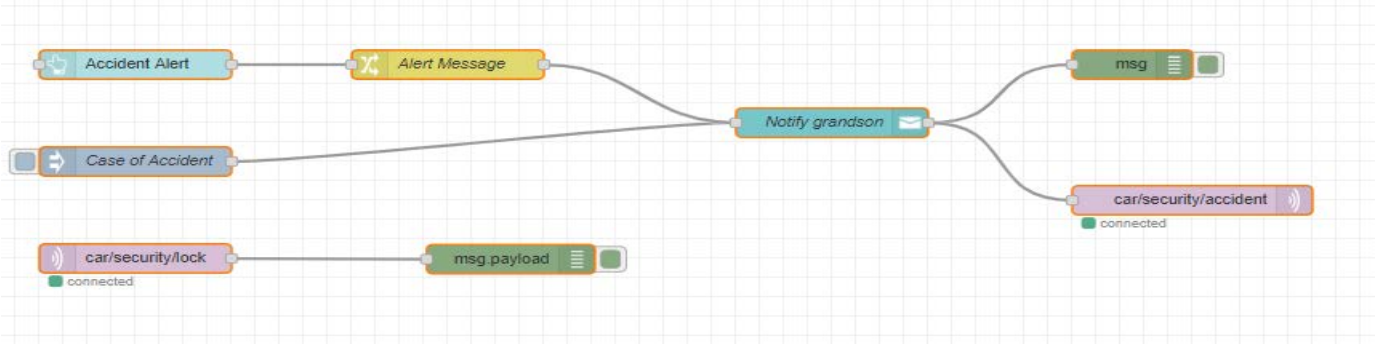
Then if the user presses the back button to move the car in reverse then the flow will turn into the back phase,



Once the car comes to the halt it again goes to the stop phase until the user uses his remote function to call the car forward or back.

6. Smart Accident Detector with emergency distress:

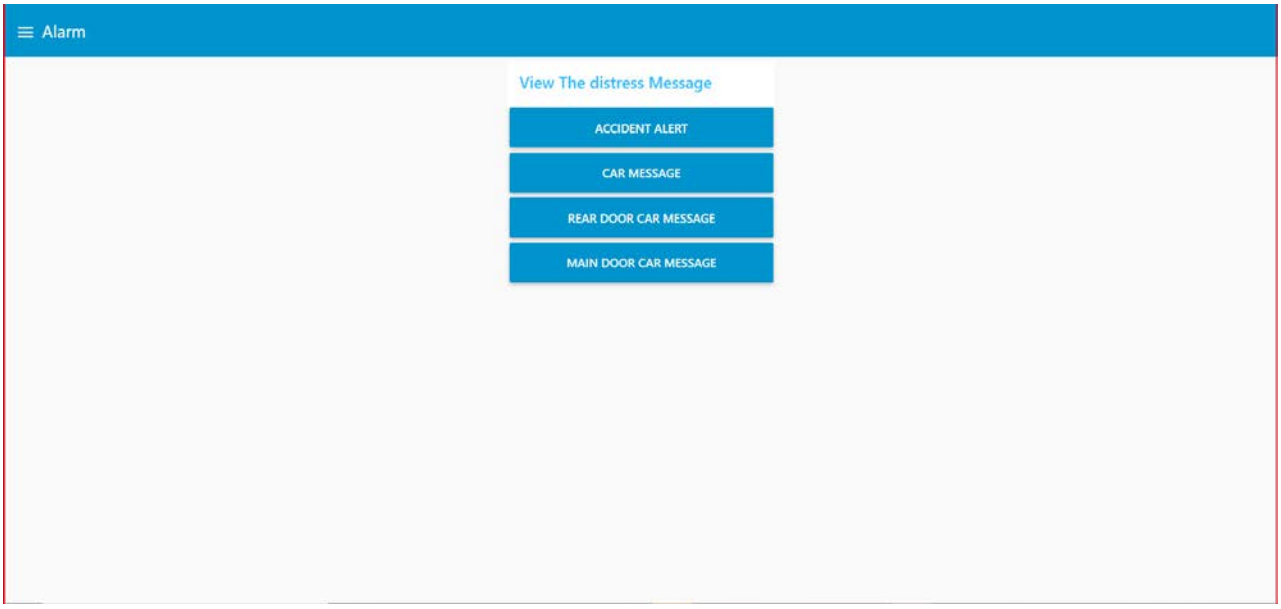
By seeing the circuit of the accident detector we can understand how the system works implementing it on the node- red is deemed to be simple here in our case as certain functionalities cannot be clearly illustrated visually, on the below flow we can get an idea of how the smart device work,



When the button is pushed from the device then it will automatically send out the distress message to the user's preference.

5. Node-RED Dashboard

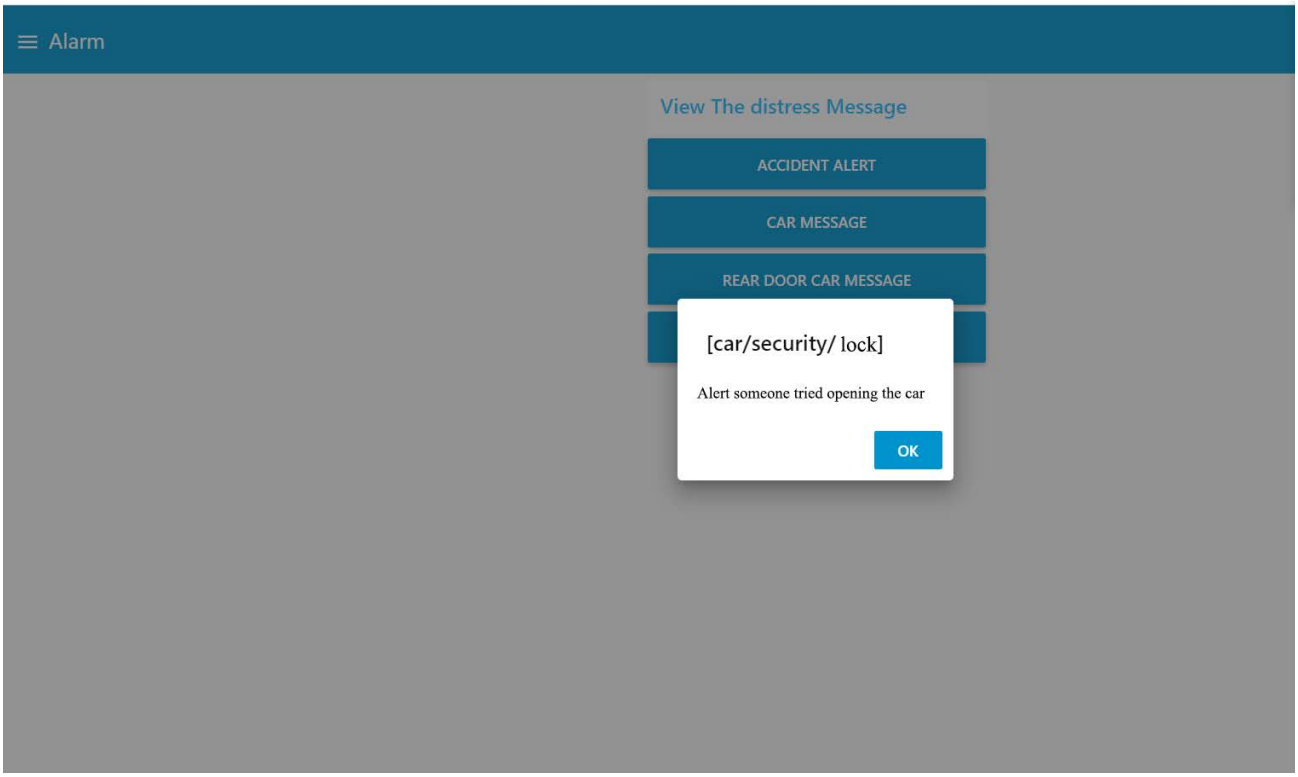
Node-RED's dashboard nodes provide a comprehensive set of UI components for building basic dashboards suitable for the Internet of Things (IoT) – offering graphs, gauges, basic text as well as sliders and inputs. However, there will always be situations when you need something custom. The template node is the solution. Below we have mentioned the UI of the dashboard for the above mentioned flows and we can see how the messages and notifications are being sent by the smart devices to the users.



This is how the user interface screen looks like from the user's perspective.

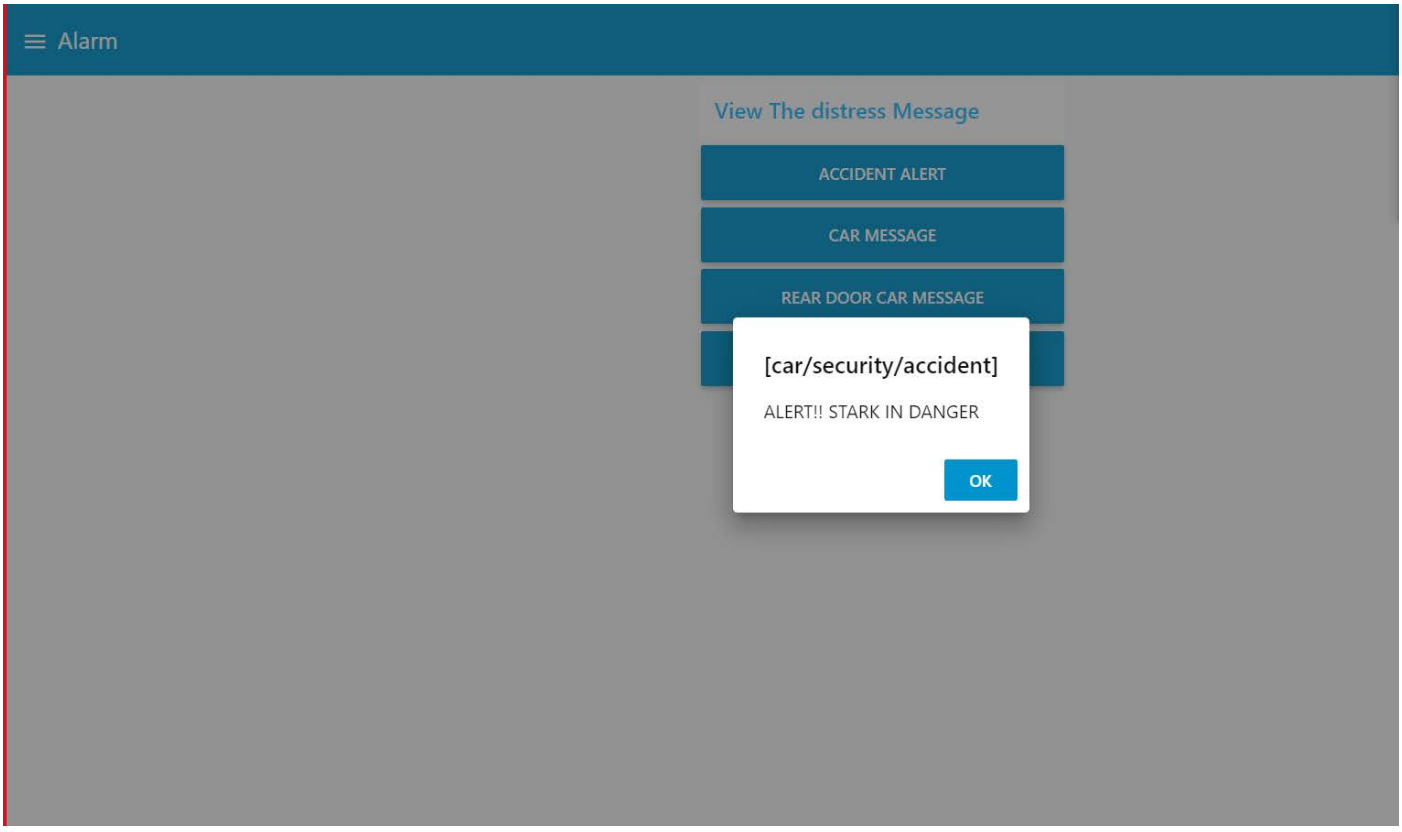
1. *Keypad Password Lock breach:*

The alert message sent to Stark's phone and Peter's Phone.



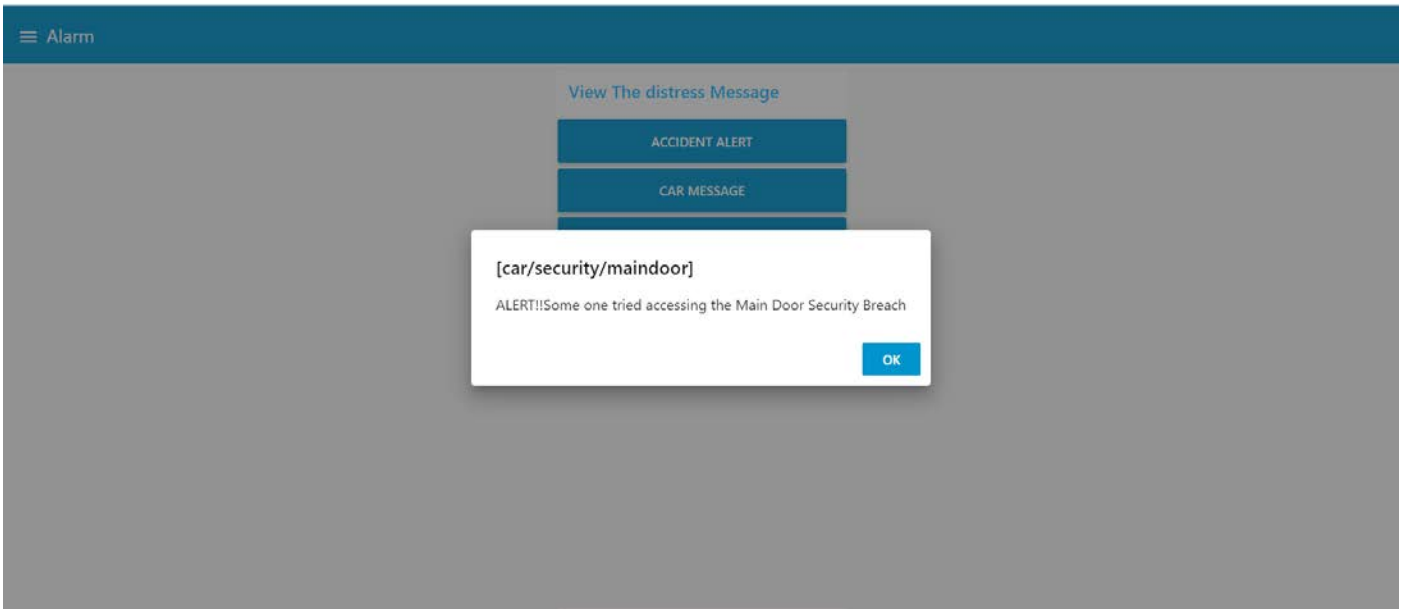
2. *Accident Detector with Emergency Distress Message:*

After an Impact is detected the alert message is being sent to Peter that Stark is in Danger

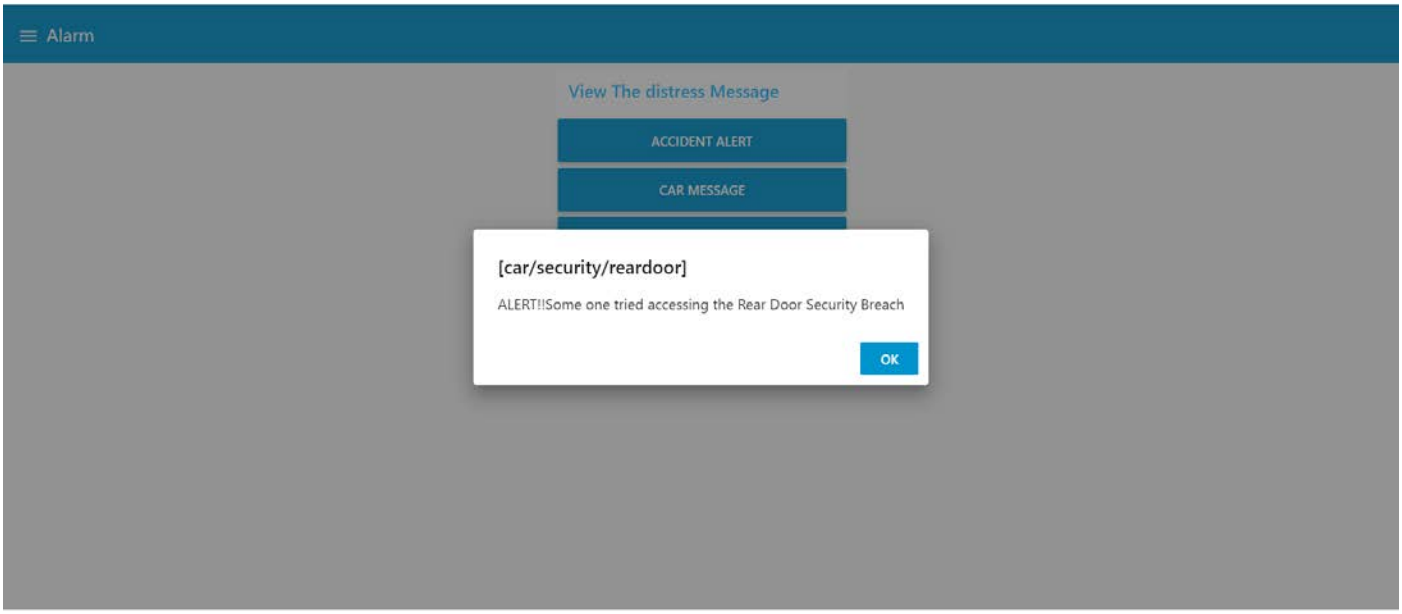


3. Smart Car Security Message:

When the Main door of the car is opened or attempted to be opened Peter will receive this notification.



When the Rear Door of the car is opened or attempted to be opened Peter will receive this notification.



CONCLUSION AND FUTURE IMPROVEMENTS FOR THE DEVICES

1. Conclusion:

So in this project we have seen how a rare car can be protected and how easily it can be used daily that too by an elderly person. I would not say that everyone must need this device but anyone with a rare car and need some extra care can try and implement this IOT concept as it is functional, cheap, easy to install and easy to maintain overall.

Since this project is done completely as a virtual implementation certain idea may not be properly conveyed and I would not say that we have completely used all the potential and functionalities of the nod-red. But to suffice the concept and the proposed scenario we think that this explanation given in the above documentation and the node – red flows arrangement might be feasible to satisfy the proposed concept.

2. Future Improvements:

As we could see that since this is a virtual based project in order to make this a real-time serious working prototype we would need more components and sensors for the exact implementation of the proposed idea. Things like WIFI adapters, Raspberry PI, Phillips Hue can be added physically to the already derived smart devices as it is not available in the tinker cad. I would say that when done physical the idea proposed can be satisfied by using the node-red's full potential and also by using other factors such as an Amazon AWS, cloud services to store all the data collected like the accident history, driven locations, no of password attempts etc. The device can be improved by adding cameras to snap the pictures of the people who try to steal the car, additional sensors for the functionalities like lane management alert, automatic braking, speed monitor etc.