5/18/2024

SMART HOME

Team 5

INTRODUCTION TO MECHATRONICS SYSTEMS

Table of contents:

TABLE OF FIGURES:

CHAPTER 1: Literature Survey

Our project is a smart home prototype, where many facilities are provided such as smoke detectors, mobile-controlled LED lights, laser security, an automatic garage door, an elevator, a solar panel and a solar tracker, and an RFID lock. This home was designed to provide maximum efficiency with the lowest consumption and cost.

Garage door:

Our plan at first was to use a shuttle roller mechanism for the garage door, but after further inspection, we found difficulties regarding the motor to be used, segment sizes, and spacing inside the model. The motor needed to have enough strength to move the segments and coincidently within our budget, therefore we chose to use a 180-degree servo motor. This slightly changed the mechanism but it allowed us to use significantly larger door segments where they would be able to be printed accurately and have enough strength to tolerate the torque of the motor. This also allowed us to save a lot of space which made the model look more aesthetic. Accordingly, we landed on our current mechanism which works by linking the servo head with a link that slides the segments on a guide between two extreme positions (60 degrees).

Elevator:

The elevator we designed to move along two floors using push buttons. We didn't face any real issues with the elevator's design mechanically, however, the problems we faced were mainly with the circuitry. The issues we faced were that the ultrasonic at first didn't read proper values but was solved using a delay in our code. The mechanism works by utilizing the ability of the ultrasonic to measure distances where if the proper button was called while the ultrasonic was in the appropriate extreme position, the DC motor attached to a pulley would rotate to move the cabin either up or down. Using our calculations, we determined that the usage of a counterweight was unnecessary as the motor is capable of tolerating the elevator independently.

Solar Panel:

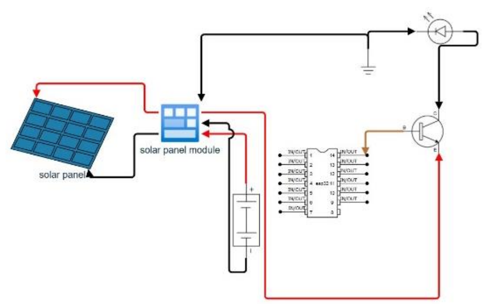

Fig 1.1
solar panel circuit

This mechanism didn't give us issues at all, we attached a module to the solar panel allowing it to charge up a battery. This was further advanced by adding a solar tracking mechanism we found from a trusted source. The solar tracking mechanism uses two LDRs and compares their values to rotate the solar panel using a servo motor toward the strongest source of light. The previous photo is a circuit diagram of how the solar panel controls the LED lights using a battery and the ESP.

RFID and gate:

Radiofrequency identification "RFID" is used to unlock the gate, where if the RFID reads the proper card, a servo with a gearhead attached to it slides open the gate, the gate has a gear rack attached to it to allow the conversion of the rotary motion of the servo to a linear motion for the gate. This initial design was altered by adding an LCD to the circuit allowing the user to visibly know if the proper card was identified. Despite the amplification of the design, the circuit increased in complexity which was our main issue for a while until we built the circuit correctly.

ESP module:

The ESP was used to elevate the "smart" factor in our model; It provides the user the capability to control devices and receive alerts using a WIFI connection. The device functions a lot like an Arduino but with further advancements which caused a slight problem at first until we learned how it properly functioned. One of these issues was the need to choose an appropriate port with sufficient power supply. Here are the systems connected to the ESP:

1)Laser Security System:

This system consists of a laser emitter, LDR, and a buzzer. Once the laser is breached the LDR detects the change causing the buzzer to ring and the ESP to send an alert to the user's phone. The ESP also controls the switching of the system where the user can fully close or open the system as he desires. The initial design did not have the ESP features which made the system less applicable and more complex with the circuitry.

2) Smoke sensor:

A smoke sensor is a critical component that is an essential part of your home's safety system, detecting smoke or gas in the air to signal potential fire hazards. When smoke is detected, the sensor triggers an alarm alerting occupants to the danger and can automatically alert you through your smartphone or buzzer.

3) LED Lights:

The lighting fixtures connected to these ESP modules enable remote control and automation of the lighting system.

CHAPTER 2: Mechanical systems

Rack and pinion gate:

To produce linear motion of rack from the rotational motion of the pinion

Linear velocity of rack (V_r) must be the same as the pitch line velocity of the pinion.

$$v_R = v_t = Rp \times w_P = \frac{dp}{2} \times wp = \frac{dp}{2} \times n \times \frac{2\pi}{60\times1000} = \frac{\pi \times d \times n}{60000} \text{ m/sec}$$

The linear displacement of the rack related to the angular displacement of the pinion (θ_P)

$$S_{\text{Rack}} = \frac{\text{Dp}}{2} \times \theta\text{p}$$

The center distance concept doesn't apply directly in rack and pinion because the center of the rack is infinite but the pitch circle of the pinion must be tangent to the pitch line of the rack.

Initial pitch diameter =30mm = 30x 0.03937=1.1811 inch,

speed=5rpm, and number of teeth of pinion = 20 teeth,

Pd (diametral pitch)= $\frac{N}{Dp} = \frac{20}{1.1811} = 16.93\ teeth\ per\ inch,$

$It's\ not\ \ standard\ so\ we\ will\ change\ pitch\ diameter\ to\ 31.75\ mm = 1.25\ inch\ ,$

$$P_d = \frac{N}{D_p} = \frac{20}{1.25} = 16\ inch,$$

$pitch\ line\ to\ back = 0.25\ inch = 6.35mm$

$distance\ from\ back\ of\ rack\ to\ pinion\ centerline = pitch\ line\ to\ back + pinion\ radius$

$$= 0.25 + \frac{1.25}{2} = 0.879inch = 22.2mm,$$

$$V\ rack = \frac{Dp}{2} \times np = \frac{1.25}{2} \times 5(rpm) \times 2\pi \times \frac{1}{12(inch)} = 1.635\frac{ft}{min} = \frac{8.3058mm}{sec},$$

$time\ to\ move\ rack\ 10mm\ distance\ (V = \frac{distance}{Time}\quad time = \frac{distance}{velocity} = \frac{100}{8.3058}$ =12 sec)

, θp (pinion revolution number to move 100mm of rack)

$$S_{\text{Rack}} = \frac{\text{Dp}}{2} \times \theta\text{p}$$

$$\theta\text{p} = \frac{S_{rack} \times 2}{\text{DP}} = \frac{100 \times 2}{31{,}75} = \frac{6.3}{2\pi} = 1{,}003 \text{ rev}$$

Garage:

Our studies did not cover the design aspects of a sectional garage door as it's complicated. Further research or specialized training may be necessary to gain to accurately design the door.

Elevator:

Maximum force acting on the cabin:

The walls of the elevator's cabin have the least area, subjecting it to the most stress. We have calculated the area of the walls subjected to the force, assumed a proper factor of safety of 2, and the yield strength of the cabin is known. Real-life functioning elevators usually have a factor of safety of at least 13, however, this is just a model, making our assumption valid.

area = length x width x number of walls = 0.07 x 0.002 x 3= $0.00042m^2$

stress= 7 MPa & FOS= 2

$Fmax_1 = \frac{stress\ x\ area}{FOS}$ = 1.47 x 10^-3 N (This is the maximum allowable force that the walls could tolerate)

Force generated by motor:

The rope is attached at the bottom of the cabin. The force supplied from the motor's torque is to be calculated using the torque rating provided by the motor's datasheet and the pulley's radius to convert the torque to a force. By subtracting the maximum allowable force that the cabin could handle from the force supplied from the motor we could obtain a proper value for the maximum allowable weight that the elevator could lift.

Torque = 0.3384 kg.cm = 0.033198 Nm

Radius of wheel = 3.25 cm = 0.0325 m

Torque = $Fmax_2$ x radius

0.033191= $Fmax_2$ x 0.0325

$Fmax_2$ = 1N

F = $Fmax_2 - Fmax_1$ = 1 - $1.47*10^{-3}$= 0.99853 N (The maximum weight that could be lifted)

Tension on the rope:

In this segment, we check if the rope can handle the tensile stress from the forces acting on it "weight of the cabin", and "force from the motor". We sum the previously mentioned forces and insert them into the design equation. The area of the rope is also calculated and the yield strength is known.

d = 1 mm

area = $\frac{\pi}{4}$ x d^2 = 0.785 mm^2

Tension in rope = 1 N + (0.04 x 9.81 + 0.99853 N) = 2.39 N

F = $\frac{Sy \ x \ area}{FOS}$ = 2.39 = $\frac{20 \ x \ 0.785}{FOS}$

FOS = 6.57 (Safe)

Bending on the motor's shaft:

The motor shaft has a force acting on its end which may cause bending, so we acquired the shaft dimensions and yield strength from the motor's datasheet and combined all the forces acting on the shaft which contributed to finding the moment generated. Using the moment, the greatest point of bending "Y" and finally the inertia of the shaft's cross-section, the bending stress was obtained allowing us to calculate the factor of safety.

Shaft length = 10 mm

Shaft diameter = 4 mm

Sy(shaft) = 240 MPa

Total force = (wheel + rope + cabin) x 9.81 = (0.02 + 0.003 + 0.04) x 9.81 = 0.62 N

M = Total force x Shaft length = 6.2 N.mm

Y = 2mm

I = $\frac{\pi \ x \ d^4}{64}$= 12.56 mm^4

Bending stress = $\frac{M \ x \ Y}{I}$ = 0.987 MPa

FOS = $\frac{Sy}{Stress}$ = $\frac{240}{0.987}$ = 243 (Extremely safe)

CHAPTER 3: Control Systems

Elevator:

The control system of an elevator is an arrangement designed to ensure smooth and efficient operation, providing passengers with safe and reliable transportation between different floors of a building. At its core, the elevator control system consists of sensors, controllers, and actuators working to regulate the elevator's motion. The closed-loop control system employs feedback mechanisms to continuously monitor the elevator's position and velocity, comparing them to the desired setpoints. This feedback loop allows the controller to make real-time adjustments to the elevator's motor speed and direction, ensuring precise floor-to-floor positioning and smooth acceleration and deceleration. The closed-loop block diagram typically includes components such as an ultrasonic sensor, a controller unit (motor driver) implemented with the Arduino, and the elevator's mechanical system. This interconnected system enables the elevator to respond dynamically to changing conditions, ensuring passenger comfort and safety throughout the journey.

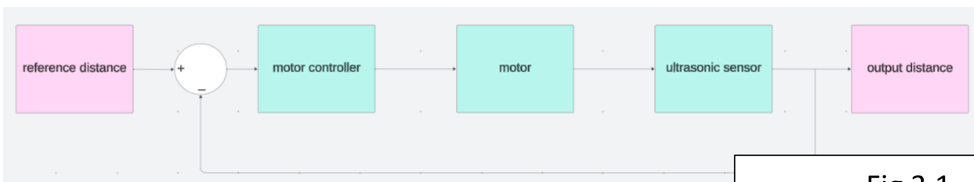

Fig 3.1
elevator block diagram

Solar tracker:

The control system of a solar tracker is a critical component in maximizing the efficiency of solar energy capture by orienting solar panels towards the sun. This system typically utilizes LDRs to detect the position of the sun relative to the tracker's location, as well as feedback mechanisms to adjust the orientation of the solar panels or mirrors accordingly. By continuously monitoring the sun's position and comparing it to predetermined setpoints, the control system calculates the optimal angle for the solar panels to maximize sunlight exposure throughout the day. This closed-loop control system ensures that the solar tracker accurately tracks the sun's movement, optimizing energy generation. The closed-loop block diagram typically includes components such as sun position sensors, a controller unit (in our case it’s the Arduino code), a servo motor, and the solar panel. This interconnected system enables the solar tracker to dynamically adjust its orientation, enhancing energy output and overall system efficiency.

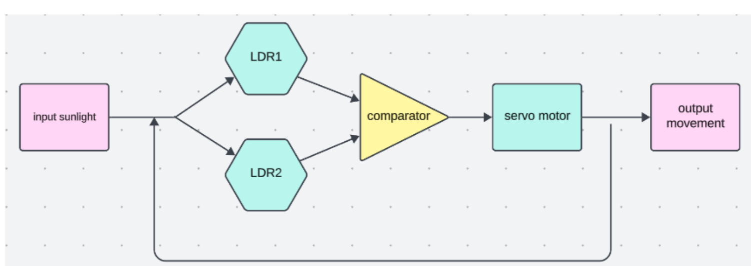

Fig 3.2
solar tracker block diagram

Laser:

The control system of an Intrusion Detection System (IDS) laser security system is perfect for detecting unauthorized access or intrusions into protected areas. This system typically comprises laser emitters, LDRs, and a central control unit (in our case an ESP module). Laser emitters project beams across the area to be monitored, while LDRs positioned on the opposite side detect any interruptions in these beams caused by objects passing through. The control unit processes the sensor data, analyzing patterns and triggering alarms when intrusions are detected. The closed-loop block diagram of an IDS laser security system involves feedback mechanisms that continuously monitor sensor readings and compare them to predetermined thresholds indicative of intrusion. This feedback loop enables real-time adjustments to the system's sensitivity and response, ensuring accurate detection while minimizing false alarms.

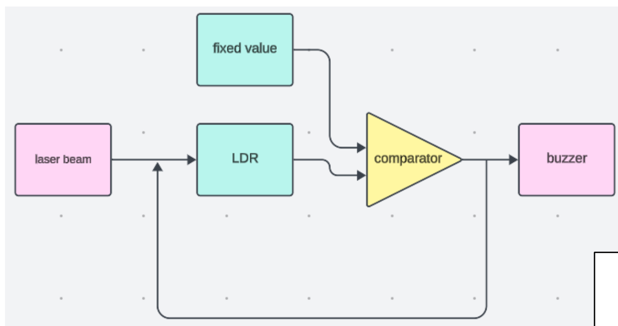

Fig 3.3
laser block diagram

Smoke sensor:

The control system of a smoke and gas detection system plays a crucial role in safeguarding against potential hazards such as fire and harmful gas leaks. This system typically consists of sensors strategically placed in areas prone to smoke or gas accumulation, along with a central control unit (ESP module) responsible for monitoring and managing sensor data. When smoke or gas is detected, the sensors send signals to the control unit, which then initiates appropriate response actions, such as sounding alarms and triggering emergency notifications. The closed-loop block diagram of a smoke and gas detection system involves continuous monitoring of sensor readings, with feedback mechanisms adjusting system parameters based on environmental conditions and user-defined thresholds. This closed-loop control ensures timely detection of smoke or gas events, prompt response, and effective qualification of potential risks to occupants and property.

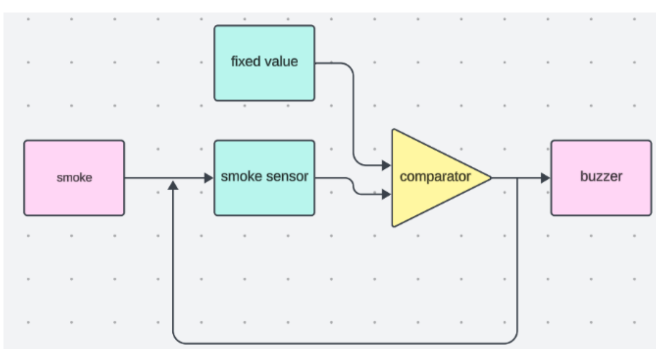

Fig 3.4
smoke sensor block diagram

CHAPTER 4: Codes, Simulations, &Results

We strategically divided our circuits between three Arduinos and an ESP module according to the placement and the available number of pins in each microcontroller.

ARDUINO 1: Garage and Solar tracker

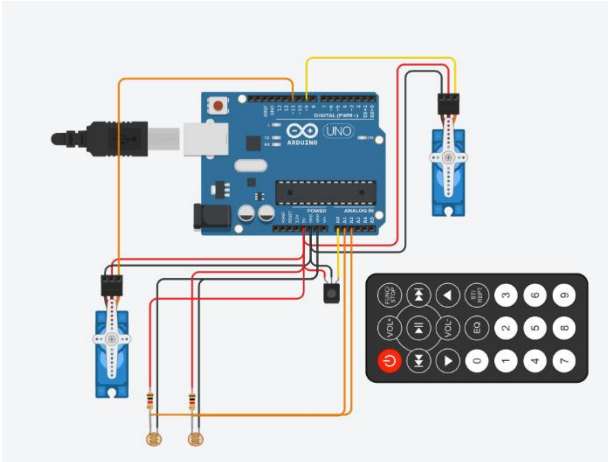

Fig 4.1.1 garage and solar tracker circuit

```
#include <IRremote.hpp>
#define IR_RECEIVE_PIN A0
#include <Servo.h>
Servo servo1;
int i = 62;
 #define LDR1 A1
#define LDR2 A2
#define error 10
int Spoint = 90;
Servo servo2;
void setup() {
  Serial.begin(9600);
  servo1.attach(9);
  servo1.write(62);
  IrReceiver.begin(IR_RECEIVE_PIN, ENABLE_LED_FEEDBACK);
  servo2.attach(11);
  servo2.write(Spoint);
  delay(1000);
}
void loop() {
 if (IrReceiver.decode()) {
    Serial.println(IrReceiver.decodedIRData.decodedRawData, HEX);
    IrReceiver.resume();
    if(IrReceiver.decodedIRData.decodedRawData==0xAD52FF00){
      while (i>=22) {
      servo1.write(i);
      delay(50);
      i=i-2;
      }
    }
    if(IrReceiver.decodedIRData.decodedRawData==0xE718FF00){
      while (i<=62) {
      servo1.write(i);
      delay(50);
      i=i+2;
```

```
      }
    }
 }
  int ldr1 = analogRead(LDR1);
  int ldr2 = analogRead(LDR2);
  int value1 = abs(ldr1 - ldr2);
  int value2 = abs(ldr2 - ldr1);
  if (value1 <= error || value2 <= error) {
  } else {
    if (ldr1 > ldr2 && Spoint > 0) {
      Spoint = constrain(Spoint - 1, 0, 180);
    }
    if (ldr1 < ldr2 && Spoint < 180) {
      Spoint = constrain(Spoint + 1, 0, 180);
    }
  }
  servo2.write(Spoint);
  delay(80);
}
```

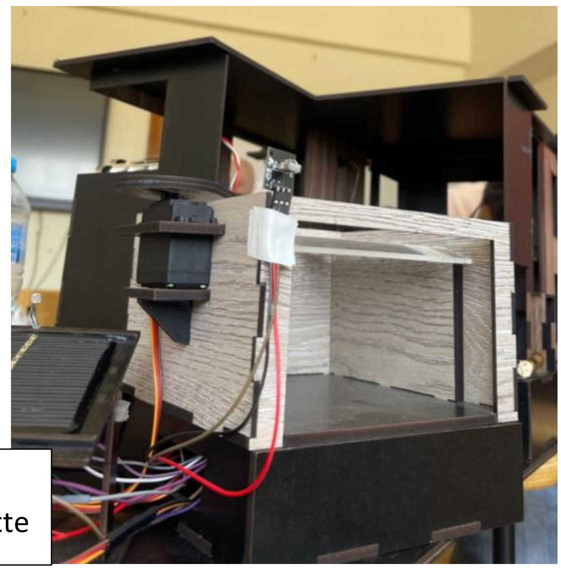

Fig 4.1.2
Garage maquette

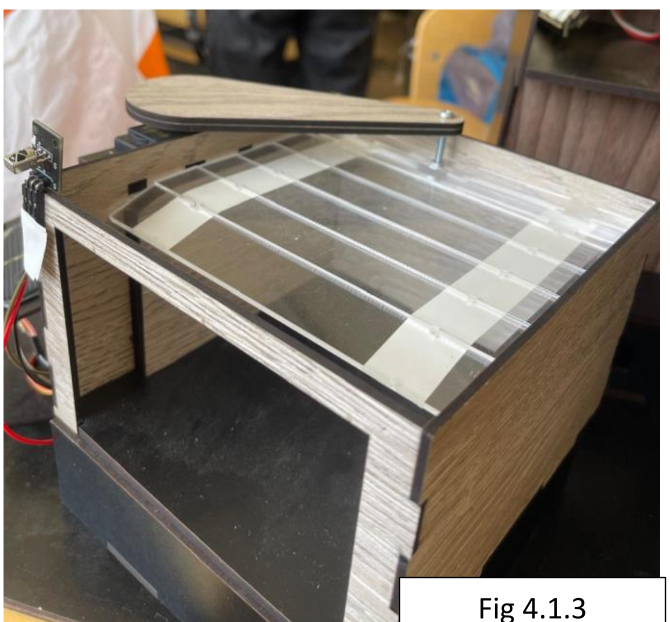

Fig 4.1.3
garage maquette

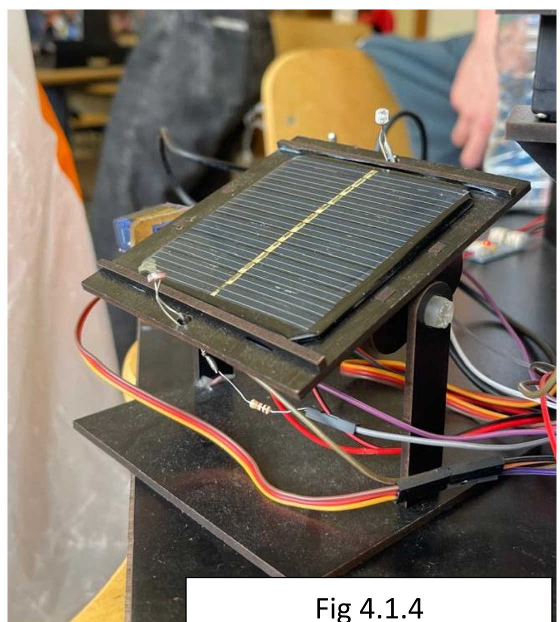

Fig 4.1.4
solar tracker maquette

ESP MODULE: (laser security, smoke sensor, and LED lights)

```
#define BLYNK_PRINT Serial
#define BLYNK_TEMPLATE_ID "TMPL2xGVVPtOa"
#define BLYNK_TEMPLATE_NAME "Test"
#define BLYNK_AUTH_TOKEN "BUO-1dpSDUzTOOlm5MJj50PVQLwMNCof"
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
char ssid[] = "Zaid";  //add netwrk name
char pass[] = "zaid1230";   //add network password
int Buzzer = 21;
int SW = 32;
int Laser = 13;
int LDR = 35;
int flag = 0;
int red = 19;
int Buzzer2 = 16;
int sensor = 34;
int Value = 0;
void setup()
  pinMode(Buzzer, OUTPUT);
  pinMode(LDR, INPUT);
  pinMode(Laser, OUTPUT);
  pinMode(SW, INPUT);
  pinMode(red, OUTPUT);
  pinMode(Buzzer2, OUTPUT);
  digitalWrite(red, LOW);
  Serial.begin(115200);
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
}
void loop() {
  Blynk.run();


  digitalWrite(Buzzer2, LOW);
  Value = analogRead(sensor);
  Serial.print("Smoke= ");
  Serial.println(Value);
  if (digitalRead(Laser)){
    delay(300);
    int LDRStatus = analogRead(LDR);
    Serial.print("LDR= ");
    Serial.println(LDRStatus);
    if (LDRStatus <=100|| flag==1) {
```

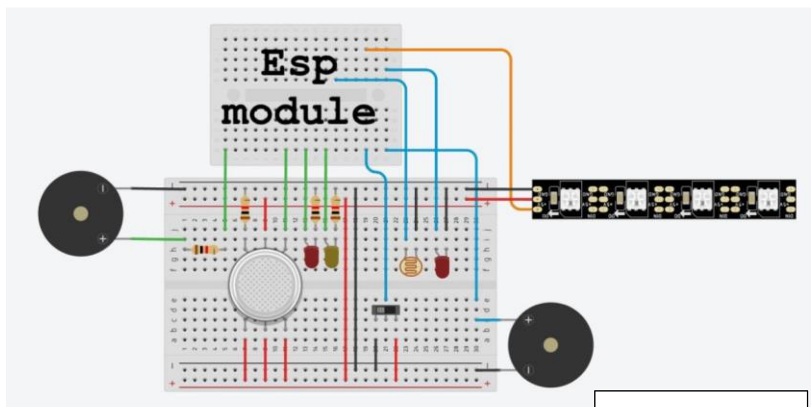

Fig 4.2.1
ESP circuit

```
      flag = 1;
      digitalWrite(Buzzer, HIGH);
      Blynk.logEvent("laser_security");
      tone(Buzzer,15000);
      delay(100);
      noTone(Buzzer);
    }
    else {
      digitalWrite(Buzzer, LOW);
      flag==0;
   }
  }
  else{
    digitalWrite(Laser, LOW);
    digitalWrite(Buzzer, LOW);
    flag = 0;
  }
  if (Value >= 550) {
    digitalWrite(red, HIGH);
    Blynk.logEvent("smoke_sensor");
    tone(Buzzer2, 1000);
    delay(200);
    noTone(Buzzer2);
  }
  else {
    digitalWrite(red, LOW);
  }
  delay(100);
}
```

Fig 4.2.2
laser maquette

Fig 4.2.3
smoke sensor maquette

ARDUINO 2: elevator:

```
int input1 = 4;
int input2 = 3;
int enable = 5;
int up = 6;
int down = 11;
int trig = 8;
int echo = 9;
int floorheight = 11;
long ultrasonic();
void setup() {
  pinMode(input1, OUTPUT);
  pinMode(input2, OUTPUT);
  pinMode(enable, OUTPUT);
  pinMode(up, INPUT);
  pinMode(down, INPUT);
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  analogWrite(enable,10);
  Serial.begin(9600);
}
void loop() {
  Serial.println(digitalRead(up));
  Serial.println(digitalRead(down));
  long distance = ultrasonic();
  if (digitalRead(up) == LOW) {
    while (distance < floorheight) {
      digitalWrite(input1, HIGH);
      digitalWrite(input2, LOW);
      distance = ultrasonic();
      delay(200);
    }
    digitalWrite(input1, LOW);
  }
  if (digitalRead(down) == LOW) {
    while (distance > 3) {
      digitalWrite(input1, LOW);
      digitalWrite(input2, HIGH);
      distance = ultrasonic();
      delay(200);
    }
    digitalWrite(input2, LOW);
  }
}
```

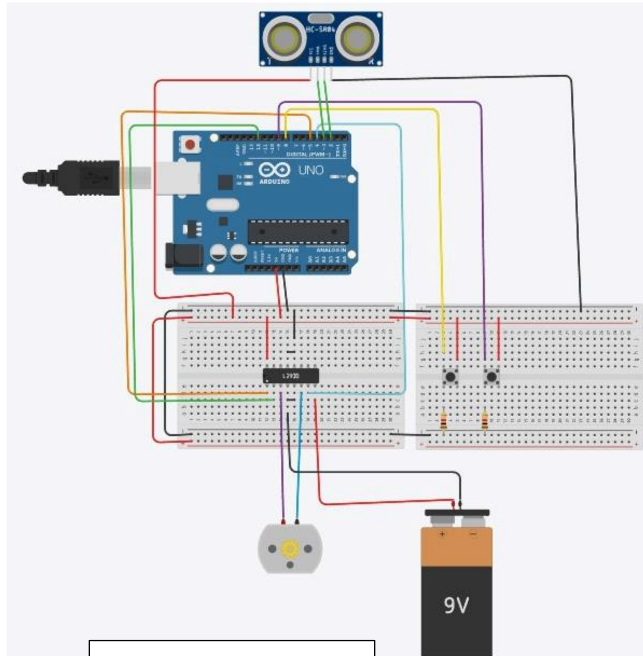

Fig 4.3.1
elevator circuit

```
long ultrasonic() {
  digitalWrite(trig, LOW);
  delayMicroseconds(2);
  digitalWrite(trig, HIGH);
  delayMicroseconds(2);
  digitalWrite(trig, LOW);
  long time = pulseIn(echo, HIGH);
  long distance = time * 0.034 / 2;
  Serial.print("distance=");
  Serial.println(distance);
  delay(20);
  return distance;
}
```

Fig 4.3.2
elevator maquette

Fig 4.3.3
elevator maquette

ARDUINO MEGA: RFID module

```
#include <Servo.h>
#include <LiquidCrystal.h>
#include <SPI.h>
#include <MFRC522.h>
Servo servo;
#define SS_PIN 10
#define RST_PIN 9
MFRC522 mfrc522(SS_PIN, RST_PIN);   // Create MFRC522 instance.
LiquidCrystal lcd(8, 7, 5, 4, 3, 2);
int contrast=0;
void setup()
{
  lcd.begin(16, 2);
  SPI.begin();      // Initiate  SPI bus
  mfrc522.PCD_Init();   // Initiate MFRC522
  pinMode(A0, OUTPUT);
  analogWrite(6,contrast);
  servo.attach(A0);
  Serial.begin(9600);
}
void loop()
{
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Put your Card");
  delay(1000);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Door locked");
  delay(1000);
  // Look for new cards
  if ( ! mfrc522.PICC_IsNewCardPresent())
  {
    return;
  }
  if ( ! mfrc522.PICC_ReadCardSerial())
  {
    return;
  }
  String content= "";
  byte letter;
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
```

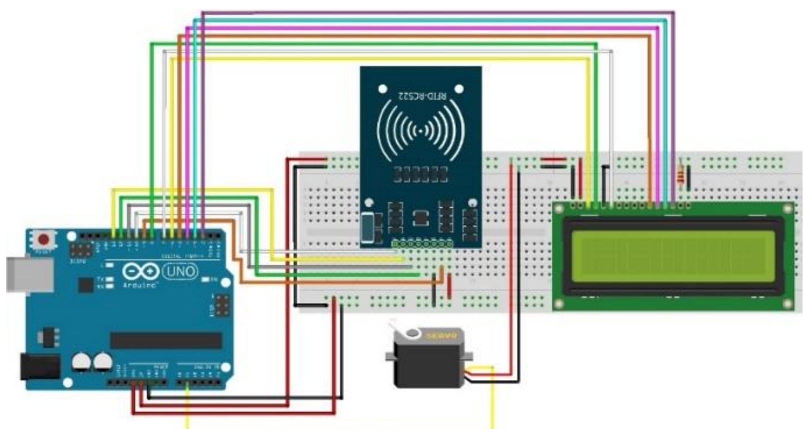

Fig 4.4.1
RFID circuit

```
      Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
      Serial.print(mfrc522.uid.uidByte[i], HEX);
      content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
      content.concat(String(mfrc522.uid.uidByte[i], HEX));
  }
  Serial.println();
  content.toUpperCase();
  if (content.substring(1) == "B3 6E 92 4B" ) //change here the UID of the card
  {
    lcd.clear();
    lcd.setCursor(0, 1);
    lcd.clear();
    lcd.print("Door unlocked");
    servo.write(105);
    delay(600);
    servo.write(90);
    delay(1000);
    servo.write(75);
    delay(600);
    servo.write(90);
    lcd.setCursor(0, 0);
    lcd.clear();
    lcd.print("Door locked");
    delay(1000);
  }
 else if (content.substring(1) == "CC 1B 1B 30")   {
    lcd.clear();
    lcd.setCursor(0, 1);
    lcd.clear();
    lcd.print("Wrong Card");
    delay(3000);
    lcd.clear();
    lcd.print("Door locked");
    delay(1000);
  }
}
```

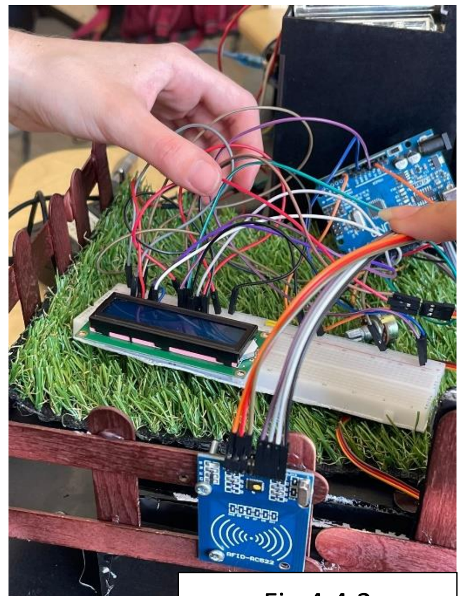

Fig 4.4.2
RFID maquette

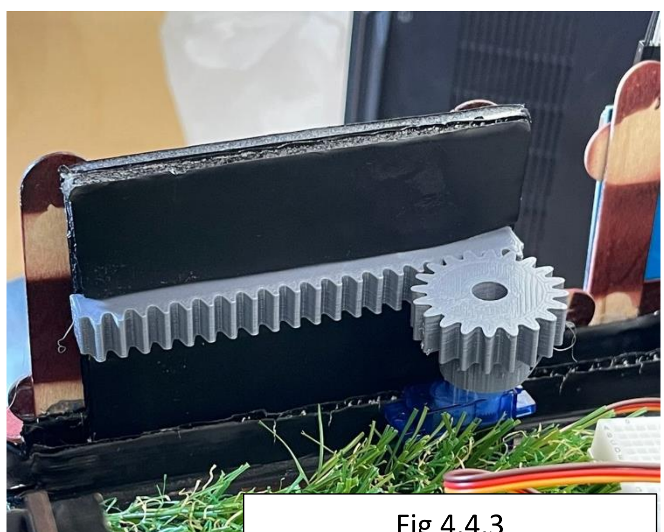

Fig 4.4.3
rack and pinion maquette

CHAPTER 5: Conclusion & Future Plan

Conclusion:

In conclusion, the emergence of smart home technology represents a groundbreaking shift in residential living, ushering in a new era of connectivity and convenience for homeowners. From automated lighting systems that adapt to our daily routines to security systems that provide remote monitoring and control, smart home devices offer unprecedented levels of comfort and efficiency. With innovations such as control of appliances, energy management solutions, and integrated home entertainment systems, the possibilities for enhancing our living spaces are boundless. As we embrace the interconnectedness of our homes, it is clear that smart technology will continue to redefine the way we live, fostering environments that are not only more intelligent but also more responsive to our evolving needs and lifestyles.

Future Plans and Modifications:

Temperature Sensor Integration:

Implementing additional temperature sensors throughout the home to provide more granular data and enable zone-based climate control for enhanced comfort and energy efficiency.

Voice Recognition Commands Expansion:

Expanding the repertoire of voice commands supported by the smart home system to include a wider range of functions and interactions, such as controlling appliances, scheduling tasks, and accessing personalized preferences.

Advanced Wiring Solutions:

Exploring advanced wiring solutions, such as wireless protocols or Power over Ethernet (PoE), to simplify installation, reduce costs, and enhance scalability for future expansion of the smart home network.

Intelligent Appliance Integration:

Integrating smart appliances and IoT devices into the home automation ecosystem to enable seamless communication and coordination between various devices, streamlining daily tasks and improving overall efficiency.

Personalized Automation Profiles:

Developing personalized automation profiles tailored to individual preferences and lifestyles, allowing homeowners to customize their smart home experience based on factors such as occupancy patterns, comfort preferences, and energy-saving goals.

COST ANALYSIS:

Component	Price	Quantity	Total
IR remote+IR sensor	50	1	50
LCD Display	90	1	90
LDR Sensor	5	3	15
Ultrasonic Sensor	45	1	45
Smoke Sensor	80	1	80
Laser	55	1	55
Buzzer	9	2	18
RFID kit	150	1	150
Motor Driver	85	1	85
DC motor	60	1	60
Servo motor	160	1	160
Servo motor micro	120	2	240
Solar panel system	400	1	400
Arduino UNO board	400	2	800
ESP32	480	1	480
Wires male-male	20	1	20
Wires male-female	20	1	20
Long wires all type	150	1	150
Breadboard large	40	1	40
Breadboard small	30	2	60
LED	0.5	10	5
Push button	0.75	2	1.5
Servo Bracket	20	1	20
Material + Laser printing	1320		1320

Total Cost:
4364.5

REFERENCES:

- https://youtu.be/OOXPMdguMn4?si=Ks_tocv7-6VOnyzl
- https://youtu.be/_InacLjy-ZU?si=XXBAlqwwDlDRFA4W
- https://youtu.be/qrQC1STa1Z0?si=h3G0LJxYCtozpL5R
- https://youtu.be/YC4kIGQYld4?si=GdpSSQ8mQ9j8QmWy
- https://youtu.be/MA3hWp2efZ8?si=9JWU2YGs-tr0A7la
- https://youtu.be/alQ9v8dyaTs?si=WKinlk_cCSp1i02j
- https://youtu.be/oZfgQdH0xQo?si=naiyFfUzud2NAqfb
- https://youtu.be/W1xG_XJb0FU?si=i1stJziLYescoaPN
- Shigley’s Mechanical Engineering Design by Richard G. Budynas and Kenneth A. Nisbett