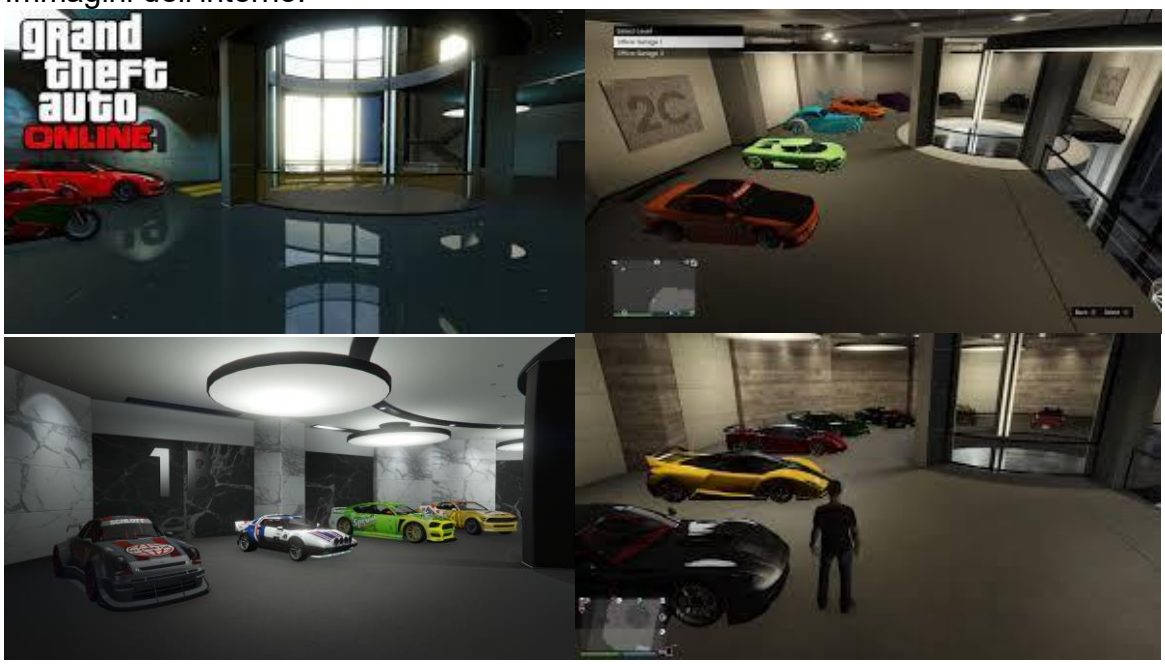
Progetto Finale Robotica

Crippa Leonardo - 5AROB - A.S.: 2023/2024

Il Progetto consiste nella riproduzione di un garage di forma circolare su più piani (2), avente un ascensore al centro per permettere la salita e la discesa dei veicoli.
Il modello da seguire è stato preso da un videogioco, ossia GTA 5 (Grand Theft Auto 5), nello specifico la struttura rappresenta il garage dell'ufficio del CEO.
Immagini dell'interno:



Le parti programmabili del garage sono:

- L'ascensore centrale
- Il sistema di illuminazione
- Il sistema di rilevazione del fumo

Elenco componenti per ogni parte del progetto:

Ascensore
I materiali utilizzati sono:

- Motore passo-passo
- Vite trapezoidale da 350mm (la stessa vite presente nelle stampanti 3D)
- 2x tubi d'acciaio da 400mm (per avere un'altezza maggiore del garage)
- Accoppiatori 5x8 (per intercambiare la larghezza della vite con la larghezza del motore)
- 2x cuscinetti a sfera (per favorire lo scorrimento dell'ascensore lungo i tubi d'acciaio)
- Alimentatore esterno 24V 2A (per l'alimentazione del motore tramite il driver)
- driver A4988 (per il controllo del motore passo-passo tramite arduino)
- Stampe 3D dedicate al piano dell'ascensore e al supporto del motore e dei tubi su cui l'ascensore scorre

Sistema di illuminazione

I materiali utilizzati sono:

- Striscia LED RGB YUNBO 5V DC da 1m

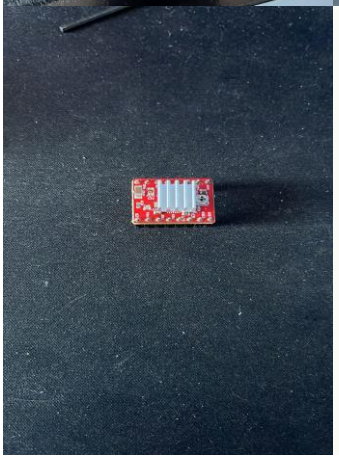
Sistema di rilevazione del fumo
I materiali utilizzati sono:

- 6x Sensori di gas/fumo MQ-2

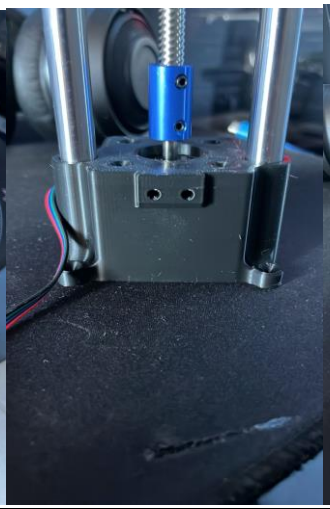
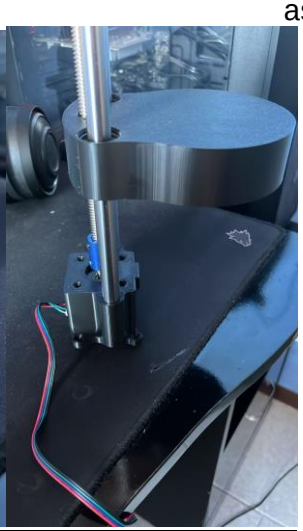
Immagini componenti utilizzati:

- Ascensore



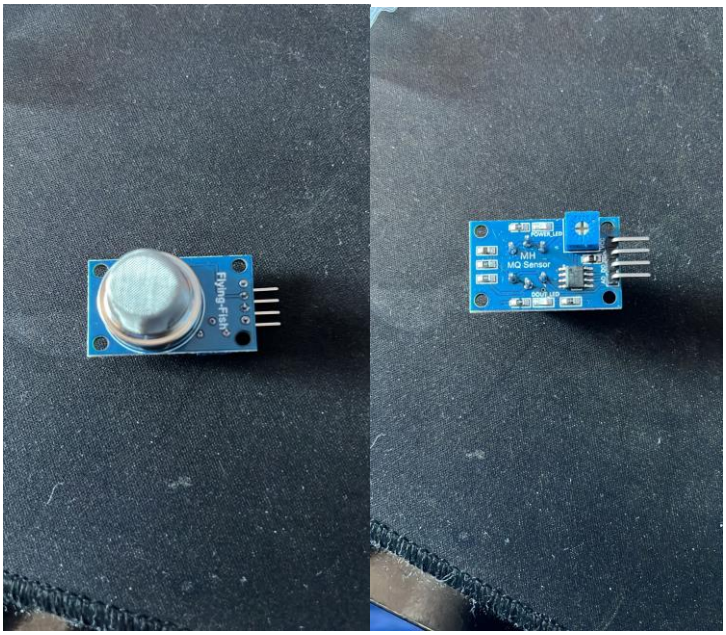


Ascensore assemblato:

- Sistema di Illuminazione

- Sistema di rilevamento del fumo



Codici di programmazione dei pezzi:

- Ascensore

```
#include <AccelStepper.h>

// Definizione dei pin per il driver A4988
#define DIR_PIN 3
#define STEP_PIN 2

// Crea un oggetto AccelStepper
AccelStepper stepper(AccelStepper::DRIVER, STEP_PIN, DIR_PIN);

void setup() {
  // Imposta la velocità massima e l'accelerazione del motore
  stepper.setMaxSpeed(1000);
  stepper.setAcceleration(500);

  // Imposta la direzione del motore (cambiare se necessario)
  stepper.setDirection(1); // Orario
```

```
  // stepper.setDirection(-1); // Antiorario
}

void loop() {
  // Fa girare il motore per un certo numero di passi
  stepper.move(200); // Numero di passi da effettuare

  // Controlla se il motore ha completato il movimento
  while (stepper.distanceToGo() != 0) {
    stepper.run();
  }

  delay(1000); // Attendi un secondo prima di ripetere
}
```

- Sistema di illuminazione

```
#include <Adafruit_NeoPixel.h>

#define PIN 6 // Pin digitale a cui è collegata la striscia LED
#define NUMPIXELS 144 // Numero di LED nella striscia
Adafruit_NeoPixel strip = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

void setup() {
strip.begin(); // Inizializza la striscia LED
strip.show(); // Assicura che tutti i LED siano spenti all'inizio
}

void loop() {
// Esempio di animazione: scorrimento di un singolo LED rosso lungo la striscia
for (int i = 0; i < NUMPIXELS; i++) {
strip.setPixelColor(i, strip.Color(255, 0, 0)); // Imposta il colore del LED su rosso
strip.show(); // Mostra il cambiamento
delay(50); // Aspetta 50 millisecondi
strip.setPixelColor(i, strip.Color(0, 0, 0)); // Spegne il LED }
}
```

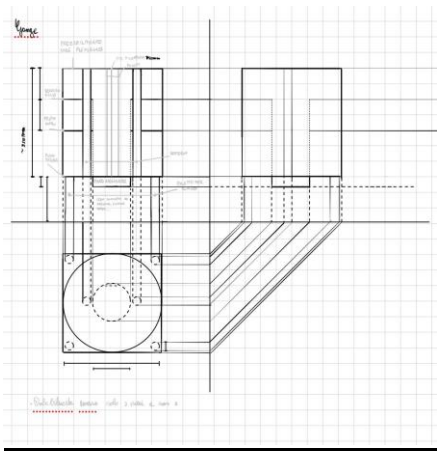
- Sistema di rilevamento del fumo

```
// Dichiarazione dei pin per i sensori di fumo
#define P_T A0 // Pin analogico per la serie di sensori di fumo del piano terra
#define P_1 A1 // Pin analogico per la serie di sensori di fumo del primo piano

void setup() {
  Serial.begin(9600); // Inizializza la comunicazione seriale
}

void loop() {
  // Leggi il valore analogico del sensore di fumo del piano terra
```

```
  int LivelloFumo_PT = analogRead(P_T);

  // Leggi il valore analogico del sensore di fumo del primo piano
  int LivelloFumo_P1 = analogRead(P_1);

  // Stampa i valori letti sulla porta seriale
  Serial.print("Livello Fumo - Piano Terra: ");
  Serial.println(LivelloFumol_PT);
  Serial.print("Livello Fumo - Primo Piano: ");
  Serial.println(LivelloFumo_P1);

  // Fai qualcosa in base ai livelli di fumo rilevati
  if (LivelloFumo_PT > 500) {
    // Il valore da superare del livello di fumo sarà scelto in seguito
    Serial.println("Fumo rilevato al piano terra!");
    // Aggiungere il codice per la gestione dell'allarme per il piano terra
  }

  if (LivelloFumo_P1 > 500) {
    // Il valore da superare del livello di fumo sarà scelto in seguito
    Serial.println("Fumo rilevato al primo piano!");
    // Aggiungere il codice per la gestione dell'allarme per il primo piano
  }

  delay(1000); // Attendi un secondo prima di leggere nuovamente i sensori
}
```

Struttura

Per quanto riguarda la struttura ho inizialmente realizzato una proiezione ortogonale del progetto in modo da avere un'idea sulle dimensioni della struttura, in seguito mi sono organizzato con Singh Jaskaran per progettare un file 3D in modo da poter stampare i pezzi al fine di avere una facilità maggiore rispetto ad una realizzazione tramite legno che risulta difficile da tagliare in modo preciso



La struttura è realizzata seguendo una logica:

- è presente un ripiano inferiore dedicato alla locazione della base dell'ascensore, di Arduino e dei cavi; come se fosse una zona logistica
- a separare la parte logistica c'è il pavimento con dei buchi dedicati al passaggio delle barre filettate necessarie per la stabilità della struttura
- il soffitto del piano terra (corrispondente con il pavimento del primo piano), Nella parte inferiore dei fori per poter inserire i sensori di fumo;
  Nella parte superiore invece, sono presenti delle canaline apribili per poter far passare tutti i cavi dei sensori e della barra LED.
  Questo piano ha una parte con una ringhiera per rispettare la struttura presente nel gioco e per un fattore estetico.
- il soffitto del primo piano ha gli stessi fori nella parte inferiore e le stesse canaline nella parte superiore, con la differenza che manca la zona vuota con la ringhiera, ma è un piano circolare totalmente chiuso

Componenti necessari per la struttura:

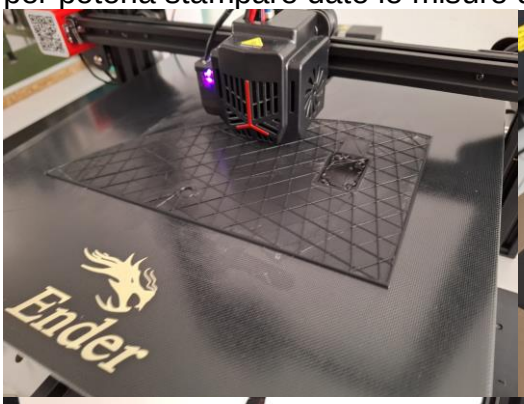
- per avere stabilità nella struttura, non avendo pareti, ho inserito 4 viti filettate m8 da 50cm l'una circa, in modo da avere dei "paletti" su cui inserire le parti stampate e rendere gli spazi dell'ascensore equidistanti tra di loro.
- per fissare la posizione dei piani ho utilizzato 32 dadi m8 da posizionare rispettivamente su ogni vite filettata, sia sopra sia sotto al piano.

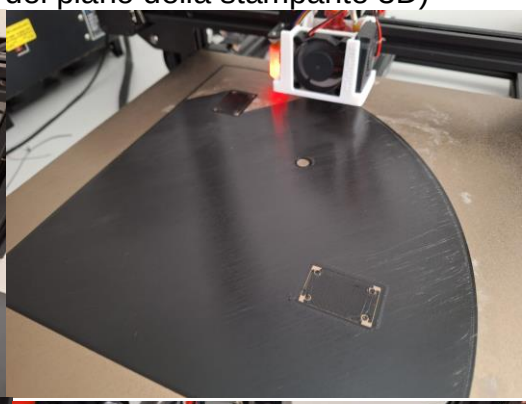
Immagini relative alle stampe della struttura:

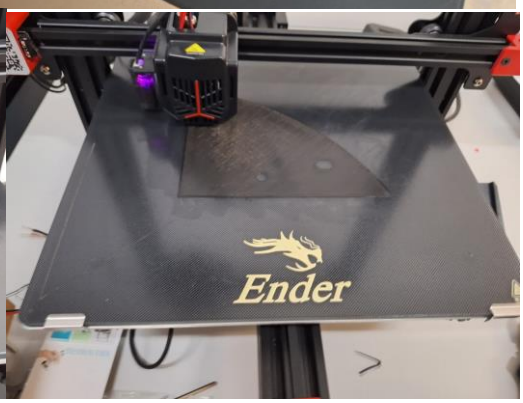
- Render 3D del progetto

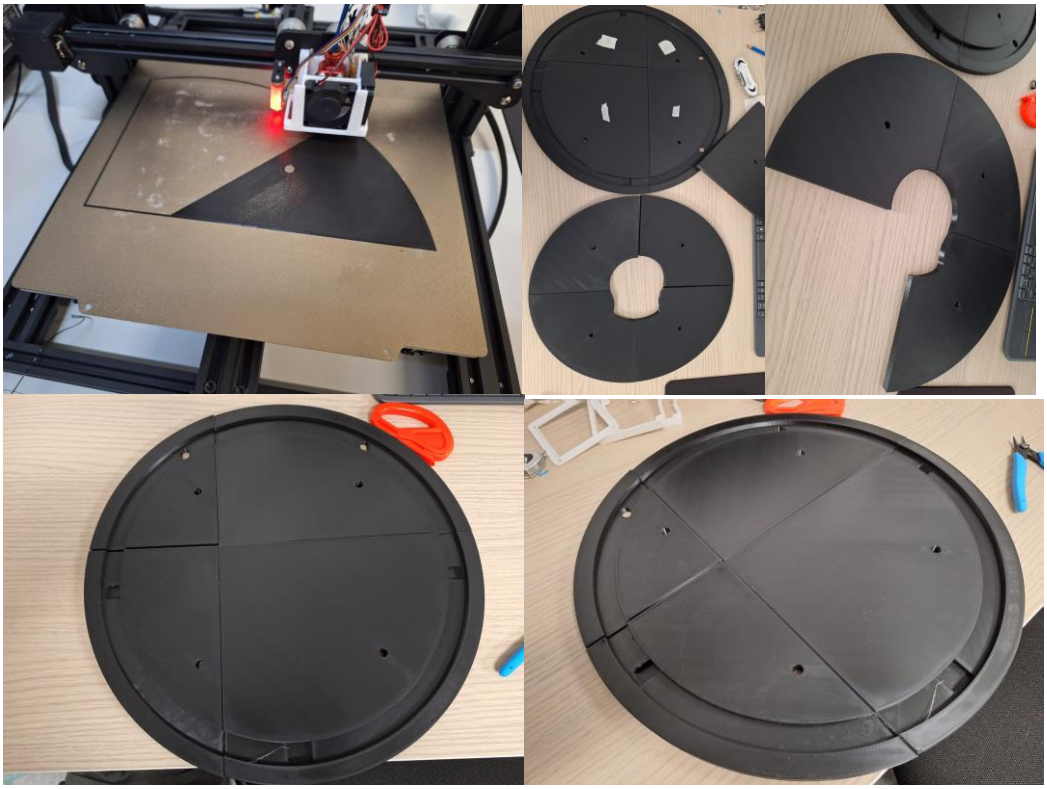
INSERIRE IMMAGINE DEL RENDER

- Immagini delle stampe in corso (essendo molto larga, è stata divisa in 4 parti per poterla stampare date le misure del piano della stampante 3D)







Assemblaggio della struttura
per assemblare la struttura ho utilizzato la colla a caldo per unire le parti come primo passo, in seguito, dopo aver incollato le 4 parti della base della parte logistica ho inserito le 4 viti filettate con 4 coppie di dadi m8 sia sopra alla base sia sotto e infine, ho fissato il motore e l'ascensore tenendo come riferimento il buco del pian terreno, per il passaggio dell'ascensore.

In seguito, ho inserito lungo le viti filettate il pavimento e i dadi m8 per fissare il piano

Nuovamente, ho inserito il soffitto del piano terra e infine anche il soffitto del primo piano

Integrazione con il sito internet

Avendo realizzato un sito internet personale su altervista su cui sono presenti i progetti svolti, ho deciso di inserire nella parte relativa a questo progetto, dopo la spiegazione, ho inserito una sezione relativa al controllo dell'ascensore e del sistema antincendio tramite dei pulsanti (bloccati tramite password), infatti, tramite Arduino UNO R4 WiFi ho realizzato un codice che permette la connessione ad internet e prende valori inviati dal sito alla pressione dei pulsanti. Questa zona, per evitare il controllo da parte di chiunque abbia accesso al sito, è una zona protetta da una password

Codice di integrazione Arduino tramite sito internet

- Arduino

```
#include <WiFiS3.h>

const char* ssid = "*****";
const char* password = "*****";

WiFiServer server(80);

bool stato = LOW;

void setup() {
Serial.begin(115200);
delay(10);

pinMode(12, OUTPUT);
pinMode(13, OUTPUT);
pinMode(3, OUTPUT);

digitalWrite(12, LOW);
digitalWrite(13, LOW);
digitalWrite(3, LOW);

// Connettiti alla rete WiFi
Serial.println();
Serial.println();
Serial.print("Connettendo a ");
Serial.println(ssid);

WiFi.begin(ssid, password);

while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
}
Serial.println("");
Serial.println("Connesso alla rete WiFi");
Serial.println("Indirizzo IP: ");
Serial.println(WiFi.localIP());

server.begin();
}

void loop() {
WiFiClient client = server.available();
if (!client) {
  return;
}

Serial.println("Nuova connessione");
```

```
while (!client.available()) {
  delay(1);
}

String request = client.readStringUntil('\r');
Serial.println("Richiesta: ");
Serial.println(request);
client.flush();

int command = -1;

if (request.indexOf("/0") != -1) {
  command = 0;
} else if (request.indexOf("/1") != -1) {
  command = 1;
} else if (request.indexOf("/2") != -1) {
  command = 2;
}

switch (command) {
  case 0:
    digitalWrite(12, LOW);
     for (int i = 0; i < 9590; i++) {
      digitalWrite(13, stato);
      delay(1);
      stato = !stato;
    }
    break;
  case 1:
    digitalWrite(12, HIGH);
    for (int i = 0; i < 9590; i++) {
      digitalWrite(13, stato);
      delay(1);
      stato = !stato;
    }
    break;
  case 2:
    tone(3, 1000); // Suona il buzzer collegato al pin digitale 3 a 1000Hz
    delay(1000);   // Suona per 1 secondo
    noTone(3);     // Interrompi il suono
    break;
  default:
    // Comando non riconosciuto
    break;
}

client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
```

```
client.println();
client.println("<!DOCTYPE HTML>");
client.println("<html>");
client.println("<head><title>Controllo Arduino Uno R4 WiFi</title></head>");
client.println("<body>");
client.println("<h1>Comando ricevuto e eseguito con successo!</h1>");
client.println("</body>");
client.println("</html>");

  delay(1);
  Serial.println("Client disconnesso");
}
```

- Sito Internet

Parte html:

```html
<!--Controllo-->
<section id="contact" class="contact">
    <div class="main-text">
        <span>Controllo</span>
        <h2>Tramite WiFi:</h2>
    </div>

    <!-- Inserimento della password -->
    <div class="password-section">
        <input type="password" id="password" placeholder="Inserisci la password">
        <button onclick="checkPassword()" class="button-1">Accedi</button>
    </div>

    <!-- Pulsanti per i comandi -->
    <div class="control-section hidden">
        <div class="btn-box-project">
            <button class="button-1" onclick="sendCommand('0')">Piano Terra</button>
            <button class="button-1" onclick="sendCommand('1')">Primo Piano</button>
            <button class="button-1" onclick="sendCommand('2')">Simula Impianto Antincendio</button>
        </div>
        <div class="status"></div>
    </div>
</section>
```

Parte Javascript:

```
    <script> // Funzione per verificare la password
        // Funzione per verificare la password
function checkPassword() {
    var passwordInput = document.getElementById("password").value;
    // Verifica della password
    if (passwordInput === "leo12344") {
        // Mostra i pulsanti per i comandi
        document.querySelector('.control-section').classList.remove('hidden');
        document.querySelector('.status').textContent = "";
    } else {
        // Se la password è errata, mostra un messaggio di errore e ripulisci la casella di input
        document.querySelector('.status').textContent = "Password errata, riprova.";
        document.getElementById("password").value = ""; // Pulisce la casella di input
    }
}

// Funzione per inviare i comandi all'Arduino
function sendCommand(command) {
    // Effettua la richiesta al server dell'Arduino Uno R4 WiFi
    // Sostituisci 'indirizzo_arduino' con l'indirizzo IP o il nome del dominio del tuo Arduino Uno R4 WiFi
    // E 'comando' con il comando appropriato da inviare
    fetch(`http://192.168.1.12/${command}`)
        .then(response => {
            if (!response.ok) {
                throw new Error('Errore di rete.');
            }
            return response.text();
        })
        .then(data => {
            console.log('Risposta dal server Arduino:', data);
            // Se necessario, gestisci la risposta del server qui
        })
        .catch(error => {
            console.error('Errore:', error);
        });
}
        </script>
</body>
```