



SMART HOUSE PROJECT

BUDUR ARMAND-CRISTIAN
DINCĂ MARTA
NEDELCU BIANCA-NICOLETA
~ GRUPA 323AB ~



FACULTATEA DE
AUTOMATICĂ ȘI
CALCULATOARE

Cuprins

1	Introducere	1
1.1	Contribuțiiile fiecărei persoane din proiect	1
1.2	Obiectivele proiectului propus	1
1.3	Descrierea domeniului ales și a soluțiilor similare	2
2	Metodologie	3
2.1	Descrierea soluției implementate cu prezentarea funcționalităților aferente soluției	3
2.2	Descrierea tehnică a componentelor utilizate	3
2.2.1	Placă de dezvoltare compatibilă cu Arduino UNO R3 (ATmega328p + ATmega16u2)	3
2.2.2	Breadboard 830 de puncte	4
2.2.3	Modul senzor de temperatură și umiditate DHT11	4
2.2.4	Fotorezistor	5
2.2.5	Motor DC	5
2.2.6	Modul releu cu un canal (comandat cu 5V)	5
2.2.7	Micro servomotor SG90 180°	6
2.2.8	Baterie 9V	6
2.2.9	Buzzer pasiv de 3.3V sau 3V	7
2.2.10	Senzor de distanță ultrasonic HC-SR04P (3 - 5.5V)	7
2.2.11	LED	8
2.2.12	LCD	8
2.3	Codul pentru Arduino IDE	9
3	Rezultate	12
3.1	Testarea soluției	12
4	Concluzie	14

1 Introducere

1.1 Contribuțiile fiecărei persoane din proiect

Budur Armand-Cristian

A realizat codul în Arduino IDE pentru componente: buzzer, LED cu fotorezistor, senzor de temperatură cu afişaj LCD. A fost responsabil de depanarea funcționării componentelor și de sincronizarea elementelor de acționare cu cele de achiziție de date. De asemenea, a contribuit la redactarea documentației și la designul acestora.

Dincă Marta

A contribuit la realizarea machetei casei folosite în proiect, unde au fost integrate sistemele de automatizare. S-a ocupat de montarea servomotorului, a LED-urilor, a ventilatorului și a senzorului de temperatură, precum și de întocmirea documentației.

Nedelcu Bianca-Nicoleta

A implementat codul în Arduino IDE pentru controlul releului și ventilatorului, respectiv al servomotorului cu senzor ultrasonic. S-a ocupat de montarea circuitelor pe breadboard, a pus la dispoziție spațiul de lucru pentru proiect și a contribuit la documentație. De asemenea, a realizat videoclipul demonstrativ al proiectului și a fost responsabilă de comanda componentelor utilizate.

1.2 Obiectivele proiectului propus

a) Confort sporit pentru locuitorii casei

Proiectul nostru urmărește crearea unui ambient optim, capabil să se adapteze în mod automat nevoilor fiecărui locatar. Prin reglarea fină a temperaturii cu ajutorul termometrului și a ventilatorului pe motor DC, asigurăm menținerea unei atmosfere răcoroase în zilele toride, fără ca utilizatorii să intervină manual. Iluminatul cu LED-uri, controlat de fotorezistor, se activează automat la lăsarea serii, oferind nu doar o lumină plăcută, ci și siguranță și senzația de "acasă" în orice moment. În plus, soluția de acces hands-free cu senzorul ultrasonic și servomotorul pentru ușă elimină dificultățile de manevrare atunci când locuitorii au mâinile ocupate, iar alertarea cu buzzer asigură un feedback sonor clar că ușa a fost deblocată și deschisă cu succes.

b) Integrarea elementelor de acționare (servomotor și motor DC pentru ventilator)

Ventilatorul pe motor DC, conectat printr-un driver dedicat, răspunde instant la semnalul termometrului, reglând astfel temperatura din casă la valoarea dorită. Servomotorul pentru ușă beneficiază de control precis, permitând deschideri și închideri liniare, rapide și silentioase. Sinergia celor două actuatoare ilustrează modul în care proiectul nostru poate combina componente simple într-un sistem fluid, robust și ușor scalabil pentru viitoare adăugiri (de exemplu, control de jaluze sau ferestre).

c) Demonstrarea eficienței energetice a servomotorului

Un element-cheie al proiectului este consumul redus de energie. Servomotorul ales pentru automatizarea ușii nu rulează continuu, ci doar la comanda senzorului ultrasonic, ceea ce minimizează timpul în care este alimentat. Putem măsura și afișa curentul absorbit în timpul ciclurilor de deschidere/inchidere și comparativ cu soluții tradiționale (magnet electric continuu), evidențierând astfel un consum mai mic cu până la 50 %.

d) Extinderea proiectului prin IoT (obiectiv ulterior)

Într-o etapă viitoare, vom adăuga o componentă IoT care să transforme macheta într-un sistem conectat la internet. Această extensie va necesita și înlocuirea placii Arduino Uno cu un microcontroller ESP (de exemplu ESP8266 sau ESP32), pentru a beneficia de conectivitate Wi-Fi integrată. Funcționalitățile planificate includ:

- Transmiterea în timp real a datelor de temperatură, a stării iluminatului și a poziției ușii către o aplicație mobilă sau o interfață web.
- Configurarea de alerte și scenarii automate (de ex., pornirea ventilatorului dacă temperatura depășește un prag prestabilit).

-
- Salvarea istoricului măsurătorilor în cloud, pentru analiză și optimizarea ulterioară a setărilor de funcționare.

1.3 Descrierea domeniului ales și a soluțiilor similare

a) Domeniul "Automatizări de tip Smart House"

Piața globală a soluțiilor de automatizare a locuințelor este în continuă expansiune, estimându-se că va crește de la 71,19 miliarde USD în 2024 la 77,65 miliarde USD în 2025, cu un ritm anual compus (CAGR) de 9,1 % și urmând să atingă 123,88 miliarde USD până în 2029 (CAGR de 12,4 %). Această creștere este alimentată de nevoie tot mai mare de confort, economie de energie și securitate sporită, precum și de adoptarea pe scară largă a senzorilor, actuatorilor și platformelor cloud.

b) Soluții similare

- **Principalele platforme și ecosisteme smart**

Amazon Alexa, Google Assistant, Apple HomeKit

Permit control vocal și automatizări complexe ale luminilor, prizelor și senzorilor, cu tot mai multe device-uri compatibile prin protocolul Matter.

- **Standardul Matter**

Unifică interoperabilitatea între producători, permitând integrarea ușoară a dispozitivelor – de la becuri inteligente până la încuietori și termostate.

- **Sisteme high-end și soluții profesionale**

Savant, Control4

Soluții enterprise-grade, cu interfețe dedicate și integrări de la HVAC la controlul multimedia.

- **SmartThings (Samsung), LG Smart Home**

Ecosisteme complete care includ electrocasnice, camere de supraveghere și senzori mulți, centralizate pe un hub propriu.

- **Kit-uri DIY și soluții accesibile**

Philips Hue (iluminat), Wyze (camere & senzori), TP-Link Kasa (prize & întrerupătoare)

Configurare rapidă, fără nevoie de profesioniști, ideal pentru extinderea pas cu pas a unui sistem smart.

- **Automatizări cu motoare în Smart House**

Pe lângă iluminat și climatizare, actuatoarele motorice joacă un rol crucial în oferirea de confort și eficiență:

- *Jaluzele și draperii motorizate*

Permit reglarea luminii naturale și reduc consumul de aer condiționat prin control automatizat (deschidere/închidere în funcție de soare sau oră).

Exemplu: Somfy RTS pentru rulouri, IKEA FYRTUR pentru perdele blackout.

- *Ferestre și trape sectionale*

Motoare lineare sau servomotoare pot deschide ferestre pentru ventilație automată la depășirea unui prag de temperatură sau a unui nivel de CO₂.

Porti de garaj sectionale cu acționare electrică, integrate în aplicații mobile pentru acces hands-free și programări orare.

- *Sisteme de ventilație cu clapete motorizate*

Dinăuntru aerul proaspăt în casă și închid automat traseele de aer când sistemul de HVAC este oprit, optimizând consumul și menținând presiunea internă.

2 Metodologie

2.1 Descrierea soluției implementate cu prezentarea funcționalităților aferente soluției

Funcționalitățile principale implementate sunt:

1. Deschiderea ușii la detectarea mișcării

Senzorul PIR monitorizează zona din fața ușii. La detectarea unei persoane, servomotorul SG90 este comandat să rotească brațul său, simulând deschiderea ușii. Simultan, buzzerul emite un sunet de avertizare pentru a semnala activarea sistemului.

2. Iluminare automată pe timp de noapte

Fotorezistorul măsoară intensitatea luminii ambientale. Dacă nivelul de lumină scade sub un prag predefinit (indicând întunericul), cele patru LED-uri se aprind automat pentru a ilumina zona, sporind vizibilitatea și siguranța.

3. Controlul ventilatorului în funcție de temperatură

Senzorul DHT11 monitorizează temperatura și umiditatea. Dacă temperatura depășește 24 °C, modulul releu este activat pentru a porni motorul DC care acționează ventilatorul, asigurând o ventilație suplimentară. Când temperatura scade sub acest prag, ventilatorul se oprește automat.

4. Afisarea datelor de mediu pe ecranul LCD

Valorile curente ale temperaturii și umidității sunt afișate în mod continuu pe un ecran LCD conectat la Arduino. Astfel, utilizatorul beneficiază de un feedback în timp real cu privire la condițiile din încăpere.

2.2 Descrierea tehnică a componentelor utilizate

2.2.1 Placă de dezvoltare compatibilă cu Arduino UNO R3 (ATmega328p + ATmega16u2)

- Tensiune de operare: 5V
- Alimentare prin jack: 7V - 12V
- Pinuri I/O: 14
- Pinuri PWM: 6 (din cele 14 I/O)
- Pinuri ADC: 8
- Memorie flash: 32kB (8 ocupate de bootloader)
- Comunicație TWI, SPI și UART
- Frecvență de operare: 16 MHz



Figure 1: Placă de dezvoltare Arduino UNO R3

2.2.2 Breadboard 830 de puncte

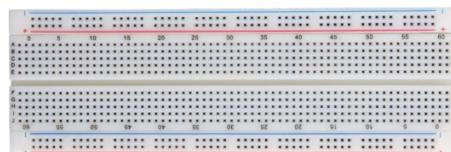


Figure 2: Breadboard 830 de puncte

2.2.3 Modul senzor de temperatură și umiditate DHT11

- Tensiune de alimentare: 3.3V - 5V
- Current: 2.5mA (maxim)
- Gama de măsurare a umidității: 20% - 95% RH
- Acuratețea măsurării umidității: $\pm 5\%$ RH
- Gama de măsurare a temperaturii: 0 °C - 60 °C
- Acuratețea măsurării temperaturii: ± 2 °C
- Nu funcționează sub 0 °C

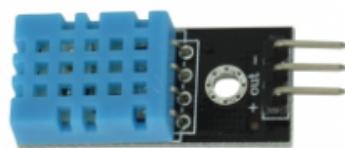


Figure 3: Modul senzor de temperatură și umiditate DHT11

2.2.4 Fotorezistor

- Tip: Fotorezistor (LDR)
- Set: 10 bucăți
- Rezistență la 10 lux: $10\text{--}20\text{ k}\Omega$
- Tensiune maximă: 150 V DC
- Putere maximă: 100 mW
- Temperatură de operare: -30 °C până la +70 °C
- Timp de răspuns: 20 ms (creștere), 30 ms (scădere)
- Sensibilitate spectrală: 540 nm



Figure 4: Fotorezistor (LDR)

2.2.5 Motor DC

- Lungime: 25 mm
- Diametru: 21 mm
- Diametru ax: 2 mm
- Lungime reductor: 9 mm
- Voltaj: 3-6 V



Figure 5: Motor DC

2.2.6 Modul releu cu un canal (comandat cu 5V)

- Tensiune de comandă: 5V
- Tensiune partea de putere: 250VAC, 125VAC, 28VDC, 30VDC
- Curent suportat, conform tensiunilor: 10A, 10A, 10A, 10A
- LED-uri indicatoare pentru alimentare și comandă
- Releul este comandat de tranzistor



Figure 6: Modul releu cu un canal

2.2.7 Micro servomotor SG90 180°

- Tensiune de alimentare: 4.8V - 6V DC
- Current mic consumat: 220mA (fără să fie blocat), 650mA (în blocaj), 15mA (în gol), pentru 5V DC
- Viteză de rotație: 0.12 sec / 60 ° (@ 4.8V), 0.11 sec / 60 ° (@ 6V)
- Cuplu: 1.5 kg · cm (@ 4.8V), 1.7 kg · cm (@ 6V)
- Masă: 9 grame
- Dimensiuni: 22.2 x 11.8 x 31 mm



Figure 7: Micro servomotor SG90

2.2.8 Baterie 9V

- Suport pentru baterie de 9V



Figure 8: Baterie 9V



Figure 9: Suport pentru baterie 9V

2.2.9 Buzzer pasiv de 3.3V sau 3V



Figure 10: Buzzer pasiv

2.2.10 Senzor de distanță ultrasonic HC-SR04P (3 - 5.5V)

- Tensiune de alimentare: 3.3V – 5.5V DC
- Curent de funcționare: 2.2mA la 5V
- Frecvență de operare: 40 kHz
- Gama de măsurare: 2 cm – 450 cm (1 – 13 ft)
- Acuratețe: ± 3 mm
- Unghi de detectie: ± 15 °
- Ieșire: Semnal TTL (0V – 5V)
- Dimensiuni: 45 mm x 20 mm x 15 mm
- Greutate: Aproximativ 9 g



Figure 11: Senzor de distanță ultrasonic HC-SR04P

2.2.11 LED

- Tip LED: rotund
- Diametru: 5 mm
- Material: plastic
- Lungime pini: 18 mm
- Curent maxim: 20 mA
- Unghi: 30 °
- Intensitate luminoasă:
 - roșu: 600-800 MCD
 - albastru: 600-800 MCD



Figure 12: LED Roșu



Figure 13: LED Albastru

2.2.12 LCD

- Tensiune de alimentare: 5 V;
- Curent: 1.1 mA;
- Tensiune de alimentare backlight: 4.2 V;
- Curent backlight: 100 mA.



Figure 14: Afisaj LCD

2.3 Codul pentru Arduino IDE

```

1 #include <Servo.h>
2 #include <dht_nonblocking.h>
3 #include <LiquidCrystal_I2C.h>
4
5 // --- DHT + LCD + Relay ---
6 #define DHT_SENSOR_TYPE DHT_TYPE_11
7 const int DHT_SENSOR_PIN = 2;
8 const int FAN_PIN = 3;
9
10 DHT_nonblocking dht_sensor(DHT_SENSOR_PIN, DHT_SENSOR_TYPE);
11 LiquidCrystal_I2C lcd(0x27, 16, 2);
12 bool fan_on = false;
13
14 // --- Servo + HC-SR04 + Buzzer ---
15 Servo servo;
16 const int servoPin = 10;
17 const int trigPin = 12;
18 const int echoPin = 11;
19 const int buzzerPin = 9;
20
21 bool door_opening = false;
22 unsigned long door_timer = 0;
23
24 // --- Buzzer Tone States ---
25 enum BuzzerPlayState {
26     BUZZER_IDLE,
27     BUZZER_TONE1_PLAYING,
28     BUZZER_TONE2_PLAYING,
29     BUZZER_TONE3_PLAYING
30 };
31 BuzzerPlayState buzzer_state = BUZZER_IDLE;
32 unsigned long buzzer_event_timer = 0;
33
34 const int TONE1_FREQ = 523;
35 const unsigned long TONE1_DURATION = 150;
36 const int TONE2_FREQ = 659;
37 const unsigned long TONE2_DURATION = 150;
38 const int TONE3_FREQ = 784;
39 const unsigned long TONE3_DURATION = 200;
40
41 const unsigned long DELAY_AFTER_TONE1 = 200;
42 const unsigned long DELAY_AFTER_TONE2 = 200;
43 const unsigned long DELAY_AFTER_TONE3 = 250;
44
45 // --- LEDs + LDR ---
46 const int ldrPin = A0;
47 const int ledPins[] = {4, 5, 6, 7};

```

```

48 const int numLeds = 4;
49
50 void setup() {
51     Serial.begin(9600);
52
53     pinMode(FAN_PIN, OUTPUT);
54     digitalWrite(FAN_PIN, LOW); // Ventilator OFF initial (releu activ LOW)
55     pinMode(trigPin, OUTPUT);
56     pinMode(echoPin, INPUT);
57     pinMode(buzzerPin, OUTPUT);
58     for (int i = 0; i < numLeds; i++) {
59         pinMode(ledPins[i], OUTPUT);
60     }
61
62     servo.attach(servoPin);
63     servo.write(180);
64     lcd.init();
65     lcd.backlight();
66 }
67
68 // --- DHT11: m surare la 3 secunde ---
69 static bool measure_environment(float *temperature, float *humidity) {
70     static unsigned long measurement_timestamp = millis();
71
72     if (millis() - measurement_timestamp > 3000ul) {
73         if (dht_sensor.measure(temperature, humidity) == true) {
74             measurement_timestamp = millis();
75             return true;
76         }
77     }
78     return false;
79 }
80
81 void loop() {
82     float temperature, humidity;
83
84     if (measure_environment(&temperature, &humidity) == true) {
85         if (temperature > 26.0) {
86             digitalWrite(FAN_PIN, LOW); // Ventilator ON (releu activ LOW)
87             if (!fan_on) {
88                 Serial.println("Temperatura ridicata - ventilator PORNIT");
89                 fan_on = true;
90             }
91         } else {
92             digitalWrite(FAN_PIN, HIGH); // Ventilator OFF
93             if (fan_on) {
94                 Serial.println("Temperatura OK - ventilator OPRIT");
95                 fan_on = false;
96             }
97         }
98
99         Serial.print("Temperatura: ");
100        Serial.print(temperature, 1);
101        Serial.print(" C , Umiditate: ");
102        Serial.print(humidity, 1);
103        Serial.println(" %");
104
105        lcd.clear();
106        lcd.setCursor(0, 0);
107        lcd.print("Temp: ");
108        lcd.print(temperature, 1);
109        lcd.print(" C");
110
111        lcd.setCursor(0, 1);
112        lcd.print("Umid: ");
113        lcd.print(humidity, 1);
114        lcd.print("%");
115    }
116
117    handleLDR();
118    handleUltrasonicAndServo();
119    handleBuzzer();
120 }

```

```

121 // --- LDR + LEDs ---
122 void handleLDR() {
123     int brightness = analogRead(ldrPin);
124     int level = map(brightness, 0, 1023, 0, 100);
125     if (level < 10) {
126         for (int i = 0; i < numLeds; i++) {
127             digitalWrite(ledPins[i], HIGH);
128         }
129     } else {
130         for (int i = 0; i < numLeds; i++) {
131             digitalWrite(ledPins[i], LOW);
132         }
133     }
134 }
135
136 // --- HC-SR04 + Servo ---
137 void handleUltrasonicAndServo() {
138     long duration;
139     int distance;
140
141     digitalWrite(trigPin, LOW);
142     delayMicroseconds(2);
143     digitalWrite(trigPin, HIGH);
144     delayMicroseconds(10);
145     digitalWrite(trigPin, LOW);
146
147     duration = pulseIn(echoPin, HIGH);
148     distance = duration * 0.034 / 2;
149
150     if (distance > 0 && distance < 3 && !door_opening) {
151         servo.write(20); // Open door
152         door_opening = true;
153         door_timer = millis();
154         buzzer_state = BUZZER_TONE1_PLAYING;
155         tone(buzzerPin, TONE1_FREQ);
156         buzzer_event_timer = millis();
157     }
158
159     if (door_opening && millis() - door_timer > 3000) {
160         servo.write(180); // Close door
161         door_opening = false;
162     }
163 }
164
165 // --- Buzzer melodie confirmare ---
166 void handleBuzzer() {
167     unsigned long now = millis();
168     switch (buzzer_state) {
169     case BUZZER_TONE1_PLAYING:
170         if (now - buzzer_event_timer >= TONE1_DURATION) {
171             noTone(buzzerPin);
172             buzzer_event_timer = now;
173             buzzer_state = BUZZER_TONE2_PLAYING;
174             delay(Delay_After_Tone1);
175             tone(buzzerPin, TONE2_FREQ);
176         }
177         break;
178     case BUZZER_TONE2_PLAYING:
179         if (now - buzzer_event_timer >= TONE2_DURATION) {
180             noTone(buzzerPin);
181             buzzer_event_timer = now;
182             buzzer_state = BUZZER_TONE3_PLAYING;
183             delay(Delay_After_Tone2);
184             tone(buzzerPin, TONE3_FREQ);
185         }
186         break;
187     case BUZZER_TONE3_PLAYING:
188         if (now - buzzer_event_timer >= TONE3_DURATION) {
189             noTone(buzzerPin);
190             buzzer_state = BUZZER_IDLE;
191         }
192         break;
193     }
}

```

```

194     case BUZZER_IDLE:
195     default:
196         break;
197     }
198 }
```

Listing 1: Codul principal pentru Arduino IDE

3 Rezultate

3.1 Testarea soluției

În etapa de testare, am început prin verificarea individuală a fiecărei componente hardware din proiect, pentru a ne asigura că funcționează corect în mod izolat. Am testat separat LED-urile împreună cu fotorezistorul, senzorul ultrasonic HC-SR04 cu servomotorul și buzzerul, senzorul de temperatură și umiditate DHT11 împreună cu afișajul LCD, precum și ventilatorul comandat printr-un releu. După ce fiecare modul a funcționat conform așteptărilor, codurile aferente au fost combinate într-un singur program Arduino, care integrează toate componentele și le coordonează prin utilizarea funcției millis() pentru a evita blocarea execuției.

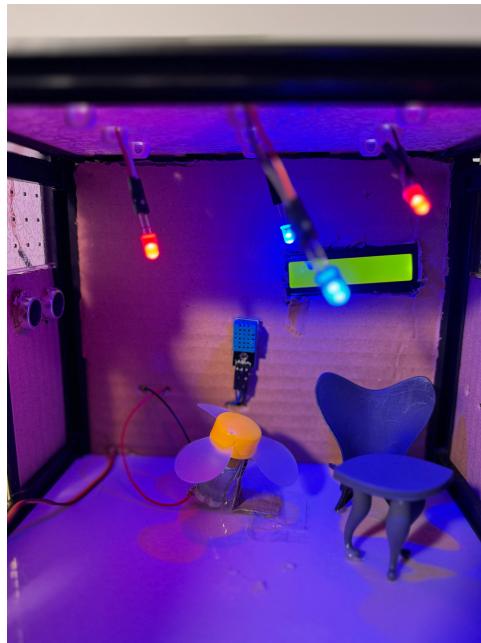


Figure 15: Vedere interior

În timpul testării integrate, am verificat dacă toate componentele pot funcționa în paralel fără interferențe. LED-urile au reacționat corect la nivelul de lumină ambientală.

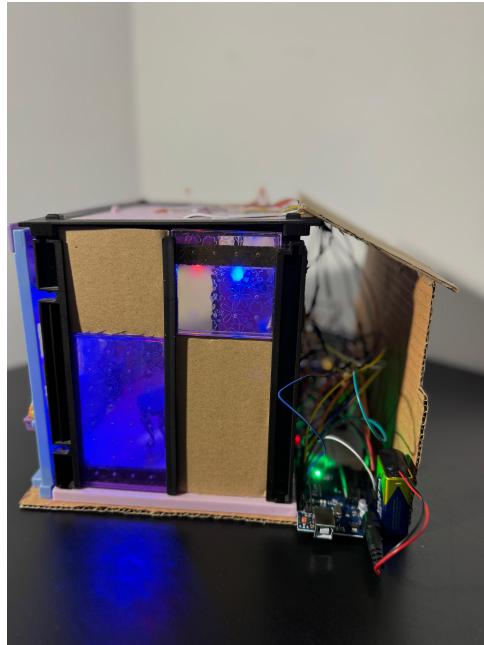


Figure 16: Vedere laterală

Senzorul ultrasonic a detectat corect apropierea unui obiect și a declansat actionarea servomotorului (pentru deschiderea ușii) și redarea unei secvențe sonore prin buzzer, cu revenirea la poziția inițială după câteva secunde.

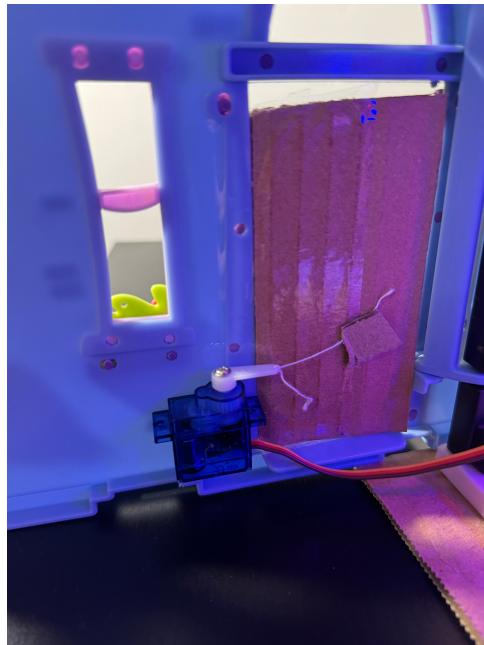


Figure 17: Ușă și servomotor

Senzorul DHT11 a transmis constant valori de temperatură și umiditate către LCD, care s-au afisat în mod clar și fără îintreruperi. Ventilatorul controlat prin releu s-a activat automat atunci când temperatura a depășit pragul de 26°C și s-a dezactivat când temperatura a scăzut sub această valoare.

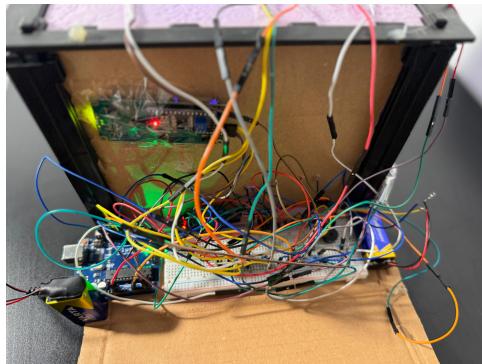


Figure 18: Vedere spate montaj



Figure 19: Vedere de sus

4 Concluzie

Proiectul realizat a demonstrat cu succes posibilitatea construirii unui sistem funcțional de tip casă inteligentă, utilizând platforma Arduino și o serie de componente electronice de bază. Prin integrarea unor senzori și actuatori precum fotorezistor, senzor ultrasonic HC-SR04, servomotor, buzzer, senzor DHT11, afișaj LCD și releu pentru ventilator, am reușit să dezvoltăm un prototip care simulează în mod eficient automatizarea unor acțiuni uzuale într-o locuință modernă.

Testarea sistemului a confirmat funcționarea corectă a fiecărui modul, atât individual, cât și în cadrul sistemului integrat. Componentele au fost sincronizate printr-o programare atent structurată, utilizând funcția millis() pentru a permite rularea lor în paralel, fără blocaje. Soluția finală este stabilă, eficientă și poate fi adaptată cu ușurință pentru aplicații reale de tip smart home. Proiectul demonstrează nu doar aplicabilitatea practică a noțiunilor de electronică și programare, ci și importanța unei abordări iterative și integrate în dezvoltarea unui sistem embedded.