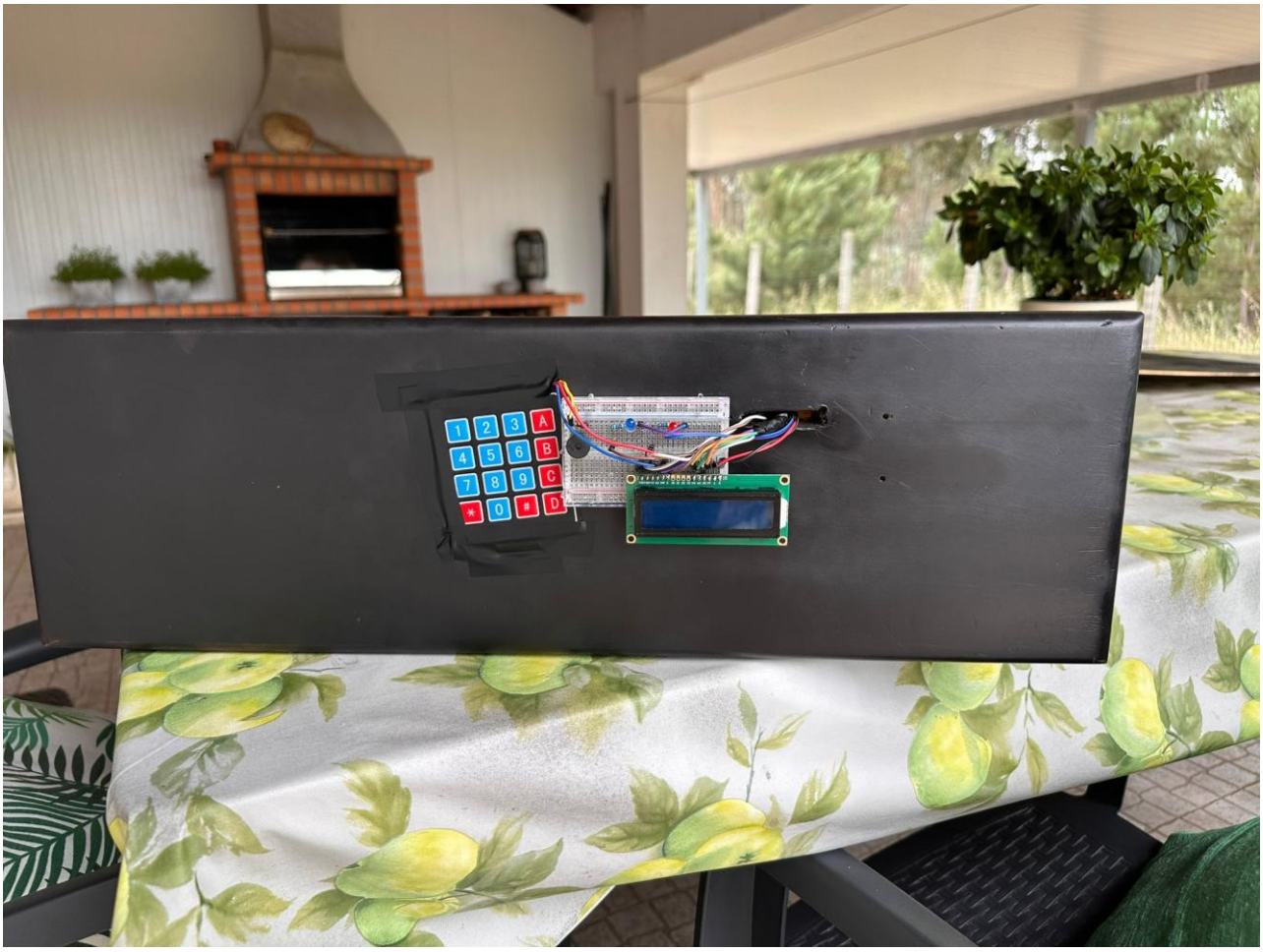
Licenciatura em Engenharia Informática

UMLOCK



ID do GRUPO:

Diogo Godinho, 27220
E: aluno27220@ipt.pt

Kanstantsin Khomchanka, 27230
E:aluno27230@ipt.pt

João Fiens, 27214
E: aluno27214@ipt.pt

Ano letivo: 2024/25

CONTEÚDO

Título do Relatório

INTRODUÇÃO/ENQUADRAMENTO

Este projeto consiste num sistema de controle de acesso com Arduino e ESP8266 integrado a uma base de dados Firebase. O Arduino lê um código numérico através de uma teclado matricial 4x4 , com recurso ao código o arduino verifica se está correto para destrancar a bloqueio (servo motor). Também deteta se o objeto que está dentro do cofre foi movido com um sensor ultrassônico e envia eventos para o ESP8266. O ESP conecta-se ao WiFi, recebe essas mensagens via serial, registra os eventos com timestamp e grava-os no Firebase Realtime Database, usando hora atual sincronizada via NTP.

OBJETIVO

Descrição: O principal objetivo deste trabalho é desenvolver um sistema de controlo de acesso de alta segurança, que requer a introdução de um código numérico para permitir a entrada. Para além do controlo de acesso, o sistema está equipado com um sensor que monitoriza a presença de um objeto no interior, registando automaticamente se este foi movido ou não. Todas as interações — incluindo tentativas de acesso corretas ou incorretas e a deteção de movimento do objeto — são registadas numa base de dados em tempo real, permitindo o acompanhamento remoto e a análise posterior dos eventos ocorridos.

FERRAMENTAS E COMPONENTES

Ferramentas: Arduino.ide; Firebase; VScode;
Componentes: Arduino Mega; ESP8266; AMS1117; Whadda WPI410; buzzer; ledR e ledB; Potenciometro; LCD 1601; Resistencias; HCSR04; ServoMotor; “Power Suply”; Teclado Matricial

DESENVOLVIMENTO DO PROJETO

Este sistema foi integrado numa estrutura física construída a partir da junção de duas gavetas, resultando numa caixa robusta, reciclada e funcional. Para a montagem dos componentes, utilizei ferramentas como o berbequim e a serra tico-tico, permitindo realizar cortes e furações precisas para a instalação do servo motor, do ecrã LCD 1602, dos LEDs, do buzzer e do próprio Arduino. Os encaixes foram planeados de forma a garantir um bom alinhamento e uma fixação segura dos módulos, prevenindo deslocamentos durante o funcionamento.
Após a instalação dos componentes, procedi à organização interna, utilizando suportes e abraçadeiras para fixar os cabos e evitar interferências ou ligações soltas. Dei também atenção ao isolamento dos contactos elétricos, de modo a garantir a segurança do sistema e prevenir curtos-circuitos. Para o acabamento exterior, limei as arestas cortadas e apliquei pequenos ajustes na madeira para melhorar a estética e a ergonomia da caixa, mantendo um aspeto limpo e funcional. O resultado final é uma caixa personalizada que acomoda de forma eficiente todos os elementos do sistema de controlo, combinando praticidade, reaproveitamento de materiais e segurança.

Arquitetura do Projeto

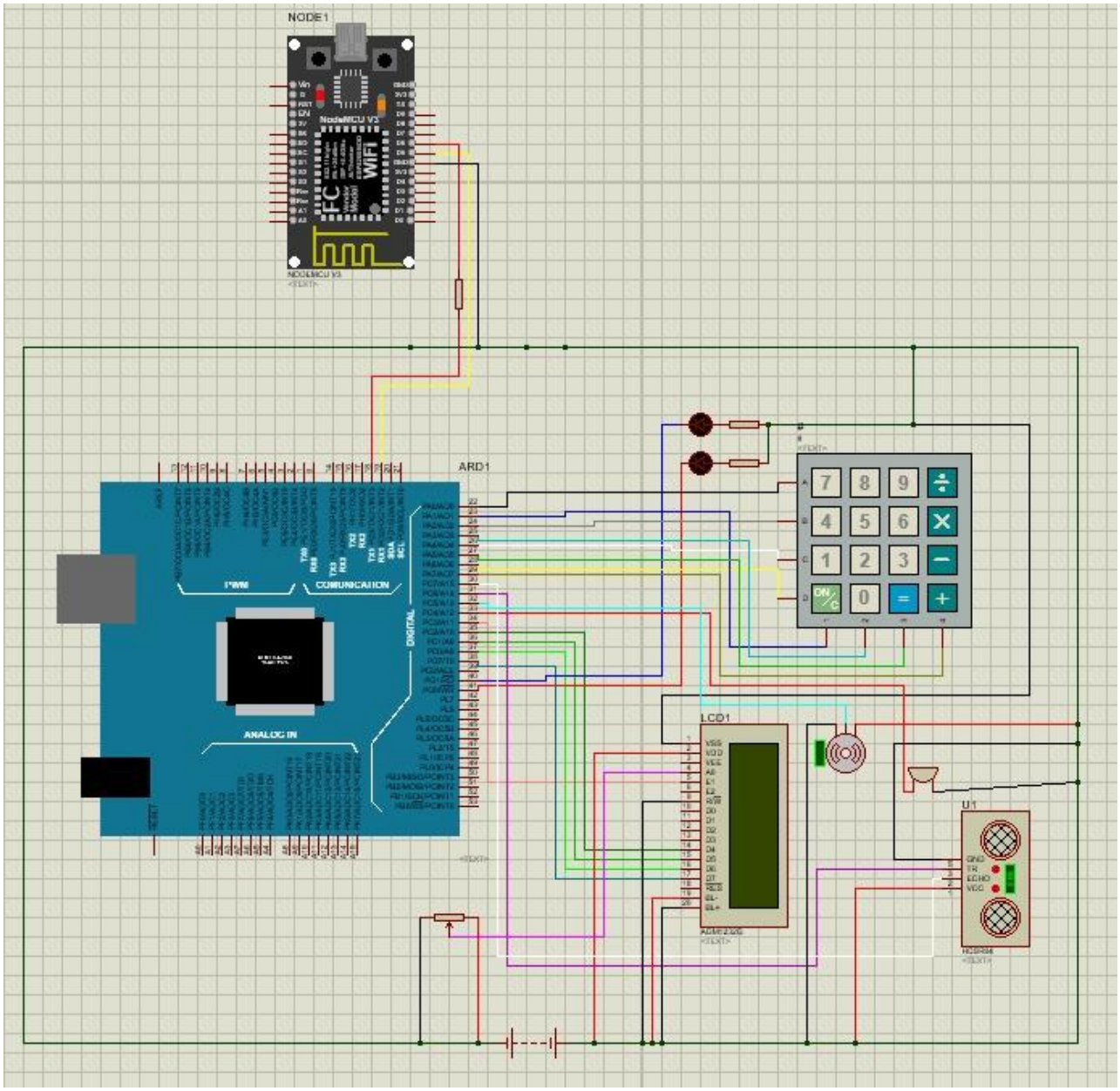


Figura 1 – Arquitetura Geral do sistema

Listagem do Código

```
Arduino Mega

#include <Keypad.h>
#include <LiquidCrystal.h>
#include <Servo.h>

// LCD: RS, EN, D4, D5, D6, D7
LiquidCrystal lcd(34, 35, 36, 37, 38, 39);

// Teclado 4x4
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'}
};
byte rowPins[ROWS] = {22, 24, 26, 28};
byte colPins[COLS] = {23, 25, 27, 29};

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

#define BUZZER 33
#define TRIG 31
#define ECHO 30
#define SERVO_PIN 32
#define LED_AZUL 40
#define LED_VERMELHO 41

Servo trancaServo;

String codigoCorreto = "1234";
String entrada = "";

int tentativas = 0;
const int maxTentativas = 5;
bool bloqueado = false;
unsigned long tempoBloqueio = 15000;
unsigned long inicioBloqueio = 0;

unsigned long ultimaDeteccao = 0;
const unsigned long intervaloBuzzer = 1000;

bool objetoDetectado = false;

bool buzzerAtivo = false;
unsigned long buzzerInicio = 0;
```

```
const unsigned long buzzerDuracao = 300;

void setup() {
  lcd.begin(16, 2);
  mostrarMensagem("Digite o codigo:", "");

  pinMode(BUZZER, OUTPUT);
  pinMode(TRIG, OUTPUT);
  pinMode(ECHO, INPUT);
  pinMode(LED_AZUL, OUTPUT);
  pinMode(LED_VERMELHO, OUTPUT);

  digitalWrite(LED_AZUL, LOW);
  digitalWrite(LED_VERMELHO, LOW);

  trancaServo.attach(SERVO_PIN);
  trancaServo.write(0); // tranca fechada

  Serial.begin(9600);
  Serial1.begin(9600);
}

void loop() {
  if (buzzerAtivo && millis() - buzzerInicio >= buzzerDuracao) {
    noTone(BUZZER);
    buzzerAtivo = false;
  }

  if (bloqueado) {
    unsigned long tempoRestante = tempoBloqueio - (millis() - inicioBloqueio);
    mostrarMensagem("ACESSO BLOQUEADO", "Tempo: " + String(tempoRestante / 1000) +
"s   ");
    if (millis() - inicioBloqueio >= tempoBloqueio) {
      bloqueado = false;
      tentativas = 0;
      entrada = "";
      mostrarMensagem("Digite o codigo:", "");
    }
    return;
  }

  verificarProximidade();

  char tecla = keypad.getKey();
  if (tecla) {
    Serial.println(tecla); // debug

    if (tecla == '#') {
      verificarCodigo();
      entrada = "";
```

```
      if (!bloqueado) {
        mostrarMensagem("Digite o codigo:", "");
      }
    } else if (tecla == '*') {
      entrada = "";
      mostrarMensagem("Codigo apagado", "");
      delay(1000);
      mostrarMensagem("Digite o codigo:", "");
    } else {
      entrada += tecla;
      lcd.setCursor(0, 1);
      lcd.print(entrada);
      lcd.print("                "); // limpa o resto da linha

      if (entrada.length() > 4) {
        entrada = "";
        mostrarMensagem("Limite excedido", "");
        delay(500);

        tentativas++;

        for (int i = 0; i < 6; i++) {
          digitalWrite(LED_VERMELHO, HIGH);
          tone(BUZZER, 300);
          delay(250);
          digitalWrite(LED_VERMELHO, LOW);
          noTone(BUZZER);
          delay(250);
        }

        if (tentativas >= maxTentativas) {
          bloqueado = true;
          inicioBloqueio = millis();
          lcd.clear();
          return;
        }

        enviarParaESP("EVENTO:Codigo errado");
        mostrarMensagem("Digite o codigo:", "");
      }
    }
  }
}

void verificarCodigo() {
  lcd.clear();

  if (entrada == codigoCorreto) {
    mostrarMensagem("CODIGO CERTO", "");
    tentativas = 0;
```

```
    enviarParaESP("EVENTO:Codigo correto");

    digitalWrite(LED_VERMELHO, LOW);
    digitalWrite(LED_AZUL, HIGH);
    abrirTranca();
    digitalWrite(LED_AZUL, LOW);

  } else {
    tentativas++;
    mostrarMensagem("CÓDIGO ERRADO", "");

    digitalWrite(LED_AZUL, LOW);
    digitalWrite(LED_VERMELHO, HIGH);
    tone(BUZZER, 300);
    delay(3000);
    noTone(BUZZER);
    digitalWrite(LED_VERMELHO, LOW);

    enviarParaESP("EVENTO:Codigo errado");

    if (tentativas >= maxTentativas) {
      bloqueado = true;
      inicioBloqueio = millis();
      lcd.clear();
    }
  }
}

void abrirTranca() {
  tone(BUZZER, 2000);
  trancaServo.write(90);
  delay(3000);
  noTone(BUZZER);
  delay(2000);
  trancaServo.write(0);
}

void verificarProximidade() {
  unsigned long agora = millis();

  if (agora - ultimaDeteccao > intervaloBuzzer) {
    digitalWrite(TRIG, LOW);
    delayMicroseconds(2);
    digitalWrite(TRIG, HIGH);
    delayMicroseconds(10);
    digitalWrite(TRIG, LOW);

    long duracao = pulseIn(ECHO, HIGH, 60000);
    float distancia = -1;
```

```
    if (duracao > 0) {
      distancia = duracao * 0.034 / 2;
    }

    if (distancia > 0 && distancia <= 20) {
      if (!objetoDetectado) {
        objetoDetectado = true;
        enviarParaESP("MOTION:Objeto detectado");
      }
    } else {
      if (objetoDetectado || ultimaDeteccao == 0) {
        objetoDetectado = false;
        enviarParaESP("MOTION:Área livre");
      }
    }
    ultimaDeteccao = agora;
  }
}

void enviarParaESP(String mensagem) {
  Serial1.println(mensagem);
  Serial.println("Enviado para ESP: " + mensagem);
}

void mostrarMensagem(String linha1, String linha2) {
  lcd.setCursor(0, 0);
  lcd.print(linha1);
  lcd.print("                ");  // limpa resto

  lcd.setCursor(0, 1);
  lcd.print(linha2);
  lcd.print("                ");  // limpa resto
}
```

Esp e bibliotecas do token e RTDB

```
#include <ESP8266WiFi.h>
#include <FirebaseESP8266.h>
#include <SoftwareSerial.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
#include <time.h>

const char* ssid = "Wazephone";
```

```
const char* password = "12345678";

FirebaseData fbdo;
FirebaseConfig config;
FirebaseAuth auth;
bool tokenReady = false;

SoftwareSerial megaSerial(D5, D6);  // RX, TX

WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org", 0, 60000); // UTC

void setup() {
  Serial.begin(9600);
  megaSerial.begin(9600);

  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWi-Fi conectado: " + WiFi.localIP().toString());

  config.api_key = "AIzaSyDvtjKEMAXRkfEJ7YGNgs-TW4H2YGzBZbM";
  config.database_url = "https://lmslock-33b95-default-rtdb.europe-
west1.firebasedatabase.app/";
  auth.user.email = "diogoalcobiagodinho@gmail.com";
  auth.user.password = "12345678";

  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);

  timeClient.begin();
}

bool emHorarioVerao(struct tm* t) {
  int ano = t->tm_year + 1900;

  // Último domingo de março
  struct tm mar = {0};
  mar.tm_year = ano - 1900;
  mar.tm_mon = 2;
  mar.tm_mday = 31;
  mar.tm_hour = 1;
  mktime(&mar);
  while (mar.tm_wday != 0) {
    mar.tm_mday--;
    mktime(&mar);
  }
```

```
  // Último domingo de outubro
  struct tm out = {0};
  out.tm_year = ano - 1900;
  out.tm_mon = 9;
  out.tm_mday = 31;
  out.tm_hour = 1;
  mktime(&out);
  while (out.tm_wday != 0) {
    out.tm_mday--;
    mktime(&out);
  }

  time_t agora = mktime(t);
  return (agora >= mktime(&mar) && agora < mktime(&out));
}

void loop() {
  if (Firebase.ready()) {
    if (!tokenReady) {
      Serial.println("☑ Token Firebase pronto!");
      tokenReady = true;
    }
  } else if (tokenReady) {
    Serial.println("🔁 Renovar token...");
    tokenReady = false;
    Firebase.begin(&config, &auth);
    Firebase.reconnectWiFi(true);
  }

  timeClient.update();

  if (tokenReady && megaSerial.available()) {
    String codigo = megaSerial.readStringUntil('\n');
    codigo.trim();
    if (codigo.length() == 0) return;

    Serial.println("Recebido: " + codigo);

    time_t rawTime = timeClient.getEpochTime();
    struct tm* t = gmtime(&rawTime);
    if (emHorarioVerao(t)) {
      t->tm_hour += 1;
      mktime(t);
    }

    char dataHora[25];
    sprintf(dataHora, "%02d/%02d/%04d %02d:%02d:%02d",
            t->tm_mday, t->tm_mon + 1, t->tm_year + 1900,
            t->tm_hour % 24, t->tm_min, t->tm_sec);
```

```
    FirebaseJson novaMsg;
    novaMsg.set("codigo", codigo);
    novaMsg.set("hora", dataHora);

    FirebaseJsonArray mensagensArray;
    if (Firebase.getArray(fbdo, "/logs/mensagens")) {
      mensagensArray = fbdo.jsonArray();
      while (mensagensArray.size() >= 15) {
        mensagensArray.remove(0);
      }
    }

    mensagensArray.add(novaMsg);
    if (Firebase.setArray(fbdo, "/logs/mensagens", mensagensArray)) {
      Serial.println("📤 Lista atualizada com sucesso");
    } else {
      Serial.println("❌ Erro: " + fbdo.errorReason());
    }

    Firebase.setJSON(fbdo, "/logs/ultima", novaMsg);
  }
}
```

```
#ifndef TOKEN_HELPER_H
#define TOKEN_HELPER_H

void printTokenStatus(FirebaseData &fbdo) {
  if (fbdo.signer.error.code != 0) {
    Serial.printf("❌ Erro no token: %s\n", fbdo.signer.error.message.c_str());
  }
}

#endif
```

```
#include <SoftwareSerial.h>

#define RX_PIN D5  // Recebe do Mega
#define TX_PIN D6  // Envia para Mega

SoftwareSerial megaSerial(RX_PIN, TX_PIN);

void setup() {
  Serial.begin(115200);          // Monitor debug via USB
  megaSerial.begin(115200);      // Comunicação com Mega
  Serial.println("ESP pronto. À espera de 'PING'...");
}

void loop() {
  if (megaSerial.available()) {
    String mensagem = megaSerial.readStringUntil('\n');
```

```
    mensagem.trim();

    Serial.println("Recebido do Mega: " + mensagem);

    if (mensagem == "PING") {
      megaSerial.println("PONG");
      Serial.println("Enviado para Mega: PONG");
    }
  }
}
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
HTML(INDEX, APRESENTACAO, RELATORIO) CSS JAVASCRIPT

<!DOCTYPE html>
<html lang="pt">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <title>Início - UMLOCK</title>
  <link rel="stylesheet" href="styles.css" />
</head>
<body>
  <header>
    <h1>UMLOCK</h1>
    <p>Bem-vindo ao sistema UMLOCK</p>
  </header>

  <main style="text-align:center;">
    <p>Escolhe uma opção:</p>
    <a class="botao" href="apresentacao.html">🔐 Acesso ao Sistema</a>
    <a class="botao" href="relatorio.html">📄 Ver Relatório</a>
  </main>

  <footer>
    &copy; 2025 Waze Kapa Jota - Todos os direitos reservados
  </footer>
</body>
</html>
<!DOCTYPE html>
<html lang="pt">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <title>Apresentação - UMLOCK</title>
  <link rel="stylesheet" href="styles.css" />
</head>
<body>
  <header>
    <h1>UMLOCK</h1>
```

```
    <p>Explorando o fascinante mundo da eletrônica e tecnologia</p>
  </header>

  <main>
    <section style="text-align:center; margin-bottom: 20px;">
      <a href="index.html" class="botao">⬅ Voltar à Página Inicial</a>
    </section>

    <section class="intro">
      <p>Seja bem-vindo(a) ao espaço dedicado à eletrônica! Aqui você encontrará
imagens inspiradoras de componentes eletrônicos, circuitos e dispositivos que
tornam possível a magia da tecnologia moderna.</p>
    </section>

    <section id="loginSection"></section>

    <section id="mensagens" style="display:none;">
      <h2>Últimas Mensagens do Sistema</h2>
      <ul id="listaMensagens">
        <li>Carregando mensagens...</li>
      </ul>
    </section>
  </main>

  <footer>
    &copy; 2025 Waze Kapa Jota - Todos os direitos reservados
  </footer>

  <!-- Firebase -->
  <script src="https://www.gstatic.com/firebasejs/9.22.1/firebase-app-
compat.js"></script>
  <script src="https://www.gstatic.com/firebasejs/9.22.1/firebase-auth-
compat.js"></script>
  <script src="https://www.gstatic.com/firebasejs/9.22.1/firebase-database-
compat.js"></script>

  <script>
    const firebaseConfig = {
      apiKey: "AIzaSyDvtjKEMAXRkfEJ7YGNgs-TW4H2YGzBZbM",
      authDomain: "lmslock-33b95.firebaseapp.com",
      databaseURL: "https://lmslock-33b95-default-rtdb.europe-
west1.firebasedatabase.app",
      projectId: "lmslock-33b95",
      storageBucket: "lmslock-33b95.appspot.com",
      messagingSenderId: "701373252784",
      appId: "1:701373252784:web:0118db4f8282c9903e2775"
    };

    firebase.initializeApp(firebaseConfig);
    const auth = firebase.auth();
```

```
const db = firebase.database();
const mensagensRef = db.ref('logs/mensagens');

const listaMensagens = document.getElementById('listaMensagens');
const loginSection = document.getElementById('loginSection');

function criarFormularioLogin() {
  const loginForm = document.createElement('form');
  loginForm.id = "loginForm";
  loginForm.innerHTML = `
    <h2>Login</h2>
    <input type="email" id="email" placeholder="Email" required />
    <input type="password" id="password" placeholder="Senha" required />
    <button type="submit">Entrar</button>
    <p id="loginError" class="error-message"></p>
  `;
  loginSection.innerHTML = '';
  loginSection.appendChild(loginForm);

  loginForm.addEventListener('submit', async (e) => {
    e.preventDefault();
    const email = loginForm.querySelector('#email').value.trim();
    const password = loginForm.querySelector('#password').value.trim();
    const loginError = loginForm.querySelector('#loginError');

    try {
      await auth.signInWithEmailAndPassword(email, password);
      loginError.textContent = "";
      loginSection.style.display = 'none';
      document.getElementById('mensagens').style.display = 'block';
      startListening();
    } catch (error) {
      loginError.textContent = error.message;
    }
  });
}

function atualizarLista(mensagens) {
  listaMensagens.innerHTML = '';

  if (!mensagens || mensagens.length === 0) {
    listaMensagens.innerHTML = '<li>Nenhuma mensagem disponível.</li>';
    return;
  }

  mensagens.forEach(msg => {
    const li = document.createElement('li');
    li.textContent = msg.codigo || "Sem código";
    const span = document.createElement('span');
    span.classList.add('time');
```

```
        span.textContent = msg.hora || "";
        li.appendChild(span);
        listaMensagens.appendChild(li);
      });
    }

    function startListening() {
      mensagensRef.on('value', (snapshot) => {
        const data = snapshot.val();
        const lista = Array.isArray(data) ? data.filter(m => m !== null) :
Object.values(data);
        atualizarLista(lista);
      });
    }

    auth.onAuthStateChanged(user => {
      if (user) {
        loginSection.style.display = 'none';
        document.getElementById('mensagens').style.display = 'block';
        startListening();
      } else {
        loginSection.style.display = 'block';
        document.getElementById('mensagens').style.display = 'none';
        criarFormularioLogin();
      }
    });
  </script>
</body>
</html>

--------------------------------------------------------------------------------
--------------------------------------------------------------------------------

<!DOCTYPE html>
<html lang="pt">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <title>Relatório - UMLOCK</title>
  <link rel="stylesheet" href="styles.css" />
</head>
<body>
  <header>
    <h1>Relatório</h1>
    <p>Aqui poderás visualizar o relatório do projeto</p>
  </header>

  <main style="text-align:center;">
    <a href="relatorio_final.pdf" class="botao" target="_blank">📥 Ver
Relatório</a><br><br>
```

```
    <a href="index.html" class="botao">⬅ Voltar à Página Inicial</a>
  </main>

  <footer>
    &copy; 2025 Waze Kapa Jota - Todos os direitos reservados
  </footer>
</body>
</html>
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Css
body {
  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
  background: #121212;
  color: #eee;
  line-height: 1.6;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  padding-bottom: 40px;
  margin: 0;
}
header {
  background: #1f2937;
  padding: 20px;
  text-align: center;
  box-shadow: 0 2px 8px rgba(0,0,0,0.7);
}
header h1 {
  font-size: 2.5rem;
  color: #38bdf8;
}
header p {
  color: #94a3b8;
}
main {
  flex: 1;
  padding: 20px;
  max-width: 1000px;
  margin: auto;
}
.botao {
  display: inline-block;
  background: #38bdf8;
  color: #121212;
  padding: 12px 25px;
  border-radius: 8px;
  text-decoration: none;
  font-weight: bold;
  font-size: 1rem;
```

```
  margin: 10px;
  transition: background-color 0.3s ease;
}
.botao:hover {
  background-color: #0ea5e9;
}
footer {
  background: #1f2937;
  padding: 15px;
  text-align: center;
  color: #64748b;
  margin-top: auto;
}
#loginForm {
  max-width: 400px;
  margin: 30px auto;
  background: #1e293b;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 4px 12px rgba(56,189,248,0.3);
}
#loginForm input {
  width: 100%;
  padding: 10px;
  margin-bottom: 15px;
  border: none;
  border-radius: 6px;
}
#loginForm button {
  width: 100%;
  padding: 10px;
  background-color: #38bdf8;
  border: none;
  border-radius: 6px;
  font-weight: bold;
  cursor: pointer;
}
#mensagens ul {
  list-style: none;
  padding: 0;
  max-height: 400px;
  overflow-y: auto;
}
#mensagens ul li {
  background: #334155;
  margin-bottom: 10px;
  padding: 10px;
  border-radius: 6px;
  display: flex;
  justify-content: space-between;
```

```
}
#mensagens ul li span.time {
  font-size: 0.9rem;
  color: #94a3b8;
}

--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
javascript
// Firebase config (podes substituir pelos teus dados)
const firebaseConfig = {
  apiKey: "AIzaSyDvtjKEMAXRkfEJ7YGNgs-TW4H2YGzBZbM",
  authDomain: "lmslock-33b95.firebaseapp.com",
  databaseURL: "https://lmslock-33b95-default-rtdb.europe-
west1.firebasedatabase.app",
  projectId: "lmslock-33b95",
  storageBucket: "lmslock-33b95.appspot.com",
  messagingSenderId: "701373252784",
  appId: "1:701373252784:web:0118db4f8282c9903e2775"
};

firebase.initializeApp(firebaseConfig);

const auth = firebase.auth();
const db = firebase.database();
const mensagensRef = db.ref('logs/mensagens');

const listaMensagens = document.getElementById('listaMensagens');
const loginSection = document.getElementById('loginSection');

// Criar formulário de login
function criarFormularioLogin() {
  const form = document.createElement('form');
  form.id = "loginForm";
  form.innerHTML = `
    <h2>Login</h2>
    <input type="email" id="email" placeholder="Email" required />
    <input type="password" id="password" placeholder="Senha" required />
    <button type="submit">Entrar</button>
    <p id="loginError" class="error-message"></p>
  `;
  loginSection.innerHTML = '';
  loginSection.appendChild(form);

  form.addEventListener('submit', async (e) => {
    e.preventDefault();
    const email = form.querySelector('#email').value.trim();
    const password = form.querySelector('#password').value.trim();
    const loginError = form.querySelector('#loginError');
    try {
```

```
      await auth.signInWithEmailAndPassword(email, password);
      loginError.textContent = "";
      loginSection.style.display = 'none';
      document.getElementById('mensagens').style.display = 'block';
      startListening();
    } catch (error) {
      loginError.textContent = error.message;
    }
  });
}

function atualizarLista(mensagens) {
  listaMensagens.innerHTML = '';
  if (!mensagens || mensagens.length === 0) {
    listaMensagens.innerHTML = '<li>Nenhuma mensagem disponível.</li>';
    return;
  }
  mensagens.forEach(msg => {
    const li = document.createElement('li');
    li.textContent = msg.codigo || "Sem código";
    const span = document.createElement('span');
    span.className = "time";
    span.textContent = msg.hora || "";
    li.appendChild(span);
    listaMensagens.appendChild(li);
  });
}

function startListening() {
  mensagensRef.on('value', (snapshot) => {
    const data = snapshot.val();
    if (!data) {
      atualizarLista(null);
      return;
    }
    let lista = Array.isArray(data) ? data.filter(m => m !== null) :
Object.values(data);
    atualizarLista(lista);
  });
}

auth.onAuthStateChanged(user => {
  if (user) {
    loginSection.style.display = 'none';
    document.getElementById('mensagens').style.display = 'block';
    startListening();
  } else {
    loginSection.style.display = 'block';
    document.getElementById('mensagens').style.display = 'none';
    criarFormularioLogin();
```

```
  }
});
```

MONTAGENS (HARDWARE)

PARTE EXTERIOR

fig2 -Parte Exterior

PARTE INTERIOR

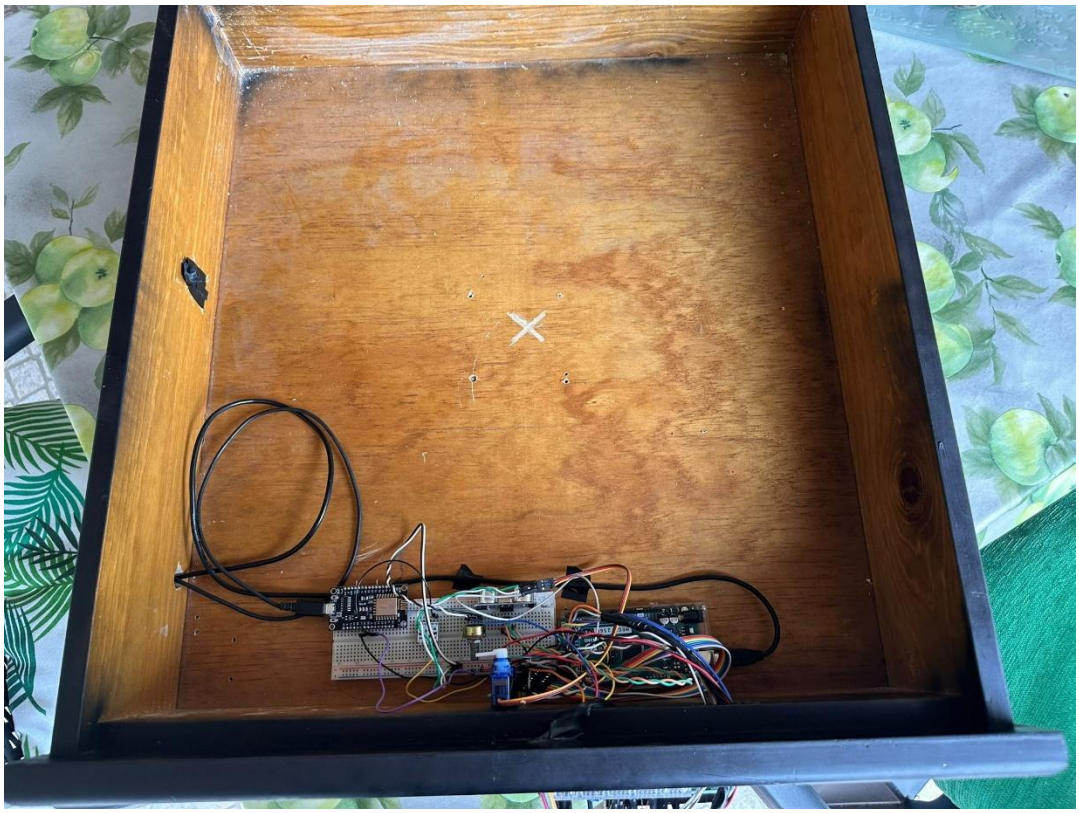


Fig.3-Parte interior do projeto

TAMPA

fig.4-Tampa do projeto

CONCLUSÕES

Concluindo tivemos vários problemas ao longo do período em que estivemos a fazer este trabalho, sendo eles a implementação de bibliotecas no arduino, que tivemos de baixar diretamente do github, as como algumas já não estavam disponíveis tivemos que as fazer com ajuda do chat gpt. Implementação de bibliotecas do esp8266, no proteus.

Ligações físicas entre o esp8266 e o arduino, tivemos que recorrer ao site do Wadda para perceber como funcionava o modulo de conversão dos sinais tx/rx.
Outro problema que ocorreu foi como implementar a firebase no meu trabalho, pois nunca tinhamos utilizado uma base de dados, logo foi um obstáculo.
O maior problema que existiu no trabalho foram as ligações serial entre o arduino e o esp8266 e a firebase. Fomos testando com o serial motor e quando conseguimos fazer a conexão entre os dois porque fizemos uma conexão bidirecional, estavam a dar caracteres incorretos, porque a alimentação de 3.3v do arduino mega não é própria para alimentar o esp8266.
Este trabalho serviu para implementar conhecimentos obtidos na aula e adientar matéria para base de dados.

Referências

ESP8266WiFi — *ESP8266WiFi Library*. GitHub. Disponível em: https://github.com/esp8266/Arduino. Acedido em: 13 de junho de 2025.

Firebase_ESP_Client — *Firebase Arduino Client Library for ESP8266 and ESP32*. GitHub - Mobizt. Disponível em: https://github.com/mobizt/Firebase-ESP-Client. Acedido em: 13 de junho de 2025.

NTPClient — *NTP Client library for ESP8266/ESP32*. GitHub - Taranais. Disponível em: https://github.com/taranais/NTPClient. Acedido em: 13 de junho de 2025.

OpenAI. *ChatGPT – Assistente de inteligência artificial baseado em linguagem natural*. Utilizado para apoio na resolução de problemas técnicos e desenvolvimento de código no projeto. Disponível em: https://chat.openai.com. Acedido em: 2 de junho de 2025 a 22 de junho.

[7] *Firebase Tutorial for Arduino and ESP8266*. YouTube. Disponível em: https://www.youtube.com/watch?v=9kRgVxULbag. Acedido em: 16 de junho de 2025.