**Documentație tehnică - Sistem Inteligent de Management al Restaurantului**

## **📌 Descrierea problemei**

Restaurantele se confruntă frecvent cu provocări legate de gestionarea eficientă a comenzilor și a stocurilor, ceea ce duce la risipă alimentară semnificativă. Această risipă nu doar că afectează profitabilitatea afacerii, dar contribuie și la probleme de mediu, precum emisiile de gaze cu efect de seră generate de alimentele irosite. În plus, lipsa unei comunicări eficiente între personalul restaurantului și clienți poate duce la erori în comenzi și la o experiență nesatisfăcătoare pentru consumatori.

## **💡 Descrierea soluției propuse**

SIMR este un sistem integrat care combină aplicații software și componente hardware pentru a optimiza gestionarea comenzilor și a stocurilor într-un restaurant. Soluția include:

* O aplicație destinată clienților, dezvoltată în Kotlin, care permite vizualizarea meniului și plasarea comenzilor personalizate.
* O aplicație pentru manageri, bucătari și chelneri, realizată în C#, care facilitează gestionarea comenzilor și monitorizarea stocurilor în timp real.
* Un sistem hardware format din senzori de presiune conectați la un microcontroler Arduino, amplasați sub o magazie, pentru a măsura precis cantitatea de alimente disponibile.
* Integrarea cu Firebase pentru stocarea și sincronizarea datelor în timp real, precum și pentru trimiterea de notificări automate atunci când stocurile scad sub un prag prestabilit.

## **🎯 Definirea publicului țintă**

Publicul țintă al proiectului include:

* Restaurante de dimensiuni mici și medii care doresc să își optimizeze operațiunile și să reducă risipa alimentară.
* Lanțuri de restaurante interesate de implementarea unor soluții tehnologice moderne pentru gestionarea eficientă a stocurilor și comenzilor.
* Manageri și bucătari care caută instrumente intuitive pentru monitorizarea și controlul inventarului.
* Clienți care apreciază o experiență de comandă rapidă și personalizată.

## **⚙️ Prezentarea funcționalităților aplicației**

### **Aplicația pentru clienți (Kotlin):**

* Vizualizarea meniului digital al restaurantului.
* Plasarea comenzilor personalizate.
* Integrare cu sistemul de drive-through pentru preluarea rapidă a comenzilor.

**Aplicația pentru manageri, bucătari și chelneri (C#):**

* Gestionarea angajaților
* Monitorizarea în timp real a comenzilor primite.
* Gestionarea și actualizarea statusului comenzilor.
* Vizualizarea stocurilor disponibile și primirea de alerte automate când acestea scad sub un anumit prag.
* Comenzi automatizate de materie prima
* Statistici în timp real privind comenzile de ingrediente și preparate

### **Sistemul hardware:**

* Senzori de presiune montați sub o magazie pentru măsurarea precisă a cantității de alimente.
* Microcontroler Arduino care colectează datele de la senzori și le transmite către Firebase.

## **🧱 Arhitectura aplicației**

Sistemul SIMR este compus din următoarele componente:

* Frontend: Aplicație pentru clienți (Kotlin) și aplicație pentru manageri/bucătari/chelneri (C#).
* Backend: Firebase pentru stocarea datelor, autentificare și trimiterea de notificări.
* Hardware: Senzori de presiune conectați la un microcontroler Arduino, amplasați sub o magazie mobilă pe șine.
* Comunicare: Datele colectate de senzori sunt transmise către Firebase, iar aplicațiile frontend se sincronizează în timp real cu backend-ul pentru actualizarea informațiilor.

## **🌟 Elemente distinctive ale aplicației / puncte forte în comparație cu competiția**

* Integrarea completă între aplicațiile software și sistemul hardware pentru o monitorizare precisă a stocurilor.
* Utilizarea senzorilor de greutate pentru măsurarea în timp real a cantității de alimente, reducând astfel risipa.
* Notificări automate pentru manageri atunci când stocurile scad sub un prag prestabilit, facilitând reaprovizionarea la timp.
* Interfață intuitivă pentru clienți, permițând o experiență de comandă rapidă și personalizată.
* Utilizarea Firebase pentru sincronizarea în timp real a datelor și pentru trimiterea de notificări.

## **🛠️ Ghid de instalare și configurare a aplicației**

1. **Configurarea hardware**:  
   * Asamblați magazia și montați senzorii de presiune sub aceasta.
   * Conectați senzorii la microcontrolerul Arduino.
   * Programați Arduino pentru a colecta datele de la senzori și a le transmite către Firebase.
2. **Instalarea aplicațiilor**:  
   * Pentru aplicația clienților, instalați-o pe o tabletă sau integrați-o într-un kiosk interactiv.
   * Pentru aplicația managerilor/bucătarilor/chelneri, instalați-o pe dispozitivele utilizate în bucătărie/sala de mese sau birou.
3. **Testare și calibrare**:  
   * Testați funcționalitatea senzorilor și calibrați-i pentru a asigura măsurători precise.
   * Verificați sincronizarea datelor între hardware, Firebase și aplicațiile frontend.

## **🧠 Justificarea folosirii tehnologiilor alese**

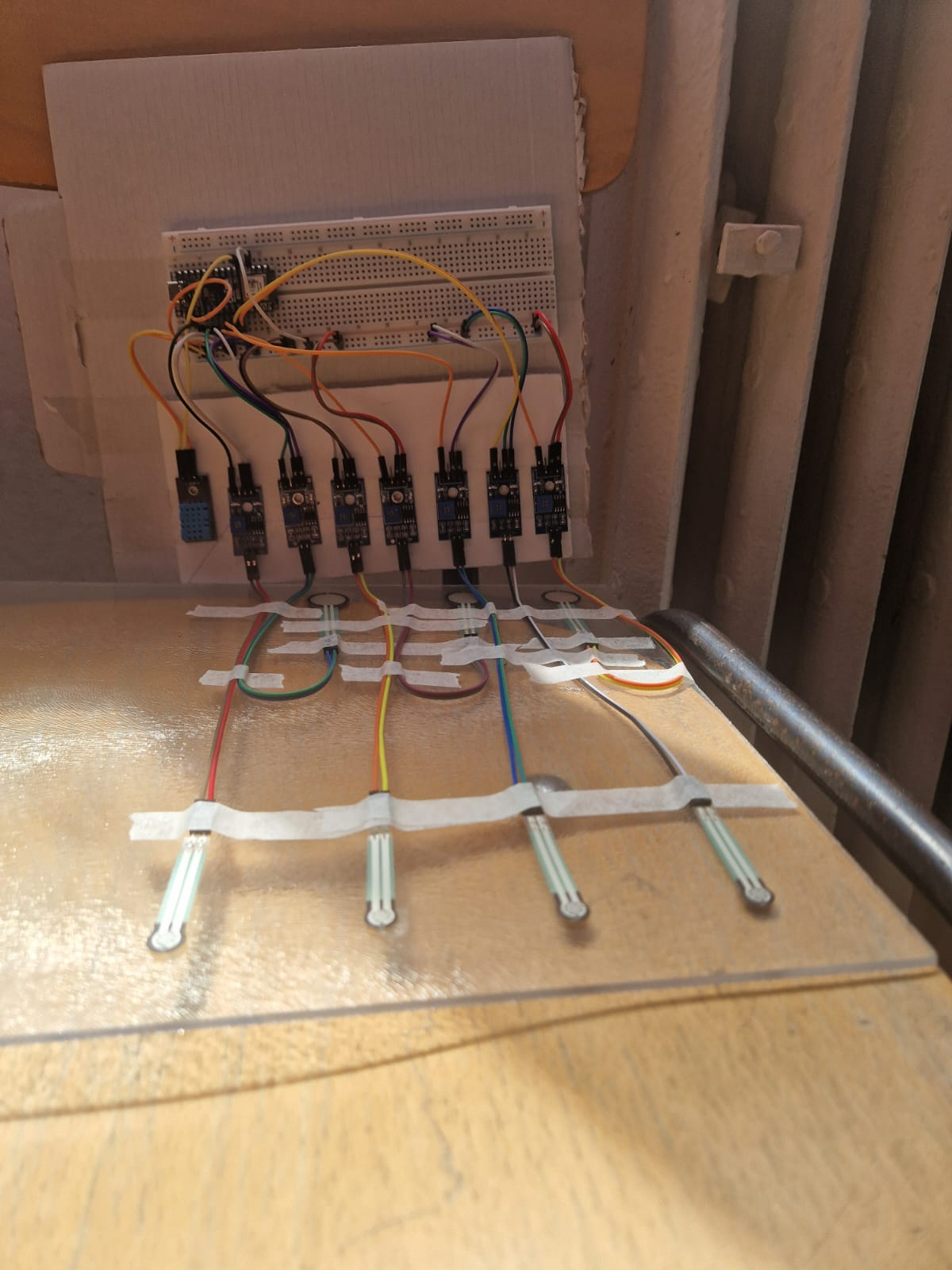
Am ales să utilizăm Firebase datorită capacităților sale de sincronizare în timp real, autentificare ușoară și trimitere de notificări push, toate esențiale pentru funcționarea eficientă a sistemului nostru. Pentru partea hardware, Arduino oferă o platformă flexibilă și accesibilă pentru colectarea datelor de la senzori. Limbajele JavaScript și C# au fost alese pentru dezvoltarea aplicațiilor frontend datorită popularității lor și a suportului extins pentru dezvoltarea de interfețe intuitive și funcționale.

## **💬 Opinia autorilor despre ideea proiectului și utilitatea acestuia pentru publicul țintă**

Considerăm că SIMR aduce o contribuție semnificativă în domeniul gestionării eficiente a resurselor într-un restaurant. Prin integrarea tehnologiilor moderne, oferim o soluție care nu doar că reduce risipa alimentară, dar îmbunătățește și experiența clienților și eficiența operațională a personalului. De exemplu, într-un restaurant pilot unde am testat sistemul, managerul a observat o reducere cu 30% a risipei alimentare și o îmbunătățire a timpului de procesare a comenzilor cu 25%.

## **🗺️ Roadmap**

* **Faza 1:** Realizarea unui studiu de piață
* **Faza 2**: Finalizarea dezvoltării aplicațiilor și a sistemului hardware.
* **Faza 3**: Testarea sistemului într-un mediu controlat și colectarea feedback-ului.
* **Faza 4**: Implementarea îmbunătățirilor bazate pe feedback și optimizarea performanței.
* **Faza 5**: Extinderea sistemului pentru a fi utilizat în lanțuri de restaurante și integrarea de funcționalități suplimentare, precum analiza predictivă a stocurilor.

Anexa 1 - Conectarea senzorilor de presiune la Arduino  


Anexa 2 - Conectarea ecranului HDMI pentru drive through cu Raspberry PI 5



Anexa 3 - Cod Arduino Nano ESP32

#include <WiFi.h>

#include <Firebase\_ESP\_Client.h>

#include <DHT.h>

#define FIREBASE\_HOST "restaurant-3e115-default-rtdb.europe-west1.firebasedatabase.app"

#define FIREBASE\_AUTH "AIzaSyDzUE\_U7yqtyJQu3ikQfw5rbYHC\_Dk-m9k"

#define WIFI\_SSID "Mihai"

#define WIFI\_PASSWORD "MihaiC2009"

#define UserID "SLIJplDtuncEpdZ0PuWYXDC7j0y1";

#define DHTPIN D2

#define DHTTYPE DHT11

DHT dht(DHTPIN,DHTTYPE);

FirebaseData fbdo;

FirebaseAuth auth;

FirebaseConfig config;

unsigned long sendDataPrevMillis = 0;

const int interval = 2000;

void setup() {

pinMode(A0, INPUT);

pinMode(A1, INPUT);

pinMode(A2, INPUT);

Serial.begin(115200);

WiFi.begin (WIFI\_SSID, WIFI\_PASSWORD);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

dht.begin();

Serial.println ("");

Serial.println ("WiFi Connected!");

config.database\_url = FIREBASE\_HOST;

config.signer.tokens.legacy\_token = FIREBASE\_AUTH;

Firebase.begin(&config, &auth);

Firebase.reconnectWiFi(true);

}

void loop() {

if (millis() - sendDataPrevMillis > interval) {

sendDataPrevMillis = millis();

int sensor1 = 4095-analogRead(A0);

int sensor2 = 4095-analogRead(A1);

int sensor3 = 4095-analogRead(A2);

int sensor4 = 4095-analogRead(A3);

int sensor5 = 4095-analogRead(A4);

int sensor6 = 4095-analogRead(A5);

int sensor7 = 4095-analogRead(A6);

float temp=dht.readTemperature();

float umiditate=dht.readHumidity();

Serial.printf("Sensor1: %d | Sensor2: %d | Sensor3: %d | Sensor4: %d | Sensor5: %d | Sensor6: %d | Sensor7: %d \n", sensor1, sensor2, sensor3, sensor4, sensor5, sensor6, sensor7);

Firebase.RTDB.setInt(&fbdo, "/kitchen/SLIJplDtuncEpdZ0PuWYXDC7j0y1/ingredients/list/1/quantity", sensor1);

Firebase.RTDB.setInt(&fbdo, "/kitchen/SLIJplDtuncEpdZ0PuWYXDC7j0y1/ingredients/list/2/quantity", sensor2);

Firebase.RTDB.setInt(&fbdo, "/kitchen/SLIJplDtuncEpdZ0PuWYXDC7j0y1/ingredients/list/3/quantity", sensor3);

Firebase.RTDB.setInt(&fbdo, "/kitchen/SLIJplDtuncEpdZ0PuWYXDC7j0y1/ingredients/list/4/quantity", sensor4);

Firebase.RTDB.setInt(&fbdo, "/kitchen/SLIJplDtuncEpdZ0PuWYXDC7j0y1/ingredients/list/5/quantity", sensor5);

Firebase.RTDB.setInt(&fbdo, "/kitchen/SLIJplDtuncEpdZ0PuWYXDC7j0y1/ingredients/list/6/quantity", sensor6);

Firebase.RTDB.setInt(&fbdo, "/kitchen/SLIJplDtuncEpdZ0PuWYXDC7j0y1/ingredients/list/7/quantity", sensor7);

Firebase.RTDB.setInt(&fbdo, "/users/SLIJplDtuncEpdZ0PuWYXDC7j0y1/DHT/temp", temp);

Firebase.RTDB.setInt(&fbdo, "/users/SLIJplDtuncEpdZ0PuWYXDC7j0y1/DHT/umd", umiditate);

}

}