PROJECT CODE

ARDUINO Program

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <ArduinoJson.h>
#include <SPI.h>
#include <MFRC522.h>

#define RST_PIN D3
#define SS_PIN D4
#define TRIG_PIN D5
#define ECHO_PIN D6
#define BUZZER_PIN D7

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

const char* supabaseUrl = "https://your-project.supabase.co/rest/v1";
const char* supabaseKey =
"YOUR_SUPABASE_ANON_OR_SERVICE_ROLE_KEY";

// Your table names
const char* inventoryTable = "inventory";
const char* rfidLogsTable = "rfid_logs";

MFRC522 mfrc522(SS_PIN, RST_PIN);

int inventory = 5;
const int maxStock = 5;
bool alerted = false;
bool restocking = false;

long distance;
unsigned long lastPenTime = 0;
bool firstPenTaken = false;

WiFiClientSecure client; // for HTTPS requests
```

```
void setup() {
  Serial.begin(115200);

  WiFi.begin(ssid, password);
  Serial.print("Connecting to WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi connected");

  // Setup pins
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  digitalWrite(BUZZER_PIN, LOW);

  SPI.begin();
  mfrc522.PCD_Init();

  client.setInsecure(); // Disable certificate verification (not secure for prod!)

  uploadInventory(inventory); // Initial upload
}

void loop() {
  // Ultrasonic sensor reading
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  distance = pulseIn(ECHO_PIN, HIGH) * 0.034 / 2;

  if (distance < 20 && !restocking) {
    unsigned long currentMillis = millis();
```

```
  if (!firstPenTaken) {
    delay(4000);
    inventory--;
    Serial.println("Pen taken");
    firstPenTaken = true;
    lastPenTime = currentMillis;
    uploadInventory(inventory);
  } else if (currentMillis - lastPenTime >= 5000) {
    if (inventory > 0) {
      inventory--;
      Serial.println("Pen taken");
      lastPenTime = currentMillis;
      uploadInventory(inventory);
    }
  }

  if (inventory == 1 && !alerted) {
    beepBuzzer(5000);
    alerted = true;
  }

  if (inventory == 0) {
    beepBuzzer(5000);
    restocking = true;
    alerted = false;
  }
  delay(1000);
}

if (restocking) {
  inventory = maxStock;
  Serial.println("Restocked");
  uploadInventory(inventory);
  restocking = false;
  firstPenTaken = false;
}

checkRFID();
```

```
}

void checkRFID() {
  if (mfrc522.PICC_IsNewCardPresent() && mfrc522.PICC_ReadCardSerial()) {
    String uid = "";
    for (byte i = 0; i < mfrc522.uid.size; i++) {
      if (mfrc522.uid.uidByte[i] < 0x10) uid += "0"; // pad leading 0
      uid += String(mfrc522.uid.uidByte[i], HEX);
    }
    uid.toUpperCase();
    Serial.print("RFID UID: ");
    Serial.println(uid);

    logRFID(uid);
    mfrc522.PICC_HaltA();
  }
}

void uploadInventory(int count) {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient https;
    String url = String(supabaseUrl) + "/" + inventoryTable + "?id=eq.1";

    // Build JSON body for PATCH (update where id=1)
    String jsonBody = "{\"count\":";
    jsonBody += String(count);
    jsonBody += "}";

    https.begin(client, url);
    https.addHeader("apikey", supabaseKey);
    https.addHeader("Authorization", "Bearer " + String(supabaseKey));
    https.addHeader("Content-Type", "application/json");

    int httpCode = https.PATCH(jsonBody);
    if (httpCode > 0) {
      String payload = https.getString();
      Serial.print("Inventory updated: ");
      Serial.println(payload);
```

```
    } else {
      Serial.print("Error updating inventory: ");
      Serial.println(httpCode);
    }
    https.end();
  }
}

void logRFID(String uid) {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient https;
    String url = String(supabaseUrl) + "/" + rfidLogsTable;

    // Build JSON body for POST (insert)
    String jsonBody = "{\"uid\":\"" + uid + "\", \"timestamp\": \"now()\"}";

    https.begin(client, url);
    https.addHeader("apikey", supabaseKey);
    https.addHeader("Authorization", "Bearer " + String(supabaseKey));
    https.addHeader("Content-Type", "application/json");

    int httpCode = https.POST(jsonBody);
    if (httpCode > 0) {
      String payload = https.getString();
      Serial.print("RFID logged: ");
      Serial.println(payload);
    } else {
      Serial.print("Error logging RFID: ");
      Serial.println(httpCode);
    }
    https.end();
  }
}

void beepBuzzer(int durationMs) {
  digitalWrite(BUZZER_PIN, HIGH);
  delay(durationMs);
  digitalWrite(BUZZER_PIN, LOW);
```

}