#include <WiFi.h>

#include <NTPClient.h>

#include <WiFiUdp.h>

#include <FirebaseESP32.h>  // Include the Firebase ESP32 library

// WiFi credentials

const char\* ssid = "Pixel\_6pro";

const char\* password = "12345687";

// Firebase credentials

#define FIREBASE\_HOST "https://l1project-681ca-default-rtdb.firebaseio.com"

#define FIREBASE\_API\_KEY "AIzaSyDPwA\_y2ZVycvIoek0bmrobve6BT1G-FZ0"

#define USER\_EMAIL "224094@example.com"

#define USER\_PASSWORD "amma1234"

// Define Firebase objects

FirebaseData firebaseData;

FirebaseAuth firebaseAuth;

FirebaseConfig firebaseConfig;

const int buttonPin = 34;     // the number of the pushbutton pin

const unsigned long interval = 7000;  // interval duration in milliseconds (7 seconds)

unsigned long starttime = 0;  // variable for storing the start time

bool timing = false;          // flag to indicate if the button press timing is active

WiFiUDP ntpUDP;

NTPClient timeClient(ntpUDP, "pool.ntp.org", 19800, 60000); // UTC +5:30 for Sri Lanka

void setup() {

  // Initialize Serial Monitor

  Serial.begin(115200);

  // Connect to Wi-Fi

  WiFi.begin(ssid, password);

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

    delay(1000);

    Serial.println("Connecting to WiFi...");

  }

  Serial.println("Connected to WiFi");

  // Initialize the NTPClient to get time

  timeClient.begin();

  // Initialize the pushbutton pin as an input

  pinMode(buttonPin, INPUT);

  // Configure Firebase

  firebaseConfig.host = FIREBASE\_HOST;

  firebaseConfig.api\_key = FIREBASE\_API\_KEY;

  firebaseAuth.user.email = USER\_EMAIL;

  firebaseAuth.user.password = USER\_PASSWORD;

  Firebase.begin(&firebaseConfig, &firebaseAuth);

  Firebase.reconnectWiFi(true);

}

void loop() {

  // Read the state of the pushbutton value

  int buttonState = digitalRead(buttonPin);

  if (buttonState == HIGH) { // button pressed

    if (!timing) {

      // Start timing if not already started

      starttime = millis();

      timing = true;

    }

  } else { // button released

    if (timing) {

      // Check if the button was pressed for at least the required interval

      if (millis() - starttime >= interval) {

        // Update NTP time

        timeClient.update();

        // Get the formatted date and time

        String formattedTime = timeClient.getFormattedTime();

        // Add milliseconds

        unsigned long ms = millis() % 1000;

        char timeWithMillis[25];

        snprintf(timeWithMillis, sizeof(timeWithMillis), "%s:%03lu", formattedTime.c\_str(), ms); // save release time in timeWithMillis

        Serial.println("Button released at: " + String(timeWithMillis));

        // Fetch the current limitCounter value from Firebase

        if (Firebase.getInt(firebaseData, "/runner/limitCounter")) {

          int limitCounter = firebaseData.intData();

          Serial.print("Fetched limitCounter: ");

          Serial.println(limitCounter);

          // Update Firebase with the recorded time

          String termPath = "/runner";  // Define the path to store the time

          String pathWithCounter = termPath + "/" + String(limitCounter) + "/Rtime";

          // Store the time in Firebase

          if (Firebase.setString(firebaseData, pathWithCounter, timeWithMillis)) {

            Serial.println("Time stored successfully.");

            // Increment the counter for the next entry

            limitCounter++;

            // Update limitCounter in Firebase

            if (Firebase.setInt(firebaseData, termPath + "/limitCounter", limitCounter)) {

              Serial.println("Counter updated successfully.");

            } else {

              Serial.print("Error updating counter: ");

              Serial.println(firebaseData.errorReason());

            }

          } else {

            Serial.print("Error storing time: ");

            Serial.println(firebaseData.errorReason());

          }

        } else {

          Serial.print("Error getting limitCounter: ");

          Serial.println(firebaseData.errorReason());

        }

      }

      // Reset timing

      timing = false;

    }

  }

}