**MRP TEPLOMER**

**Členovia tímu:**

* Radovan Žiak
* Marek Guráň
* Peter Spišiak

**Cieľ projektu:**

* Cieľom tohto projektu je vytvoriť zariadenie, ktoré bude čítať dáta zo senzoru DHT11 a posielať ich na Firebase databázu a zobrazovať na Nextion obrazovke. Vytvorená Android aplikácia bude komunikovať s Firebase databázou, stiahne z nej dáta a v reálnom čase ich bude zobrazovať.

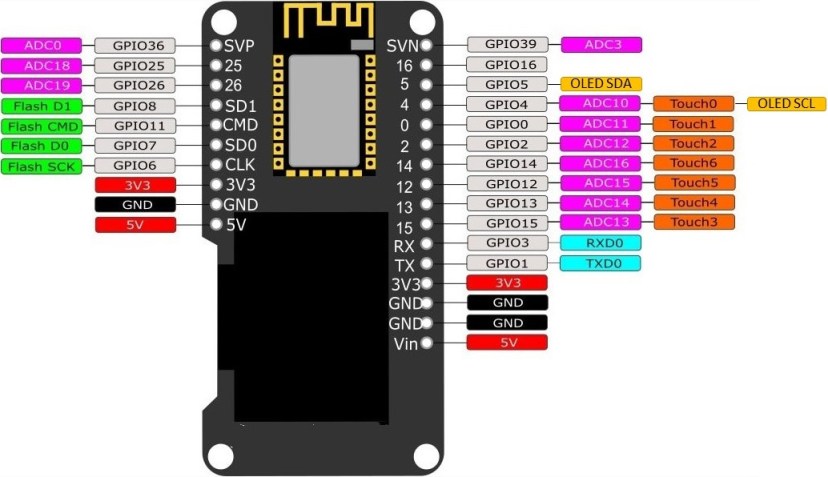
**Súčiastky:**

* LoLin32 ESP32
* Senzor DHT11
* Nextion NX4024T032 displej
* F-F GPIO kábliky
* Krabička

**LoLin32 ESP32:**

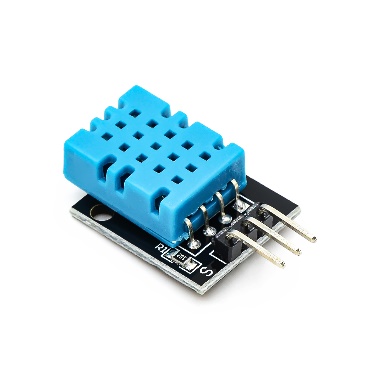
* Typ: ESP32 WROOM ESPRESSIF
* Pracovné napätie: 2.2V – 3.6V
* Prúd v režime spánku: 2 μA
* Procesor: Dual Core Tensilica LX6 240 MHz s výkonom  600 DMIPS
* SRAM: 520 kb
* FLASH: 4 MB
* Wifi protokoly: 802.11 B/G/N/E/I
* Bluetooth: 2 režimy, tradičný a s nízkou spotrebou

**PINOUT:**



**DHT11:**

* Meranie vlhkosti: 20% – 90% RH, ±5% RH
* Meranie teploty: 0 ~ 50 °C, ±2°C
* Operačné napätie: 3.3V ~ 5V



**Nextion NX4024T032:**

* Napájanie: 5V, ~85 mA
* Komunikácia: UART
* Veľkosť displeja: 3.2″ (3.2 palcov) – 69.90 mm x 41.76 mm
* Rozlíšenie: 400 x 240 pixelov
* Farby: 65k
* Flash: 4M
* Možnosť ladenia intenzity svetla
* Zabudovaný micro SD slot pre firmware upgrade

**Schéma zapojenia:**

Obrázok, na ktorom je text, kresba, náčrt, biela tabuľa

Automaticky generovaný popis

**Finálny produkt:**







**Prílohy:**

* Kód ESP32
* Kód Android

**Príloha 1:**

**Kód ESP32:**

//Knižnice

#include <Wire.h>

#include <Adafruit\_Sensor.h>

#include "DHT.h"

#include <ArduinoJson.h>

#include <WiFi.h>

#include <HTTPClient.h>

#include <FirebaseESP32.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#define WIFI\_SSID ""

#define WIFI\_PASSWORD ""

#define FIREBASE\_HOST ""

#define FIREBASE\_AUTH ""

#define OPENWEATHER\_API\_KEY ""

#define DHTPIN 5

#define DHTTYPE DHT11

FirebaseData firebaseData;

WiFiClient wifiClient;

HTTPClient httpClient;

//Premenné, konštanty

float temperature = 0;

float humidity = 0;

float pressure = 0;

DHT dht(DHTPIN, DHTTYPE);

void setup() {

Serial.begin(9600);

dht.begin();

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

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

delay(1000);

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

}

Serial.println("Connected to WiFi!");

// Initialize Firebase

Firebase.begin(FIREBASE\_HOST, FIREBASE\_AUTH);

}

void loop() {

delay(10000); //každých 10 sekúnd

getHumidity();

sendHumidityToNextion();

getTemperature();

sendTemperatureToNextion();

// Prepare JSON data to upload to Firebase

String json = "{ \"data\": { \"teplota\": \"" + String(temperature) + "℃\", \"vlhkost\": \"" + String(humidity) + "%\", \"tlak\": \"" + String(pressure) + " hPa\" } }";

Serial.println(json);

// Upload data to Firebase

FirebaseJson fbjsonBuffer;

fbjsonBuffer.setJsonData(json);

if (Firebase.setJSON(firebaseData, "/", fbjsonBuffer)) {

Serial.println("Data uploaded to Firebase!");

} else {

Serial.println("Failed to upload data to Firebase.");

Serial.println(firebaseData.errorReason());

}

}

//funkcie

float getTemperature()

{

temperature = dht.readTemperature();

}

float getHumidity()

{

humidity = dht.readHumidity();

}

void sendHumidityToNextion()

{

String command = "humidity.txt=\""+String(humidity,1)+"\"";

Serial.print(command);

Serial.write(0xff);

Serial.write(0xff);

Serial.write(0xff);

}

void sendTemperatureToNextion()

{

String command = "temperature.txt=\""+String(temperature,1)+"\"";

Serial.print(command);

Serial.write(0xff);

Serial.write(0xff);

Serial.write(0xff);

}

**Príloha 2:**

**Kód Android:**

package com.marekguran.esp32teplomer

import android.annotation.SuppressLint

import android.content.ContentValues.TAG

import android.content.Context

import android.net.ConnectivityManager

import android.net.Network

import android.net.NetworkCapabilities

import android.net.NetworkRequest

import android.os.Bundle

import android.os.Handler

import android.os.Looper

import android.util.Log

import android.view.LayoutInflater

import android.view.View

import android.view.ViewGroup

import android.widget.\*

import androidx.core.content.ContextCompat

import androidx.fragment.app.Fragment

import com.google.firebase.database.DataSnapshot

import com.google.firebase.database.DatabaseError

import com.google.firebase.database.FirebaseDatabase

import com.google.firebase.database.ValueEventListener

import com.marekguran.esp32teplomer.databinding.FragmentHomeBinding

class HomeFragment : Fragment() {

private var binding: FragmentHomeBinding? = null

private var database: FirebaseDatabase? = null

private val handler = Handler(Looper.getMainLooper())

private lateinit var runnable: Runnable

private lateinit var teplotaValue: TextView

private lateinit var vlhkostValue: TextView

private lateinit var wifi\_off\_teplota: ImageView

private lateinit var wifi\_off\_vlhkost: ImageView

private var isAttached: Boolean = false

override fun onAttach(context: Context) {

super.onAttach(context)

isAttached = true

}

override fun onDetach() {

super.onDetach()

isAttached = false

}

@SuppressLint("SetTextI18n")

override fun onCreateView(

inflater: LayoutInflater,

container: ViewGroup?,

savedInstanceState: Bundle?

): View? {

binding = FragmentHomeBinding.inflate(inflater, container, false)

database = FirebaseDatabase.getInstance()

binding?.let {

teplotaValue = it.teplotaValue

vlhkostValue = it.vlhkostValue

wifi\_off\_teplota = it.wifiOffTeplota

wifi\_off\_vlhkost = it.wifiOffVlhkost

}

// Vyvolané lebo sú to lateint, ktoré sa nevytvoria kým nebudú zavolané a ak by to nebolo, tak by checkInternetConnection() crashovalo aplikáciu

// Check if the fragment is attached to the activity

if (isAdded) {

// Move the code that requires the context to a later point in the fragment lifecycle

view?.post {

checkInternetConnection()

}

}

// Schedule the internet connectivity check to run every 15 seconds

runnable = object : Runnable {

override fun run() {

checkInternetConnection()

handler.postDelayed(this, 15000)

}

}

handler.post(runnable)

val connectivityManager =

requireContext().getSystemService(Context.CONNECTIVITY\_SERVICE) as ConnectivityManager

val networkCallback = object : ConnectivityManager.NetworkCallback() {

override fun onAvailable(network: Network) {

try {

// Check if the database variable is not null

if (database != null) {

// Retrieve "teplota", "vlhkost", and "vzduch" data from Firebase and update the corresponding texts

val teplotaRef = database!!.getReference("data").child("teplota")

teplotaRef.addValueEventListener(object : ValueEventListener {

override fun onDataChange(dataSnapshot: DataSnapshot) {

if (isAdded()) {

val teplota = dataSnapshot.value as? String ?: ""

binding?.teplotaValue?.text = teplota

// Extract the numerical value from the temperature string

val teplotaValue =

teplota.substringBefore("℃").toFloatOrNull() ?: 0f

if (teplotaValue < 14f) {

// Set the progress ring color to "zima"

val colorResId = R.color.zima

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)?.setTint(

ContextCompat.getColor(

requireContext(),

colorResId

)

)

binding?.temperatureDial?.setImageDrawable(

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)

)

}

if (teplotaValue >= 14f) {

// Set the progress ring color to "chladno"

val colorResId = R.color.chladno

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)?.setTint(

ContextCompat.getColor(

requireContext(),

colorResId

)

)

binding?.temperatureDial?.setImageDrawable(

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)

)

}

if (teplotaValue >= 21f) {

// Set the progress ring color to "normalna"

val colorResId = R.color.normalna

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)?.setTint(

ContextCompat.getColor(

requireContext(),

colorResId

)

)

binding?.temperatureDial?.setImageDrawable(

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)

)

}

if (teplotaValue >= 27f) {

// Set the progress ring color to "teplo"

val colorResId = R.color.teplo

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)?.setTint(

ContextCompat.getColor(

requireContext(),

colorResId

)

)

binding?.temperatureDial?.setImageDrawable(

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)

)

}

}

}

override fun onCancelled(error: DatabaseError) {

// Handle any errors that may occur while retrieving the data

// For example, you could log the error message using Log.e()

}

})

val vlhkostRef = database!!.getReference("data").child("vlhkost")

vlhkostRef.addValueEventListener(object : ValueEventListener {

override fun onDataChange(dataSnapshot: DataSnapshot) {

val vlhkost = dataSnapshot.value as? String ?: ""

binding?.vlhkostValue?.text = vlhkost

// Extract the numerical value from the temperature string

val vlhkostValue =

vlhkost.substringBefore("%").toFloatOrNull() ?: 0f

if (vlhkostValue < 30f) {

val colorResId = R.color.vlhkost\_mala

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)?.setTint(ContextCompat.getColor(requireContext(), colorResId))

binding?.humidityDial?.setImageDrawable(

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)

)

}

if (vlhkostValue >= 30f) {

val colorResId = R.color.vlhkost\_optimalna

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)?.setTint(ContextCompat.getColor(requireContext(), colorResId))

binding?.humidityDial?.setImageDrawable(

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)

)

}

if (vlhkostValue >= 60f) {

val colorResId = R.color.vlhkost\_velka

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)?.setTint(ContextCompat.getColor(requireContext(), colorResId))

binding?.humidityDial?.setImageDrawable(

ContextCompat.getDrawable(

requireContext(),

R.drawable.progress\_ring

)

)

}

}

override fun onCancelled(error: DatabaseError) {

// Handle any errors that may occur while retrieving the data

// For example, you could log the error message using Log.e()

}

})

}

} catch (e: Exception) {

Log.e(TAG, "Error occurred while retrieving data from Firebase: ${e.message}")

}

}

override fun onLost(network: Network) {

// Handle the case when the network connection is lost

}

}

val networkRequest = NetworkRequest.Builder()

.addTransportType(NetworkCapabilities.TRANSPORT\_CELLULAR)

.addTransportType(NetworkCapabilities.TRANSPORT\_WIFI)

.build()

connectivityManager.registerNetworkCallback(networkRequest, networkCallback)

return binding?.root

}

override fun onDestroyView() {

super.onDestroyView()

database = null

binding = null // ak bude zase crashovat aplikácia, tak odstrániť tento riadok, povodne má prečistiť pamäť keď sa prepne fragment alebo prejde do inej aplikácie

}

@SuppressLint("SetTextI18n")

@Suppress("DEPRECATION")

private fun checkInternetConnection() {

val connectivityManager = activity?.applicationContext?.getSystemService(Context.CONNECTIVITY\_SERVICE) as? ConnectivityManager

if (connectivityManager != null) {

val networkInfo = connectivityManager.activeNetworkInfo

if (networkInfo != null && networkInfo.isConnected) {

// If there is an internet connection, show the TextView and hide the ImageView

teplotaValue.visibility = View.VISIBLE

vlhkostValue.visibility = View.VISIBLE

wifi\_off\_teplota.visibility = View.GONE

wifi\_off\_vlhkost.visibility = View.GONE

} else {

// If there is no internet connection, hide the TextView and show the ImageView

teplotaValue.visibility = View.GONE

vlhkostValue.visibility = View.GONE

wifi\_off\_teplota.visibility = View.VISIBLE

wifi\_off\_vlhkost.visibility = View.VISIBLE

}

}

}

@Suppress("DEPRECATION")

private fun isConnectedToInternet(): Boolean {

val connectivityManager = context?.getSystemService(Context.CONNECTIVITY\_SERVICE) as? ConnectivityManager

val activeNetwork = connectivityManager?.activeNetworkInfo

return activeNetwork != null && activeNetwork.isConnectedOrConnecting

}

}