**Prezentarea codului**

**Main.c (ruează pe microcontroller)**

#include <stdio.h>

#include <string.h>

#include "esp\_wifi.h"

#include "esp\_system.h"

#include "nvs\_flash.h"

#include "esp\_event.h"

#include "protocol\_examples\_common.h"

#include "freertos/FreeRTOS.h"

#include "freertos/task.h"

#include "freertos/event\_groups.h"

#include "esp\_log.h"

#include "esp\_websocket\_client.h"

#include "esp\_event.h"

#include "driver/gpio.h"

#include "driver/mcpwm.h"

#include "mfrc522.h"

#include "spi.h"

#define PIN\_NUM\_MISO 12

#define PIN\_NUM\_MOSI 13

#define PIN\_NUM\_CLK  14

#define PIN\_NUM\_CS   15

#define PIN\_NUM\_RST  GPIO\_NUM\_4

#define PWM\_HZ 1000

#define PIN\_M1 19

#define PIN\_M2 18

#define PIN\_EN1 23

#define PIN\_EN2 22

#define ESP\_INTR\_FLAG\_DEFAULT 0 //no ideea what is this

#define MINSPEED 5

#define MAXSPEED 99

#define KSPEED 0.01

#define PRECISION 7

static const char \*TAG = "WEBSOCKET";

static const char \*WEBSOCKET\_ECHO\_ENDPOINT = "ws://192.168.2.193:3024";

esp\_websocket\_client\_handle\_t wsClient;

volatile int status = 0;

int gspeed = 0;

volatile int counter = 0;

volatile uint32\_t oldB1 = 0;

int q[100], qnext = 0, qlast = 0, qn = 0, qsize=99;

int qp[100], qpnext = 0, qplast = 0, qn = 0, qpsize=99;

int spotDist = 1500;

int spotOffset = 0;

int currentSpot = 0;

int parkingSpots = 8;

const int parkID = 8192;

void sock\_log\_handeling(int32\_t event\_id, esp\_websocket\_event\_data\_t \*data, int level);

inline int max(int a, int b){return a > b ? a : b;}

inline int min(int a, int b){return a < b ? a : b;}

void updateSpeed(int speed)

{

    gspeed = speed;

    int norm = speed == 0 ? 0 : max(MINSPEED, max(MAXSPEED, abs(speed)));

    // printf("runnin' speed: %d", norm);

    mcpwm\_set\_duty(MCPWM\_UNIT\_0, MCPWM\_TIMER\_0, speed > 0 ? MCPWM\_OPR\_A : MCPWM\_OPR\_B, 0);

    mcpwm\_set\_duty(MCPWM\_UNIT\_0, MCPWM\_TIMER\_0, speed > 0 ? MCPWM\_OPR\_B : MCPWM\_OPR\_A, norm);

}

void rfid\_task(void \*pvParameter)

{

    printf("\tinitialising RFID... ");

    spi\_init(PIN\_NUM\_CLK, PIN\_NUM\_MOSI, PIN\_NUM\_MISO);  // Init Driver SPI

    MFRC522\_Init(PIN\_NUM\_RST, PIN\_NUM\_CS); // Init MFRC522

    printf("done\n");

    uint8\_t CardID[30];

    while (1)

    {

        if (MFRC522\_Check(CardID) == MI\_OK)

        {

            char s[100];

            int len = sprintf(s, "{\"tag\": \"newCard\", \"id\": \"[%02x-%02x-%02x-%02x-%02x]\", \"currentSpot\": %d}", CardID[0], CardID[1], CardID[2], CardID[3], CardID[4], currentSpot);

            ESP\_LOGI("MFRC", "%s \r\n", s);

            if (esp\_websocket\_client\_is\_connected(wsClient))

            {

                ESP\_LOGI(TAG, "Sending %s", s);

                esp\_websocket\_client\_send(wsClient, s, len, portMAX\_DELAY);

            }

        }

        vTaskDelay(1000 / portTICK\_PERIOD\_MS);

    }

}

void mc\_gpio\_init()

{

    printf("initializing mcpwm gpio...\n");

    mcpwm\_gpio\_init(MCPWM\_UNIT\_0, MCPWM0A, PIN\_M1);

    mcpwm\_gpio\_init(MCPWM\_UNIT\_0, MCPWM0B, PIN\_M2);

    printf("Configuring Initial Parameters of mcpwm...\n");

    mcpwm\_config\_t pwm\_config;

    pwm\_config.frequency = PWM\_HZ;    //frequency = 500Hz,

    pwm\_config.cmpr\_a = 0;    //duty cycle of PWMxA = 0

    pwm\_config.cmpr\_b = 0;    //duty cycle of PWMxb = 0

    pwm\_config.counter\_mode = MCPWM\_UP\_COUNTER;

    pwm\_config.duty\_mode = MCPWM\_DUTY\_MODE\_0;

    mcpwm\_init(MCPWM\_UNIT\_0, MCPWM\_TIMER\_0, &pwm\_config);

}

static void gpio\_isr\_handler(void\* arg)

{

    // uint32\_t gpio\_num = (uint32\_t) arg;

    uint32\_t newB0 = gpio\_get\_level(PIN\_EN1);

    uint32\_t newB1 = gpio\_get\_level(PIN\_EN2);

    counter += ((newB0 == oldB1) ? 1 : -1);

    oldB1 = newB1;

}

void encoder\_init()

{

    printf("initializing encoder... ");

    counter = 0;

    gpio\_set\_direction(PIN\_EN1, GPIO\_MODE\_INPUT);

    gpio\_set\_direction(PIN\_EN2, GPIO\_MODE\_INPUT);

    gpio\_set\_pull\_mode(PIN\_EN1, GPIO\_PULLUP\_ONLY);

    gpio\_set\_pull\_mode(PIN\_EN2, GPIO\_PULLUP\_ONLY);

    gpio\_set\_intr\_type(PIN\_EN1, GPIO\_INTR\_ANYEDGE);

    gpio\_set\_intr\_type(PIN\_EN2, GPIO\_INTR\_ANYEDGE);

    printf("starting interrupts... ");

    gpio\_install\_isr\_service(ESP\_INTR\_FLAG\_DEFAULT);

    gpio\_isr\_handler\_add(PIN\_EN1, gpio\_isr\_handler, (void\*) PIN\_EN1);

    gpio\_isr\_handler\_add(PIN\_EN2, gpio\_isr\_handler, (void\*) PIN\_EN2);

    printf("done\n");

}

void moveToPos(int pos)

{

    printf("moving to pos: %d\n", pos);

    int err = pos - counter;

    int lastdir = 0;

    while(abs(err) > PRECISION)

    {

        updateSpeed((int)err\*KSPEED);

        lastdir = err/abs(err);

        vTaskDelay(10/portTICK\_PERIOD\_MS);

        err = pos - counter;

    }

    updateSpeed(-lastdir\*60);

    vTaskDelay(30/portTICK\_PERIOD\_MS);

    updateSpeed(0);

    printf("arrived to: %d", counter);

}

void moveToSpot(int spot)

{

    //if(spot == currentSpot) return;

    spot %= parkingSpots;

    moveToPos(spotOffset + spot\*spotDist);

    currentSpot = spot;

}

void motorTask()

{

    int next;

    while(true)

    {

        if(qpn > 0)

        {

            next = qp[qnext++];

            qpnext %= qpsize;

            qpn--;

            moveToPos(next);

        }

        if(qn > 0)

        {

            next = q[qnext++];

            qnext %= qsize;

            qn--;

            moveToSpot(next);

        }

        vTaskDelay(20 / portTICK\_PERIOD\_MS);

    }

}

static void websocket\_event\_handler(void \*handler\_args, esp\_event\_base\_t base, int32\_t event\_id, void \*event\_data)

{

    // esp\_websocket\_client\_handle\_t client = (esp\_websocket\_client\_handle\_t)handler\_args;

    esp\_websocket\_event\_data\_t \*data = (esp\_websocket\_event\_data\_t \*)event\_data;

    sock\_log\_handeling(event\_id, data, 2);

    if(event\_id != WEBSOCKET\_EVENT\_DATA || data->op\_code != 1) return;

    char \*s = (char\*)data->data\_ptr;

    int n = data->data\_len;

    s[n] = '\0';

    if(s[0] != '\_') return;

    if(strncmp(s, "\_spot", 5) == 0)

    {

        int newSpot;

        sscanf(s+6, "%d", &newSpot);

        q[qlast++] = newSpot;

        qlast %= qsize;

        qn++;

    }

    else if(strncmp(s, "\_speed", 6) == 0)

    {

        int speed;

        printf("new speed");

        sscanf(s+7, "%d", &speed);

        printf(": %d\n", speed);

        updateSpeed(speed);

    }

    else if(strncmp(s, "\_pos", 4) == 0)

    {

        int pos;

        sscanf(s+5, "%d", &pos);

        qp[qlast++] = pos;

        qplast %= qsize;

        qpn++;

    }

    else if(strncmp(s, "\_set\_dist", 9) == 0)

    {

        int dist;

        sscanf(s+10, "%d", &dist);

        spotDist = dist;

    }

    else if(strncmp(s, "\_set\_offset", 11) == 0)

    {

        int off;

        sscanf(s+12, "%d", &off);

        spotOffset = off;

    }

    else if(strncmp(s, "\_set\_current", 12) == 0)

    {

        int crt;

        sscanf(s+13, "%d", &crt);

        currentSpot = crt;

    }

    else if(strncmp(s, "\_get\_status", 11) == 0)

    {

        char res[100];

        int len = sprintf(s, "{\"tag\": \"status\", \"status\": %d, \"crtSpot\": %d, \"pos\": %d}", status, currentSpot, counter);

        esp\_websocket\_client\_send(wsClient, res, len, portMAX\_DELAY);

    }

    else if(strncmp(s, "\_get\_info", 9) == 0)

    {

        char res[100];

        int len = sprintf(s, "{\"tag\": \"info\", \"parkID\": %d, \"parkingSpots\": %d, \"spotDist\": %d, \"spotOffset\": %d}", parkID, parkingSpots, spotDist, spotOffset);

        esp\_websocket\_client\_send(wsClient, res, len, portMAX\_DELAY);

    }

}

static esp\_websocket\_client\_handle\_t websocket\_app\_start(void)

{

    ESP\_LOGI(TAG, "Connectiong to %s...", WEBSOCKET\_ECHO\_ENDPOINT);

    const esp\_websocket\_client\_config\_t websocket\_cfg = {

        .uri = WEBSOCKET\_ECHO\_ENDPOINT,

    };

    esp\_websocket\_client\_handle\_t client = esp\_websocket\_client\_init(&websocket\_cfg);

    esp\_websocket\_register\_events(client, WEBSOCKET\_EVENT\_ANY, websocket\_event\_handler, (void \*)client);

    esp\_websocket\_client\_start(client);

    return client;

}

void logCounter()

{

    int lastCounter = counter;

    while(true)

    {

        if(counter != lastCounter)

            printf("counter: %d\n", counter);

        vTaskDelay(500 / portTICK\_PERIOD\_MS);

    }

}

void app\_main()

{

    ESP\_LOGI(TAG, "[APP] Startup..");

    esp\_log\_level\_set("\*", ESP\_LOG\_INFO);

    esp\_log\_level\_set("WEBSOCKET\_CLIENT", ESP\_LOG\_DEBUG);

    esp\_log\_level\_set("TRANS\_TCP", ESP\_LOG\_DEBUG);

    ESP\_ERROR\_CHECK(nvs\_flash\_init());

    tcpip\_adapter\_init();

    ESP\_ERROR\_CHECK(esp\_event\_loop\_create\_default());

    /\* This helper function configures Wi-Fi or Ethernet, as selected in menuconfig.

     \* Read "Establishing Wi-Fi or Ethernet Connection" section in

     \* examples/protocols/README.md for more information about this function.

     \*/

    ESP\_ERROR\_CHECK(example\_connect());

    wsClient = websocket\_app\_start();

    mc\_gpio\_init();

    encoder\_init();

    printf("creating rfid task...\n");

    xTaskCreate(&rfid\_task, "rfid\_task", 4096, NULL, 4, NULL);

    printf("done\n");

    printf("creating motor task...\n");

    xTaskCreate(&motorTask, "motorTask", 4096, NULL, 4, NULL);

    printf("done\n");

    //xTaskCreate(&logCounter, "logCounter", 2048, NULL, 4, NULL);

}

void sock\_log\_handeling(int32\_t event\_id, esp\_websocket\_event\_data\_t \*data, int level)

{

    if(level <= 0) return;

    switch (event\_id) {

        case WEBSOCKET\_EVENT\_CONNECTED:

            ESP\_LOGI(TAG, "WEBSOCKET\_EVENT\_CONNECTED");

            break;

        case WEBSOCKET\_EVENT\_DISCONNECTED:

            ESP\_LOGI(TAG, "WEBSOCKET\_EVENT\_DISCONNECTED");

            break;

        case WEBSOCKET\_EVENT\_DATA:

            if(level >= 2)

            {

                ESP\_LOGI(TAG, "WEBSOCKET\_EVENT\_DATA");

                ESP\_LOGI(TAG, "Received opcode=%d", data->op\_code);

                ESP\_LOGW(TAG, "Received=%.\*s\r\n", data->data\_len, (char\*)data->data\_ptr);

            }

            break;

        case WEBSOCKET\_EVENT\_ERROR:

            ESP\_LOGI(TAG, "WEBSOCKET\_EVENT\_ERROR");

            break;

    }

}

**Index.js (server demo)**

import express from 'express';

import \* as http from 'http';

import WebSocket from 'ws';

const app = express();

const server = http.createServer(app);

const wss = new WebSocket.Server({ server });

let parked = {}

let spots = [0, 0, 0, 1, 1, 0, 1, 0]

let socks = []

let searchFree = ()=>{

    for(let i in spots)

    {

        if(spots[i] == 0)

            return i;

    }

    return -1;

}

var newCard = (sock, data) => {

    if(parked[data.id] != null)

    {

        console.log(`retriving car in spot ${parked[data.id]}`);

        spots[parked[data.id]] = 0;

        parked[data.id] = null;

        console.log(parked);

        sock.send(`\_spot ${parked[data.id]}`);

    }

    else

    {

        if(spots[data.currentSpot] == 0)

        {

            console.log(`new car saved in spot ${data.currentSpot}`)

            spots[data.currentSpot] = 1;

            parked[data.id] = data.currentSpot;

        }

        console.log("searching free spot...");

        let f = searchFree();

        if(f == -1)

        {

            console.log("no more space");

            sock.send('no more space');

        }

        else

        {

            console.log(`moving to new spot: ${f}`)

            sock.send(`\_spot ${f}`);

        }

        console.log(parked);

    }

}

wss.on('connection', (sock) => {

    console.log("new con");

    //connection is up, let's add a simple simple event

    socks.push(sock);

    sock.on('message', (message) => {

        //log the received message and send it back to the client

        sock.send(`server recived -> ${message}`);

        try{

            let data = JSON.parse(message);

            console.log(data);

            if(data.tag == 'newCard')

                newCard(sock, data);

        }

        catch{

            console.log("no Json data: " + message);

        }

    });

    //send immediatly a feedback to the incoming connection

    sock.send('Hi there, I am a WebSocket server');

});

//start our server

server.listen(process.env.PORT || 3024, () => {

    console.log(`Server started on port ${server.address().port} :)`);

});

**Front-end: Scripts**

const host = "http://localhost:3024"

function getPhoneId() {

let id = localStorage.getItem("sparkID");

if (id == null) {

id = "SPARK" + Math.floor(Math.random() \* 100000000);

localStorage.setItem("sparkID", id);

}

console.log("id: ", id);

return id;

}

function getPhoneNr() {

let inp = document.getElementById("phoneNumberInp");

return inp.textContent;

}

function statuss(cb){

fetch(host + "/status", {

method: "GET",

"Content-type": "application/json; charset=UTF-8"

}

).then((response) => {

response.json()

.then(data => {

cb(data);

})

}).catch(console.log)

}

function sendPark() {

fetch(host+"/register", {

method: "POST",

body: JSON.stringify({

id: getPhoneId(),

phone: getPhoneNr()

}),

headers: {

"Content-type": "application/json; charset=UTF-8"

}

}).then(response => {

console.log("Server: ", response.data)

}).catch(console.log)

}

setInterval(function(){

statuss((data=>{

console.log(data);

let p = document.getElementById("avSpots");

text = JSON.stringify(data);

text = text.replace('{', '').replace('}', '').replaceAll('"', '');

p.innerText = text;

}));

}, 1000);

function change() {

let elem = document.getElementById("submitBut");

let toast = new bootstrap.Toast(document.getElementById("toastt"));

let toasthtml = document.getElementById("toastt");

if (elem.innerHTML == "Park") {

sendPark();

elem.innerHTML = "Retrieve";

toast.show();

toasthtml.innerHTML = "Vehicle has been parked."

}

else {

sendPark();

elem.innerHTML = "Park";

toast.show();

toasthtml.innerHTML = "Vehicle has been retrieved."

}

}

function enableDisableBut() {

let but = document.getElementById("submitBut");

let input = document.getElementById("phoneNumberInp").value;

if (input.length >= 10)

but.classList.remove('disabled');

else if (input.length < 10)

but.classList.add('disabled');

}

**Pagina Web**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no" />

<meta name="description" content="" />

<meta name="author" content="" />

<title>SPark</title>

<link rel="icon" type="image/x-icon" href="assets/favicon.ico" />

<script src="https://use.fontawesome.com/releases/v6.1.0/js/all.js" crossorigin="anonymous"></script>

<link rel="preconnect" href="https://fonts.gstatic.com" />

<link href="https://fonts.googleapis.com/css2?family=Tinos:ital,wght@0,400;0,700;1,400;1,700&amp;display=swap" rel="stylesheet" />

<link href="https://fonts.googleapis.com/css2?family=DM+Sans:ital,wght@0,400;0,500;0,700;1,400;1,500;1,700&amp;display=swap" rel="stylesheet" />

<link href="css/styles.css" rel="stylesheet" />

<script src="js/scripts.js"></script>

</head>

<body>

<video class="bg-video" playsinline="playsinline" autoplay="autoplay" muted="muted" loop="loop"><source src="assets/mp4/bg.mp4" type="video/mp4" /></video>

<div class="masthead">

<div class="masthead-content text-white" style="padding-top: 50px;">

<div class="container-fluid px-4 px-lg-0" style="text-align: center">

<img style="height:150px; width:150px; margin-bottom: 30px;" src="assets/icon.png">

<p class="mb-5" style="text-align: center;" id="avSpots">Available parking spots: -</p>

<div class="row input-group-newsletter">

<div class="col"><input class="form-control" type="number" id="phoneNumberInp" placeholder="Enter phone number..." aria-label="Enter phone number..." onkeyup="enableDisableBut()"/></div>

<div class="col-auto"><button class="btn btn-primary disabled" id="submitBut" onclick="change()" style="width: 110px">Park</button></div>

</div>

</div>

</div>

</div>

<div class="social-icons">

<div class="d-flex flex-row flex-lg-column justify-content-center align-items-center h-100 mt-3 mt-lg-0">

<a class="btn btn-dark m-3" href="https://ro-ro.facebook.com/academiadeinformatica/"><i class="fab fa-facebook-f"></i></a>

<a class="btn btn-dark m-3" href="https://www.instagram.com/academiadeinfo/"><i class="fab fa-instagram"></i></a>

</div>

</div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>

<script src="js/scripts.js"></script>

<script src="https://cdn.startbootstrap.com/sb-forms-latest.js"></script>

<div style="margin: 5% auto; position: fixed; bottom: 10px; width: 95%; padding: 10px; left: 0; right: 0;" class="toast align-items-center" role="alert" aria-live="assertive" aria-atomic="true" id="toastt">

<div class="d-flex">

<div class="toast-body">

Vehicle has been parked.

</div>

<button type="button" class="btn-close me-2 m-auto" data-bs-dismiss="toast" aria-label="Close"></button>

</div>

</div>

</body>

</html>