DEZVOLTAREA UNEI PLATFORME ONLINE PENTRU CONTORIZAREA LOCURILOR DE PARCARE DISPONIBILE ÎN PROXIMITATEA UNEI LOCAȚII

DEVELOPMENT OF AN ONLINE PLATFORM FOR MEASURING AVAILABLE PARKING SPACES NEAR A LOCATION

VLĂDOI Vasile-Bogdan
Facultatea: Inginerie Industrială și Robotică, Specializarea: Inginerie Economică Industrială,
An de studii: IV, e-mail: vlbogdan@yahoo.com

Conducător științific: Conf.dr.ing. **Mădălin-Gabriel CATANĂ**

ABSTRACT: In this paper, we can see how to build and test a piece of equipment connected to the date care base that ensures the efficiency of how to locate a place of care in the desired area. The equipment is integrated in a virtual platform for the care of data and information about the Arduino UNO board and uses several sensors from the same board to detect their presence or absence with vehicles in a place of care. The care data will be collected after counting the parking spaces will be hosted on an online WAMP platform, and the entire communication of the hardware and the online application is food for the data network.

CUVINTE CHEIE: Locuri de parcare, Arduino UNO, platforma online, senzori, bază de date.

1. Introducere

În decursul ultimilor ani numărul automobilelor a crescut foarte mult, astfel găsirea și parcarea autovehiculelor a devenit foarte complicată deoarece timpul petrecut pentru astfel de acțiuni este un timp pierdut. Găsirea unui loc de parcare in marile orașe este destul de grea, de multe ori, după ce ai scos mașina pe stradă deja te gândești, “Bun, dar unde o să las mașina?”.

De multe ori, conducătorii auto mergând la o anumită parcare, observă că toată este plină, absolut nici un loc nu este disponibil. Acest lucru îl face să caute o altă parcare, să piardă timp și să mai consume și combustibil din care rezultă destul de mult dioxid de carbon (CO2).

Pentru a beneficia de un transport mai eficient este necesar pentru a implementa un sistem de monitorizare a parcărilor. Acesta poate oferi acces în permanență la un site WEB cu mai multe detalii despre locurile de parcare, toate acestea având ca scop diminuarea consumului de carburant și respectiv eliminarea noxelor.

Motivul care m-a împins să fac această cercetare a fost unul destul de simplu. și eu sunt șofer și de multe ori pățesc să nu găsesc absolut nici un loc liber de parcare la nevoie. Și totusi, trebuie să conștientizăm că orașele trebuie să fie intr-o continua modernizare.

Astfel, tema cercetării constă în dezvoltarea unui sistem care să faciliteze conectarea la o baza de date, actualizată in permanență cu date clare a locurilor de parcare. Prin aceste metode, fiecare localnic sau vizitator poate să verifice starea unei parcări pentru a avea certitudinea că ajunge in locul dorit și poate să își parcheze autovehicolul fără nici o problemă.

2. Structura sistemului

În momentul de față, Arduino este o platformă ce se bazează pe hardware și software simplu de folosit. Plăcuțele Arduino UNO sunt capabile să citească intrări și ieșiri cum ar fi: lumina cu un senzor;

apăsarea unui buton; un mesaj; activarea unui servomotor; aprinderea unui LED; se poate publica ceva on-line sau intr-o bază de date.

Pentru a putea realiza dezvoltarea platformei online pentru contorizarea locurilor de parcare disponibile, este necesară folosirea urmatoarelor echipamente și softwuri: placă Arduino UNO, senzor ultrasonic pentru monitorizarea distanței între obiect și senzor, LED-uri de avertizare, rezistențe pentru protejarea LED-urilor, baza de date WAMP, limbaj de programare Notepad++ și Arduino.
Plăcuța Arduino UNO (Fig. 1) este realizată pe baza microcontrolerului ATmega328P. Acesta din urmă prezintă un procesor AVR cu o arhitectură de tip RISC (set restrâns de instrucțiuni). Fig. 2 prezintă caracteristicile microcontrolerului [1]:

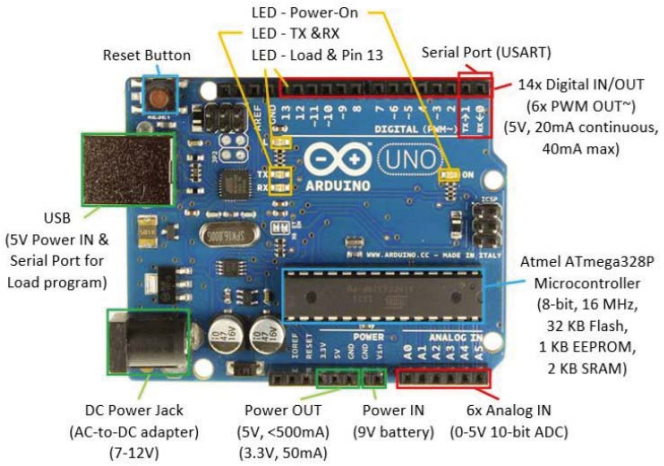


Fig. 1.

Name	Value
Program Memory Type	Flash
Program Memory Size (KB)	32
CPU Speed (MIPS/DMIPS)	20
SRAM Bytes	2,048
Data EEPROM/HEF (bytes)	1024
Digital Communication Peripher...	1-UART, 2-SPI, 1-I2C
Capture/Compare/PWM Periph...	1 Input Capture, 1 CCP, 6PWM
Timers	2 x 8-bit, 1 x 16-bit
Number of Comparators	1
Temperature Range (C)	-40 to 85
Operating Voltage Range (V)	1.8 to 5.5
Pin Count	32
Low Power	Yes

Fig. 2.

Placa dispune de 6 pini analogici cu tensiuni la intrare între 0-5V, unde sunt citite valori între 0-1023 și 14 pini digitali de intrare/ieșire, pini care sunt notați cu (~) și sunt capabili să genereze semnale (PWM).

Senzorul ultrasonic HC-SR04 este unul dintre cei mai utilizați senzori pentru aflarea distanței. În special folosit pentru proiectele cu plăci de dezvoltare Arduino UNO, are avantaje față de senzorii analogici, necesitând doar pini I/O digitali și are imunitate mai mare la zgomotul din jur.[2]

Fig. 3. Senzor ultrasonic –HC-S04

Conexiune Hardware:

- VCC- 5V Supply
- Trig- Trigger Pulse Input
- Echo-Echo Pulse Output
- GND- 0V Ground

Server Web:

WampServer.este un.mediu de dezvoltare Web, acesta.permițând crearea de aplicații web cu PHP și baza de date MySQL. Pe lângă acestea, aplicația PhpMyAdmin.permite gestionarea cu ușurință a bazei de date printr-o interfață grafică [3].

Relația este următoarea: utilizatorul (clientul) aflat în dreptul unui computer pe care are instalată o aplicație tip browser solicită (serverului) prin intermediul unui url o anumită pagină web; serverul rulează anumite linii de cod și returnează un rezultat (Fig. 4).

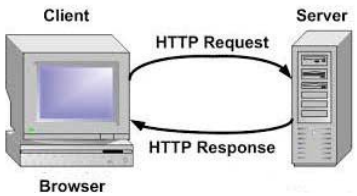


Fig. 4. Relația client- server

3. Arhitectura sistemului și a programului de comandă

Proiectarea arhitecturii sistemului s-a făcut cu ajutorul platformei Tinkercad [4], aceasta făcând parte din suita Autodesk.

Pentru început s-a ales un număr de 4 senzori de tip ultrasonic, împreună cu 4 LED-uri de avertizare. Pentru fiecare LED în parte se montează în serie cu acesta un Rezistor de 220 Ohmi pentru protejarea LED-ului. Toate componentele în parte au fost montate corespunzător pe Breadboard. (Fig.5).

În figură se prezintă modul de conectare a tuturor componentelor la plăcuța Arduino UNO (Fig. 6). Pinul VCC al senzorului se leagă la 5V al placuței, Trig (emițătorul) se leagă la un pin digital, ECHO (receptorul) la alt pin digital iar GND se leagă la masa plăcuței(GND). Se procedeaza la fel și cu LED-ul, Catodul (-) se leagă la GND iar Anodul (+) la un pin digital.

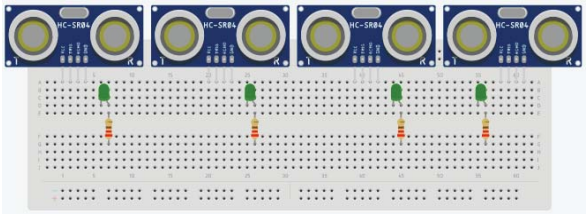


Fig. 5 Montajul componentelor

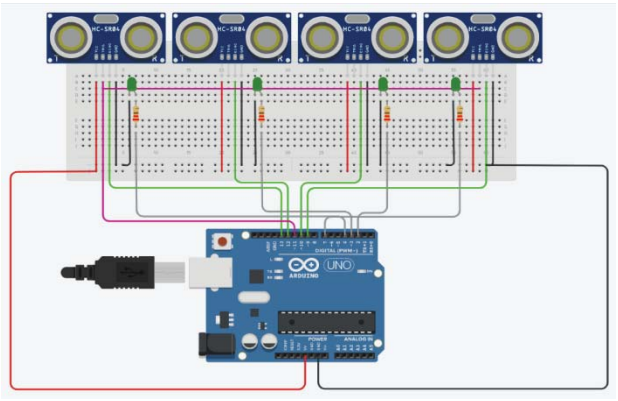


Fig. 6 Conectarea componentelor

Descrierea programului de control pentru sistemul de monitorizare al locurilor de parcare se realizează în programul Arduino (Fig. 7). Se declară toate elementele de intrare-ieșire cu pinul digital propriu, urmând să fie scris codul cu formula de calcul pentru un senzor, urmând ca programul să utilizeze aceeași formulă pentru fiecare senzor în parte.

```
int ultrasonic1=13;
int ultrasonic2=12;
int ultrasonic3=10;
int ultrasonic4=9;
int TrigPin=11;
long duration;
long distance;

int LED1=4;
int LED2=3;
int LED3=2;
int LED4=7;
int val1;
int val2;
int val3;
int val4;

int ReadSensor(int ultrasonic, int LEDposition)
{
  int valoare;

    digitalWrite(TrigPin,HIGH);
  delayMicroseconds(10);
   digitalWrite(TrigPin,LOW);

  duration = pulseIn(ultrasonic, HIGH);
  valoare=duration*0.034/2;

Serial.println("cm: ");
  Serial.print(valoare);

    if(valoare<=30)
    {digitalWrite(LEDposition, HIGH);

    }
  else{
    digitalWrite(LEDposition, LOW);
    }

}


void setup(){
  pinMode(ultrasonic1, INPUT);
  pinMode(ultrasonic2, INPUT);
  pinMode(ultrasonic3, INPUT);
  pinMode(ultrasonic4, INPUT);
  pinMode(TrigPin, OUTPUT);

  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(LED4, OUTPUT);
  Serial.begin(9600);

}

void loop(){


ReadSensor(ultrasonic1, LED1);
  delay(10);
  ReadSensor(ultrasonic2, LED2);
  delay(10);
  ReadSensor(ultrasonic3, LED3);
  delay(10);
  ReadSensor(ultrasonic4, LED4);
  delay(10);
     delay(50);


}
```

Fig. 7 Programul de control

După un anumit număr de testări practice s-a ajuns la concluzia că distanța optimă dintre senzor și autovehicul, la care LED ul să se aprindă este de 30cm (Fig. 8). În final a fost realizată și testarea programului (Fig. 9) prin intermediul aceluiași program Tinkercad, LED-urile aprinse indicând locurile ocupate, si cel stins locul liber.

Fig. 8 Functionare senzor

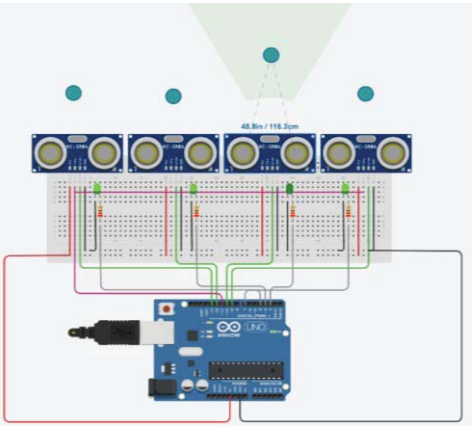


Fig. 9 Testarea programului

Implementarea bazei de date :

Baza de date creată cu ajutorul aplicației WAMP, descrisă in capitolul anterior, are rolul de a stoca toate informațiile despre parcări. În figura următoare se prezintă crearea unui tabel (Fig.10). Pentru testarea bazei de date, s-au folosit date fictive pentru monitorizarea unui singur loc de parcare cu un senzor ultrasonic (Fig.11).

Fig. 10 Creare tabel

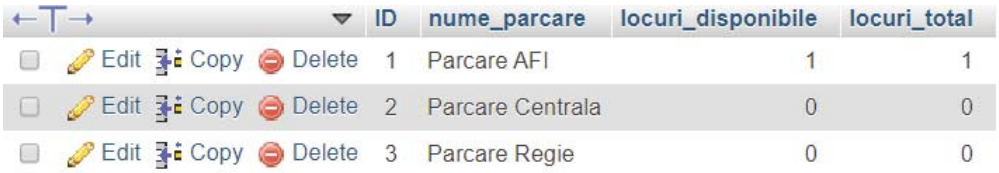


Fig. 11 Inserare date

În folderul “www” al aplicației WAMP se va adăuga un nou folder denumit “includes “ (Fig. 12) unde vor fi create trei fisiere de tip *.php (Fig. 13). Acestea vor ajuta la comunicarea permanentă între baza de date, server și programul Arduino.

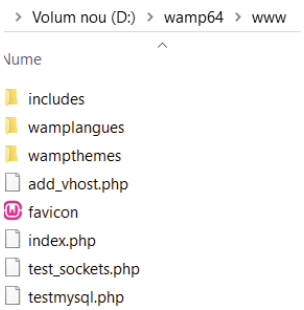


Fig. 12 Folderul “www”

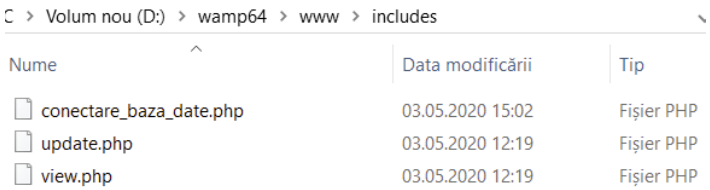


Fig. 13 Crearea fișierelor *.php

În figurile următoare se prezintă scrierea codurilor în Notepad++ pentru realizarea conectării la baza de date (Fig. 14), comunicării programului Arduino cu baza de date (Fig. 15) și codul pentru afișarea tabelului cu parcări (Fig. 16).

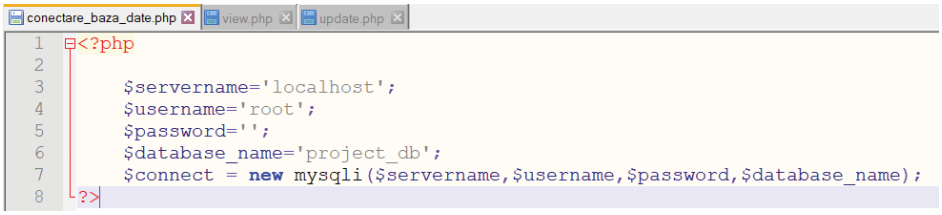


Fig. 14 Cod conectare bază de date

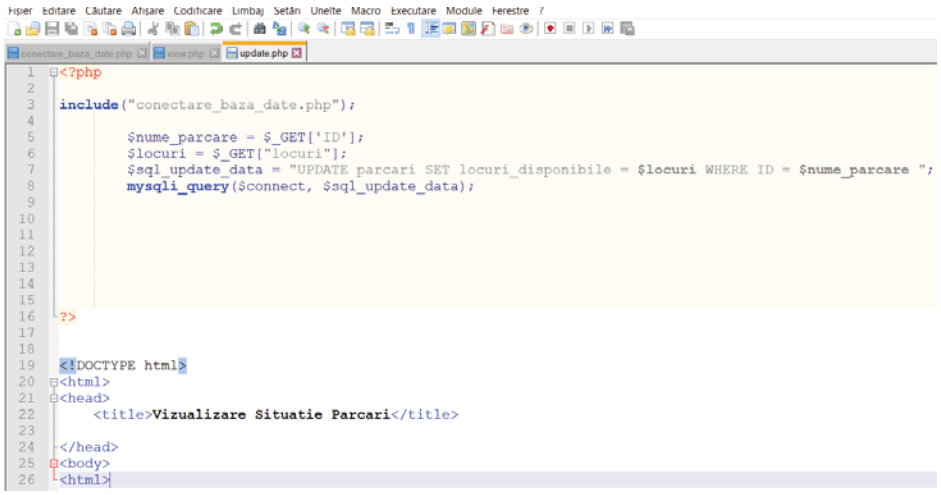


Fig. 15 Cod comunicare Arduino-baza de date

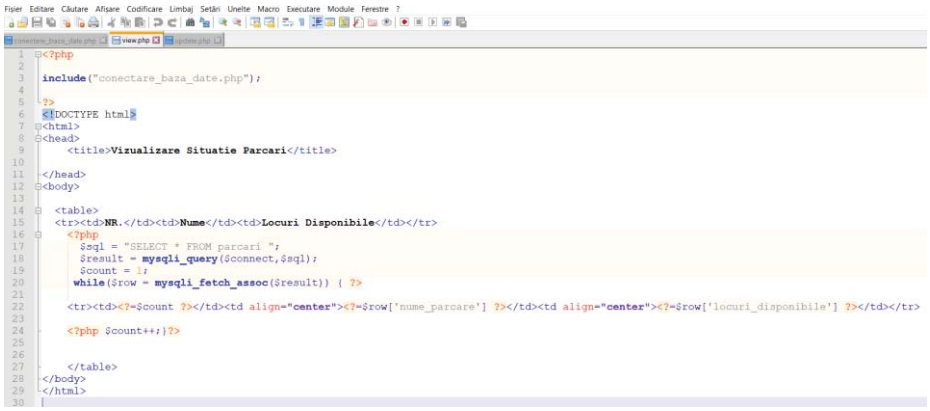


Fig. 16 Cod afișare tabel

Pentru ca placa Arduino UNO să fie conectată la internet și să comunice cu limbajul din Notepad++, programul Arduino trebuie să primească un update în scrierea codului. (Fig. 17).

Funcționarea sistemului realizat în regie proprie în mod practic :

Funcționarea întregului sistem, presupune conectarea componentelor la o sursă de alimentare, iar apoi se așteaptă o durată de timp până se realizează automat conectarea la rețeaua wireless. În momentul în care senzorul detectează autovehicolul la o distanță mai mică de 30cm (Fig. 18), LED-ul de avertizare se aprinde iar programul urmează să trimită date către server pentru a actualiza baza de date (Fig. 19).

Fig. 17 Update cod arduino

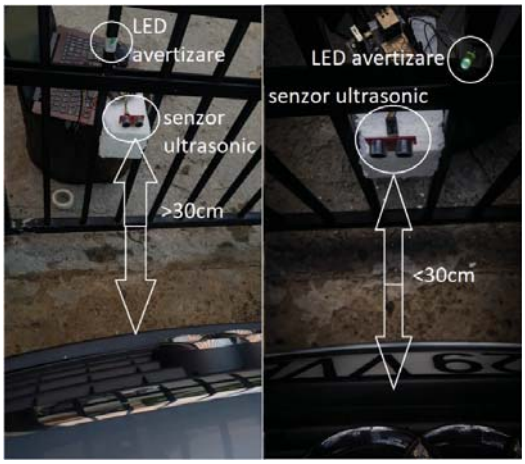
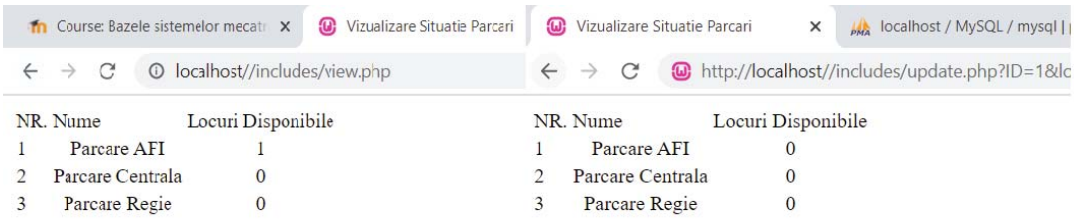


Fig. 18 Funcționare sistem



NR.	Nume	Locuri Disponibile
1	Parcare AFI	1
2	Parcare Centrala	0
3	Parcare Regie	0

NR.	Nume	Locuri Disponibile
1	Parcare AFI	0
2	Parcare Centrala	0
3	Parcare Regie	0

Fig. 19 Afișarea datelor online

4.Concluzii

În final s-a prezentat modul de construire și de testare a unui echipament conectat la o bază de date care servește la eficientizarea modului de localizare a unui loc de parcare într-o zonă dorită. Echipamentele au fost integrate într-o platformă virtuală care primind date și informații de la o placă Arduino UNO cu ajutorul unor senzori, s-a identificat prezența sau absența unui vehicul într-un loc de parcare.

Ca dezvoltări ulterioare pentru sistemul prezentat, se va realiza implementarea aplicației într-un server global si crearea unei aplicații mobile care să faciliteze rezervarea unui loc de parcare din orice zonă prin intermediul internetului, astfel eliminând pur și simplu ‘vânarea’ lor.

5.Bibliografie

[1]https://ftp.utcluj.ro/pub/users/peculea/CAN/Laboratoare/Introducere%20Arduino/Introducere%20Arduino.pdf accesat la data 27.04.2020
[2]https://sites.google.com/site/arduinoelectronicasiprogramare/arduino-si-senzori/1 accesat la data 28.04.2020
[3] http://localhost/phpmyadmin/
[4] https://www.tinkercad.com/dashboard?type=circuits&collection=designs
[5]Prof. Ileana Dugăeșescu, Bazele Sistemelor Mecatronice, Note de curs, 2019-2020.