

MINISTERUL EDUCAȚIEI



UNIVERSITATEA TEHNICĂ

DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

SMART HOME ÎN IoT

PROIECT DE DIPLOMĂ

Autor: **Sergiu Luntrașu**

Conducător științific: **șl.dr. ing. Dan Goța**

2023



UNIVERSITATEA TEHNICĂ

DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

Vizat,

DECAN

Prof. dr. ing. Liviu MICLEA

DIRECTOR DEPARTAMENT AUTOMATICĂ

Prof. dr. ing. Honoriu VĂLEAN

Autor: **Sergiu Luntrașu**

Smart Home în IoT

1. **Enunțul temei:** *Tema proiectului de diplomă presupune în elaborarea unei aplicații de automatizare a unei case inteligente. Proiectul va pune în evidență funcționalități generale în IoT precum realizarea unui sistem de iluminat automat, unui sistem în caz de incendiu sau verificarea valorilor temperaturii, umidității sau a gazului într-un anumit loc. Toate acestea vor fi complet controlabile dintr-o aplicație mobilă, putând totodată să și notifice utilizatorul în anumite circumstanțe*
2. **Conținutul proiectului:** *Pagina de prezentare, Declarație privind autenticitatea proiectului, Sinteza proiectului, Cuprins, Introducere, Studiu bibliografic, Interfața mobilă și baza de date, Partea hardware și crearea server-ului, Testare și validare, Concluzii, Bibliografie.*
3. **Locul documentării:** *Universitatea Tehnică din Cluj-Napoca, Internetul*
4. **Consultanți:**
5. **Data emiterii temei:** 26.10.2022
6. **Data predării:** 03.09.2023

Semnătura autorului

MINISTERUL EDUCAȚIEI



UNIVERSITATEA TEHNICĂ

DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

Semnătura conducătorului științific

**UNIVERSITATEA TEHNICĂ**

DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

**Declarație pe proprie răspundere privind
autenticitatea proiectului de diplomă**

Subsemnatul(a) **Sergiu Luntrașu**,
legitimat(ă) cu CI seria XV nr. 596312 , CNP_
5000821330200,

autorul lucrării:

Smart Home în IoT

elaborată în vederea susținerii examenului de finalizare a studiilor de licență la **Facultatea de Automatică și Calculatoare**, specializarea **Automatică și Informatică Aplicată**, din cadrul Universității Tehnice din Cluj-Napoca, sesiunea Septembrie 2023 a anului universitar 2022-2023, declar pe proprie răspundere, că această lucrare este rezultatul propriei activități intelectuale, pe baza cercetărilor mele și pe baza informațiilor obținute din surse care au fost citate, în textul lucrării, și în bibliografie.

Declar, că această lucrare nu conține porțiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislației române și a convențiilor internaționale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în fața unei alte comisii de examen de licență.

În cazul constatării ulterioare a unor declarații false, voi suporta sancțiunile administrative, respectiv, *anularea examenului de licență*.

Data

03.09.2023

Sergiu Luntrașu

(semnătura)



SINTEZA

proiectului de diplomă cu titlul:

Smart Home în IoT

Autor: **Sergiu Luntrașu**

Conducător științific: **șl.dr. ing. Dan Goța**

1. Cerințele temei: Un sistem de iluminat , un sistem în caz de incendiu cu notificare a utilizatorului , monitorizarea temperaturii, umidității și a gazului dintr-o arie specifică , controlarea absolută de pe o aplicație mobile cu recepționarea datelor în timp real
2. Soluții alese: Crearea unei interfețe mobile (Blynk), a unui server în limbaj PHP (Visual Code) unde să remită datele într-o bază de date (phpMyAdmin), microcontroller Nodemcu ESP8266 , senzori IoT și alte componente (led-uri, buzzere, rezistențe)
3. Rezultate obținute: Datele transmise de la senzori putând fii observate în aplicația de pe mobil, notificând user-ul în anumite circumstanțe primejdioase totodată, observându-se și evoluția datelor în timp real
4. Testări și verificări: Testarea microcontroller-ului ESP8266 , testarea senzorilor fiind montați corespunzător , testarea aplicație recepționând informațiile transmise de plăcuță, server-ul reușind să facă legătura cu baza de date ajutându-ne de Postman
5. Contribuții personale: Implementarea părții hardware și software de asemenea, putând face posibila conexiunea acestora printr-o aplicație mobile, realizarea unui sistem de notificare a user-ului.
6. Surse de documentare: Articole științifice , cursuri cumpărate , tutoriale video, vezi și bibliografia

Semnătura autorului

Semnătura conducătorului științific

Cuprins

1	INTRODUCERE.....	2
1.1	CONTEXT GENERAL	2
2	STUDIU BIBLIOGRAFIC.....	3
2.1	SMART HOME ȘI IOT.....	3
2.1.1	Protocoale Smart Home.....	5
2.1.2	Aplicații Smart Home.....	6
2.1.3	Design Smart Home	7
2.2	NODEMCU ESP8266.....	7
2.3	SENZORI	8
2.4	LIMBAJUL PHP	12
2.4.1	Analiza PHP în Rascal.....	12
2.4.2	Metoda Copy-on-write.....	14
3	INTERFAȚA MOBILE ȘI BAZA DE DATE	15
3.1	CREAREA INTERFEȚEI MOBILE	15
3.1.1	Blynk introducere	16
3.1.2	Crearea device-ului	16
3.1.3	Adăugare widgets.....	20
3.1.4	Adăugare notificări	21
3.2	CREAREA BAZEI DE DATE.....	23
3.2.1	phpMyAdmin introducere.....	24
3.2.2	Crearea tabelelor	25
4	PARTEA HARDWARE ȘI CREAREA SERVER-ULUI.....	25
4.1	PARTEA HARDWARE	25
4.1.1	Total componente utilizate	26
4.1.2	Schema electrică.....	29
4.1.3	Arduino IDE introducere.....	29
4.1.4	Legătură între Arduino IDE și Blynk	30
4.1.4.1	Librării Blynk	30
4.1.4.2	Conectare device	30
4.2	CREAREA SERVER-ULUI.....	31
4.2.1	Visual Studio Code introducere	32
4.2.2	Scrierea server-ului	32
4.2.3	Evaluarea funcționalităților	36
5	TESTARE ȘI VALIDARE	39
6	CONCLUZII.....	43
6.1	REZULTATE OBTINUTE ȘI PĂRERI FINALE	43
6.2	DIRECȚII DE DEZVOLTARE.....	44
7	REFERENCES.....	45

1 Introducere

1.1 Context general

IoT este cunoscut și sub numele de Internetul lucrurilor. Este modul de conectare a obiectelor fizice prin internet la alte dispozitive. Termenul internet al lucrurilor a fost dat de Kevin Ashton și el a menționat prima dată în anul 1999. Internetul lucrurilor înseamnă momentul în care lucrurile sau obiectele sunt mai mult conectate la Internet decât în oameni. Lucrurile din IoT s-au definit ca obiecte care pot fi persoana sau automobilul cu un senzor încorporat având adrese IP cu capacitatea de a colecta și transfera datele pe Internet[4].Descrieți importanța lucrării, de ce merita să o faceți, plasați ideile într-un context larg.[2]

Ca o componentă importantă a Internetului lucrurilor (IoT), casele inteligente (Smart Homes) servesc eficient utilizatorilor prin comunicarea cu diverse dispozitive digitale bazate pe IoT. În versiunea ideală a unui viitor cu fir, toate dispozitivele din casele inteligente comunică între ele fără probleme. Tehnologia smart home bazată pe IoT a schimbat viața umană, oferind conectivitate tuturor, indiferent de timp și loc. Sistemele de automatizare a locuinței au devenit din ce în ce mai sofisticate în ultimii ani. Aceste sisteme oferă infrastructură și de asemenea, metode de schimb de informații și servicii pentru toate tipurile de aparate. O casă inteligentă este un domeniu al IoT-ului, care simbolizează o rețea de dispozitive fizice care oferă conectivitate electronică, cu senzori implementați, o parte software și o rețea în interiorul unei case.[1]

Din 2010, cercetătorii au analizat aplicațiile de casă inteligentă bazate pe IoT folosind mai multe abordări. Indiferent de categoria lor, articolele de cercetare existente se concentrează pe provocările care împiedică utilizarea deplină a aplicațiilor IoT pentru casă inteligentă și oferă recomandări pentru a atenua aceste probleme. Cercetarea aplicațiilor pentru casă inteligentă este dinamică și diversă. În continuare, ne propunem să ecranizăm perspective valoroase asupra mediilor tehnologice și să valorificăm sprijinul cercetătorilor prin înțelegerea opțiunilor și a lacunelor disponibile în această linie de cercetare. Să facem lumină asupra eforturilor cercetătorilor ca răspuns la tehnologiile noi și perturbatoare, să cartografieze peisajul cercetării într-o taxonomie coerentă și să determinăm trăsăturile care caracterizează această linie emergentă de cercetare în tehnologia casei inteligente.

IoT creează o lume în care toate obiectele (numite și obiectele inteligente) din jurul nostru sunt conectate la internet și comunica între ei cu minimul de intervenție uman. Scopul final este de a crea „o lume mai bună pentru ființe umane, unde obiectele din jurul

nostru știi ceea ce noi ce ne dorim și ce avem nevoie și acționăm în consecință fără instrucțiuni explicite.[18]

Cercetările actuale despre internetul obiectelor (IoT) în principal se concentrează pe modul de a permite obiectelor generale să vadă, să audă și să miros lumea fizică pentru ei înșiși și le face conectat pentru a împărtăși observațiile. În acest sens, monitorizarea și luarea deciziilor pot fi mutate din partea umană spre partea mașinii.[19]

Întrucât IoT este considerat conexiunea în rețea a obiecte sau dispozitive fizice. Una dintre definițiile IoT de către cercetătorul este [3], o rețea deschisă și cuprinzătoare de obiecte inteligente care au capacitatea de a se autoorganiza, împărtășește informații, date și resurse, reacționând și acționând în fața situațiilor și schimbărilor din mediu.[20]

Profitând de ajutorul tehnologiilor de comunicare precum rețele de senzori fără fir (WSN) și frecvența radio de identificare (RFID)[21], are loc un schimb de informații [4]. Deci, în general, putem spune că IoT permite oamenilor și lucrurilor să fie conectat oricând, oriunde, cu orice și oricine folosind orice rețea și orice serviciu.[14]

2 Studiu bibliografic

În acest capitol vom ecraniza studiile bibliografice despre casele inteligente și ce presupune în general IoT, microcontroler-ul ESP8266, cum este formatat și în ce scop este folosit și senzorii pe care îi vom folosi.

2.1 Smart Home și IoT

Automatizarea locuinței este unul dintre cele mai comune IoT aplicații. Deoarece îmbunătățește și viața oamenilor, mulți lucrările de cercetare au abordat această idee. Din sursă deschisă standarde, la protocoale și aplicații proprietare, există diverse implementări care sprijină conceptul de casă inteligentă.

Casele inteligente au atras un interes din ce în ce mai mare din punctul de vedere atât al construcției eficiente din punct de vedere energetic, cât și al locuinței confortabile. Disponibilitatea unui număr de TIC și tehnologii energetice le permite rezidenților să integreze convenabil tehnologii și servicii în casele lor. O casă inteligentă este „un mediu rezidențial cu tehnologii de informare și comunicare, care oferă funcții adecvate pentru nevoile rezidenților de confort, securitate, divertisment și confort[22]. Conceptul principal al unei case inteligente este folosirea unor funcții de management bine concepute, care oferă rezidenților posibilitatea de a controla în mod optim întreținerea caselor lor.

Integrarea tehnologiilor TIC și energetice recente și moderne în contextul locuințelor oferă un potențial puternic pentru noi servicii și aplicații (Shapsough și Zualkernan, 2019). Printre astfel de tehnologii de calcul omniprezente, care permit tuturor obiectelor să fie conectate oriunde și în orice moment, folosesc senzori minusculi pentru a furniza informații întregi pentru a face utilizatorii să facă parte dintr-o rețea complet conectată. Aceasta înseamnă că utilizatorii fac parte dintr-un sistem de mașini de calcul interconectate pentru trimiterea și primirea de date printr-o rețea fără forme tradiționale de comunicare. Acesta este conceptul principal din spatele IoT. Pe baza mediului IoT, utilizatorii pot interacționa cu diverse dispozitive și servicii din casele inteligente (Neagu et al., 2017).

Wang și colab. (2021) au conceptualizat domeniile IoT și au extras rețelele de senzori fără fir și securitatea/confidențialitatea IoT ca principalele curente ale IoT. Apoi, ei au indicat că apariția și aplicațiile IoT au fost amplificate în mod constant în medii inteligente (de exemplu, oraș inteligent și casă inteligentă).[24]

Singh și colab. (2020) a oferit o privire de ansamblu asupra subiectelor apărute despre IoT și agricultură. Pe baza constatărilor unei analize bibliometrice, impactul IoT asupra cercetării agricole a crescut enorm. Mai mult, ei au indicat că utilizarea tehnologiilor și aplicațiilor IoT le permite cercetătorilor agricoli să aibă o oportunitate de a explora noi domenii de cercetare.

Deoarece IoT a afectat și a remodelat în mod constant societatea noastră, cum ar fi zonele urbane, agricultura, transportul, mediile de locuințe și sistemele de sănătate, utilizarea IoT în zonele de casă inteligentă stabilește un domeniu unic de casă inteligentă cu IoT (SHIoT).

Casele inteligente și IoT ne-au afectat în mod constant viețile prin intermediul serviciilor care sunt conectate oriunde și în orice moment. Având în vedere această tendință, câțiva cercetători au susținut că o casă inteligentă, care este definită ca „rețeaua de dispozitive fizice care oferă conectivitate electronică, senzori, software și rețea în interiorul unei case”, face parte din domeniul IoT deoarece integrează sistemele de automatizare a locuinței cu bazele principiilor IoT. Sistemele de tip casă inteligentă (Smart Home) constau dintr-un număr mare de senzori și comutatoare care sunt conectate la poarta de acces principală, care este principalul sistem de control pe care îl accesează utilizatorii prin intermediul dispozitivelor lor digitale, cum ar fi smartphone-urile sau computerele personale desktop.[13]

Khalaf și colab. (2019) au examinat abordări notabile de aplicare a conceptului IoT la aplicațiile pentru casă inteligentă. Acest lucru implică faptul că noile piețe emergente și domenii de cercetare, cum ar fi SHIoT, sunt prezentate ca domenii de cercetare viitoare. Mai multe studii au investigat utilizarea serviciilor și produselor IoT în medii de casă inteligentă și eliminarea problemelor tehnice și aplicabile conexe care împiedică difuzarea SHIoT și au luat în considerare metode pentru a îmbunătăți atât eficiența, cât și fezabilitatea implementării SHIoT (Park et al., 2017).

Deci putem spune că studiul actual își propune să prezinte perspective notabile asupra viitoarelor domenii de cercetare prin înțelegerea stării actuale și a mediilor de cercetare. Această abordare poate pune în lumină contribuțiile și eforturile cercetătorilor din domeniile SHIoT, să se poată oferi o imagine de ansamblu asupra tendințelor coerente de cercetare și caracteristicile și direcțiile emergente ale SHIoT.

2.1.1 Protocoale Smart Home

Open Interconnect Consortium (OIC) a încercat să creeze un proiect universal open source numit IoTivity [7] (un proiect de software cu sursă deschisă care permite o conectivitate fără întreruperi de la dispozitiv la dispozitiv, în care miliarde de dispozitive Internet of Things (IoT) cu fir și fără fir se pot conecta în siguranță între ele și la internet). Alte tehnologii pot fi ușor integrate în acest proiect folosind plugin-uri. Ca rezultat, multe tipuri de dispozitive cu diferite metode de comunicare pot interacționa între ele prin IoTivity. Obiectivul designului este de a crea un standard utilizat de dispozitive fără fir și cu fir pentru a comunica prin intermediul internetul. Acest lucru poate facilita dezvoltarea unei case inteligente proiect. IoTivity încearcă să se asigure că fiecare dispozitiv nou inteligent, se poate conecta la ecosistemul IoT. Deoarece este o sursă deschisă de tip proiect, încurajează dezvoltatorii să contribuie și să extindă cadrul cu informații necesare pentru conectarea tuturor tipurilor de dispozitive în profilul corespunzător. Cadrul API IoTivity se bazează pe un model de arhitectură RESTful [7]. Conține patru blocuri care fac comunicare posibilă:

- Descoperire – acceptă mai multe metode pentru dispozitiv descoperire, atât în proximitate, cât și la distanță;
- Transmiterea datelor – comunicare bazată pe model de mesagerie și streaming;
- Managementul datelor – stocare și calcul sau date;
- Managementul dispozitivelor – configurarea dispozitivelor.

Prin urmare, cu acest API mai multe tehnologii pentru pot fi incorporate informații de transport. De exemplu se pot baza pe Bluetooth, Wi-Fi sau comunicații în cloud.

Există, de asemenea, protocoale care permit dispozitivelor inteligente să comunice între ele și prin Internet. Exemple în această zonă includ ZigBee și Z-Wave [11]. Acestea sunt tehnologii wireless concepute pentru control de la distanță. Deși ideea este asemănătoare, diferă prin frecvența pe care o folosesc, gama de rata de transmisie și de date. ZigBee este un standard wireless deschis bazat pe IEEE Standarde radio de rețea 802.15.4. Diferența dintre două standarde este că, în timp ce ultimul oferă Stratul 1 și 2 comunicare (nivelul fizic și controlul accesului media), celălalt se adresează straturilor de rețea și aplicații [8]. Prin urmare, ZigBee oferă o stivă de software pentru niveluri superioare. Poate fi folosit fie pentru conexiuni directe între dispozitive sau în rețele bazate pe o topologie de pornire sau ochiuri arborescente. În aceste scenarii una dintre entități acționează ca coordonator și dictează comunicarea dintre celelalte noduri. Dacă

două noduri nu pot comunica direct, atunci pot trece informații prin alte entități până ajunge la destinație. Intervalul maxim dintre două noduri este de 10 metri.

Principalul avantaj al acestui standard este flexibilitatea acestuia. Software-ul specific este dezvoltat pentru a funcționa cu ZigBee. Aceasta este cunoscută ca profil și poate fi integrat de dezvoltatori în produsele lor. Stack ZigBee include deja o casă profil de automatizare. Spre deosebire de ZigBee, Z-Wave nu este un standard deschis. Este doar disponibil pentru clienții Zensys/Sigma Design [8]. Conexiunea dintre noduri se bazează pe aceeași idee, dar folosește game diferite. Deși ZigBee acceptă mai multe noduri (65000 vs 232), Z-Wave are o gamă mai mare de comunicare între acestea de până la 30 de metri. O altă diferență este că Z-Wave este un wireless proprietar standard care funcționează la accesul fizic și media straturi de control. Acest lucru se realizează prin încorporarea unui modul fizic în produse. Z-Wave este folosit în principal pentru monitorizarea și controlul funcțiilor de acasă, cum ar fi lumini, încuietori, senzori de temperatură sau de securitate. ZigBee și Z-Wave au aceeași aplicație țintă IoT. Între aceste două soluții, ZigBee este cea preferată de dezvoltator. Profilurile disponibile minimizează produsul timpului de dezvoltare. Dezavantajul ambelor protocoale este că dezvoltatorii au nevoie de un număr mare de dispozitive pentru a deveni buni acoperire, deoarece intervalele lor sunt scăzute.

2.1.2 Aplicații Smart Home

Un exemplu de sistem de casă inteligentă este Apple HomeKit [7]. Folosind aceasta, dispozitivele de acasă pot fi controlate de a smartphone care rulează iOS. Nu este nevoie de un dispozitiv suplimentar care acționează ca gateway și datele transferate sunt criptate.

HomeKit beneficiază de peste 50 de marci de produse cu software integrat [7]. Unele dintre ele se găsesc în domenii precum fulgere, încuietori, întrerupătoare, controale de temperatură și chiar nuanțe. După ce le-ai activat pe smartphone folosind software-ul corespunzător din App Store, utilizatorul poate controla le și automatizează comportamentul lor. Ei pot pune scene și chiar dați comenzi vocale directe folosind Siri.

În acest domeniu al produselor pentru casă inteligentă, un procent semnificativ este reprezentat de gadgeturi de securitate. Danalock Bluetooth Z-Wave Smart Lock [8] este un dispozitiv de blocare a ușii care poate fi instalat cu ușurință în orificiul șurubului. Cu asta, ușa poate fi deschisă de pe un smartphone, care funcționează fie cu Android, fie Dispozitive compatibile cu iOS. Are și posibilitatea de a acorda permisiunea altor persoane să descuie ușa. O altă soluția de blocare este oferită de Oplink Connected CMPOPG2204OPL01 [9][12].

Samsung oferă un kit care combină mai multe funcții pentru control și monitorizare acasă. Se numește SmartThings Home Kit de monitorizare [10] și include gadget-uri tradiționale monitoare de acasă la aparate care pot fi controlate de la distanță de pe un

dispozitiv mobil. Prin urmare, are camere, fum, monitoare și chiar sisteme de control al luminilor.

Au fost dezvoltate multe alte dispozitive de securitate și majoritatea folosesc camere pentru monitorizarea acasă, cum ar fi Dropcam Cameră de monitorizare video wireless Pro WiFi, Belkin NetCam Cameră IP fără fir HD sau D-Link – Camera cloud 5000 [11].

2.1.3 Design Smart Home

La baza design-ului pentru acest sistem de casă inteligentă se află ideea de a face un produs ușor de utilizat care să integreze mai multe tipuri de sarcini în interiorul unei case. Are o interfață simplă cu câțiva pași de configurare. Utilizatorul se conectează la cloud, trimite comenzile dorite către gateway-ul smart home de la casa și componenta fizică corespunzătoare aceasta activat.

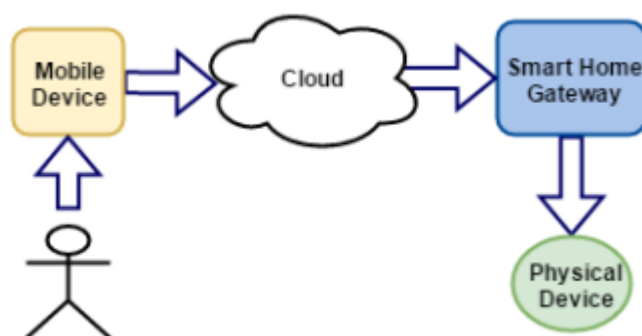


Fig. 2.1 Schemă Design

Interacțiunea dintre utilizator și cloud este realizată folosind un browser de internet care poate fi instalat pe orice mobil dispozitiv precum smartphone-uri, tablete sau laptop. Utilizatorul poate alege ce tip de dispozitive preferă pe partea clientului.

2.2 Nodemcu ESP8266

NodeMCU este o platformă IoT open source care rulează SoC Wi-Fi ESP8266 și hardware-ul se bazează pe ESP-12 modul. Se referă mai degrabă la firmware decât la dezvoltarea kituri-lor care oferă acces la aceste GPIO-uri ale ESP8266 și acesta este utilizat pe scară largă în diverse aplicații IoT. Oferă acces la GPIO (General Purpose Input/Output) al modulului și poate fi fie pinul de intrare, fie pinul de ieșire, al cărui comportamentul poate fi controlat în momentul alergării.

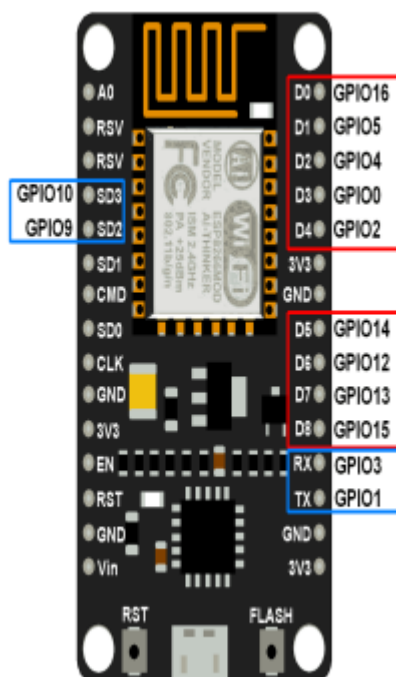


Fig. 2.2 Notățiile GPIO ale ESP8266

Cipul de procesor are 16 linii GPIO, dintre care unele sunt folosite pentru a interfața cu alte componente ale SoC-ului, cum ar fi memoria flash. Au mai rămas aproximativ 11 pini GPIO pentru folosit în scopul GPIO. Doi pini din 11 GPIO sunt rezervat în general pentru RX (receptor) și TX (transmițător) pentru a comunica cu un PC gazdă de pe care a fost descărcat codul obiect compilat. Acest modul se încarcă și informațiile sunt transferate de la gazdă la bord prin cablul USB.



Fig. 2.2.1 Microcontroler NodeMCU ESP8266 cu modul Wi-Fi

2.3 Senzori

Senzorii joacă un rol important în automatizarea oricăruia aplicare prin măsurarea și prelucrarea datelor colectate pentru detectarea schimbărilor în lucrurile fizice.[5] Ori

de câte ori există o modificare a oricărei stări fizice pentru care este realizat un senzor, produce un răspuns măsurabil. Fig. următoare arată detectarea elemente și semnalul electric corespunzător.[23]

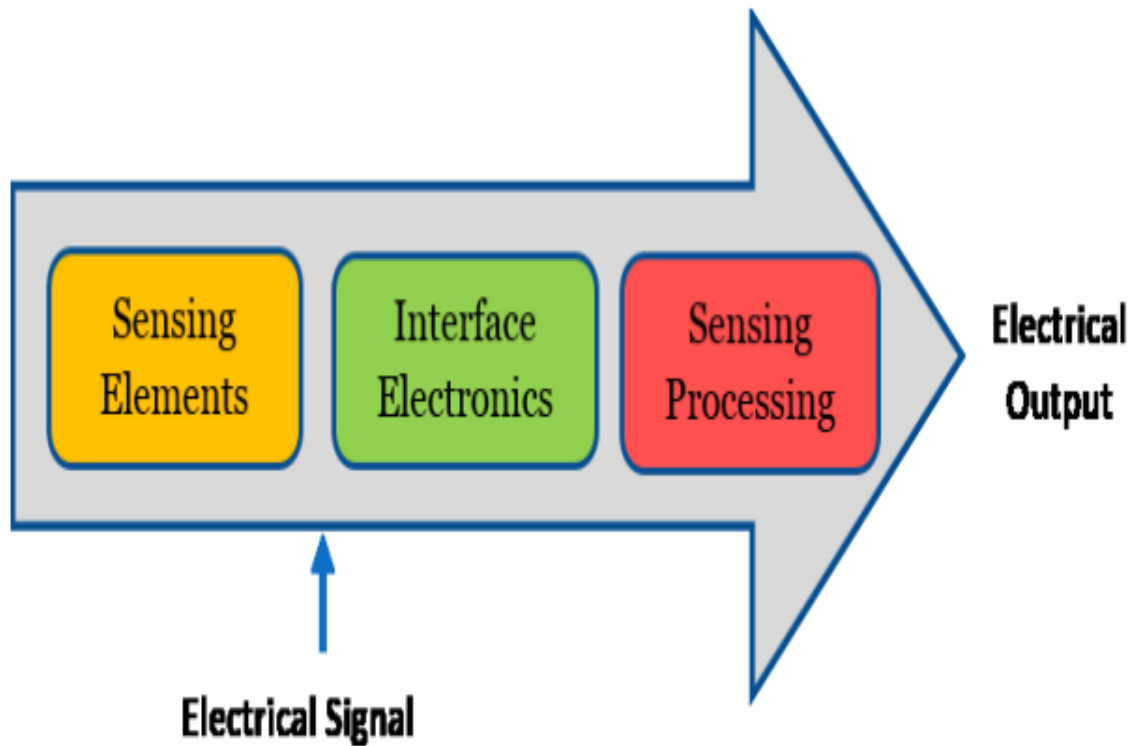


Fig. 2.3.1 Elemente de detectare și semnal electric corespunzător

Există diferite tipuri de senzori care pot varia de la foarte simplu spre complex. Clasificarea senzorilor poate fi pe baza specificațiilor lor, metoda de conversie a acestora, tipul de materialul utilizat, fenomenul fizic de detectare, proprietăți pentru ceea ce măsoară și domeniul de aplicare. Fig. următoare prezintă diverse tipuri de senzori în IoT, care sunt explicate mai jos.[25]

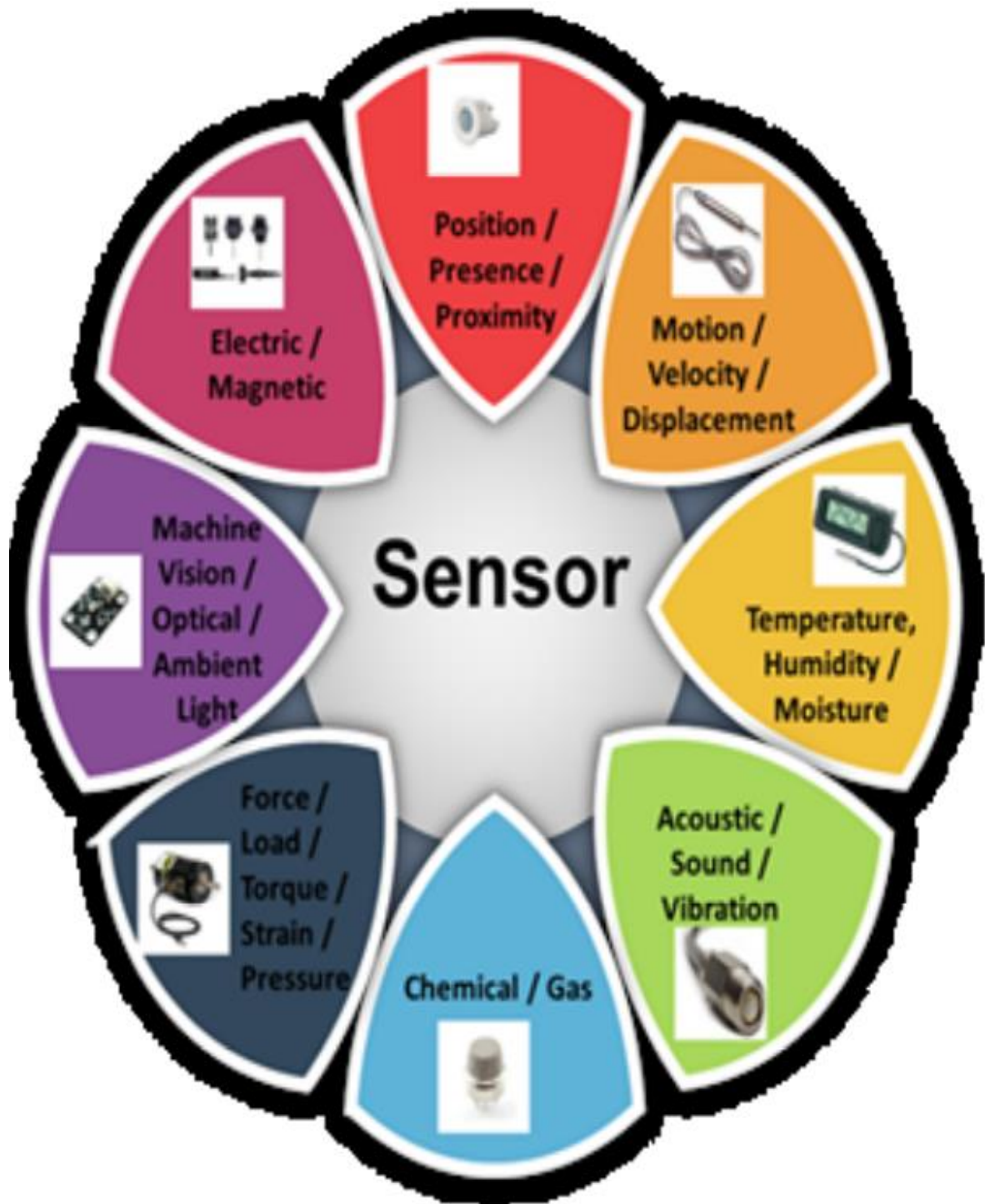


Fig. 2.3.2 Diferite tipuri de senzori IoT

Am ales senzorii de temperatura, umiditate , foc si gaz pentru monitorizarea mediului și totodată am ales utilizarea unui singur senzor pentru detecția umidității și a temperaturii. Deci, am ales cipul cu senzor compozit DHT11 și care oferă citiri atât pentru umiditate cât și pentru temperatură, toate împreună, are mare fiabilitate și stabilitate excelentă pe termen lung și are trei terminalele VCC, DATA și GND. Are dimensiuni mici,

cost scăzut. De asemenea, are o calitate bună, capacitate anti-interfață puternică, răspuns rapid cu o calibrare precisă. [16]



Fig. 2.3.3 DHT11

Un detector de flacără, senzor conceput pentru a detecta și a răspunde la prezența unei flăcări sau a unui incendiu. Răspunsurile la o flacără detectată depind de instalație, dar pot include sunetul unei alarme, dezactivarea unei conducte de combustibil (cum ar fi o conductă de propan sau gaz natural) și activarea unui sistem de stingere a incendiilor. Senzorul de flacără IR este utilizat în acest proiect și este prezentat mai jos, acești senzori mai sunt denumiți, uneori, modul senzor de incendiu sau senzor detector de flacără.[6]

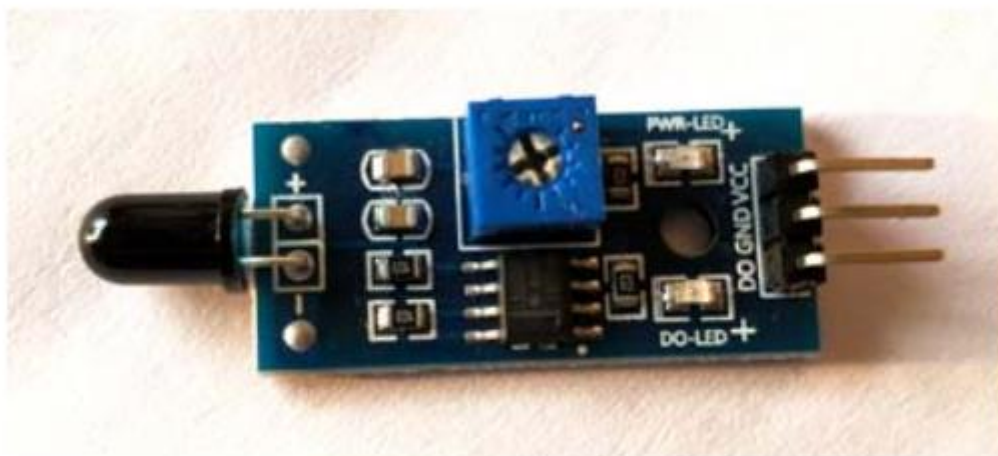


Fig. 2.3.4 Senzor de flacără IR

Iar senzorul de gaz MQ2 a fost selectat pentru monitorizarea nivelului de gaz în mediu. Senzorul de gaz MQ2 funcționează la 5V DC și consumă aproximativ 800mW. Poate detecta concentrații de GPL, fum, alcool, propan, hidrogen, metan și monoxid de carbon cuprinse între 200 și 10000 ppm. Acest senzor are o sensibilitate ridicată și un timp de răspuns rapid [17].



Fig. 2.3.5 MQ2

2.4 Limbajul PHP

PHP este un limbaj popular pentru aplicațiile server-side. În PHP, atribuirea variabilelor copiază valorile alocate, conform acestuia așa-numita semantică Copy-on-assignment. În schimb, un PHP tipic implementarea folosește o schemă de Copy-on-write pentru a reduce copia overhead prin întârzierea copiilor cât mai mult posibil.

2.4.1 Analiza PHP în Rascal

Cadrul PHP Air este alcătuit dintr-un număr de instrumente și biblioteci reutilizabile construite pentru a sprijini cercetarea în domeniul empiric, ingineria software, evoluție software, analiză de programe, și metrice software pentru sistemele construite folosind PHP. În figura 2.4.1 arată o prezentare generală a PHP Air. Acesta în sine este construit folosind o combinație de PHP, Java și meta-programarea Rascal [15]; este indicată limba folosită pentru orice pas dat în fiecare cutie. Cutiile cu colțuri rotunjite reprezintă unelte furnizate de sistem sau sarcini create de utilizatori (de exemplu, personalizate interogări pe un sistem PHP), în timp ce casete cu colțuri pătrate reprezintă artefacte, cum ar fi

sistemul original PHP, abstract arbori de sintaxă (AST) pentru fișierele PHP sau rezultatele rulării PHP AiR, cum ar fi tabele, fișiere cu rezultatele analizei, vizualizări, sau alte documente.

Începând cu un sistem PHP, format din unul sau mai multe PHP fișiere sursă, PHP Air creează AST-uri pentru fiecare dintre fișiere. Aceste AST-uri sunt o structura de bază folosită de PHP Air pentru motiv despre codul PHP și sunt folosite împreună cu informațiile calculate folosind Rascal (de exemplu, grafice de apeluri, informații din bibliotecă extrase din documentația PHP) precum și informațiile limitate (de exemplu, linii de informații de cod pentru fișierele din sistemele, versiunea PHP minimă necesară și data lansării a fiecărui sistem) furnizate de instrumente externe în fișiere CSV și fiind citite folosind funcția de resurse Rascal [8]. Aceste AST-uri sunt organizate în sisteme, unde un singur sistem reprezintă toate fișierele și AST-urile asociate pentru un anumit produs și versiune.

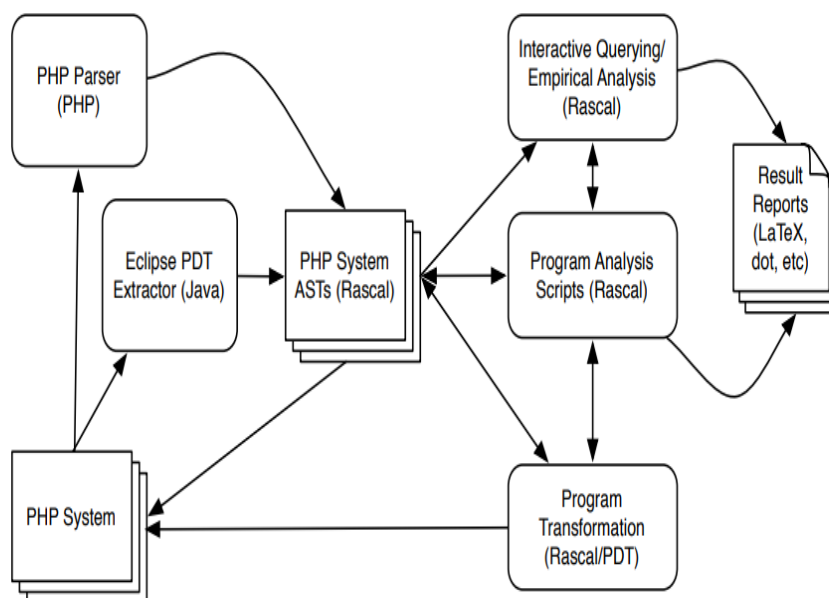


Fig. 2.4 Prezentare la nivel înalt PHP Air

PHP AiR folosește două analizoare diferite, și în funcție de scenariul de utilizare dorit necesită viteză sau precizie. Primul este un analizor extern pentru PHP, scris tot în PHP, care se bazează pe analizatorul folosit în interiorul PHP însuși. Acest analizator poate fi folosit pentru a analiza fișiere individuale, dar este în marea majoritate folosit pentru procesarea în lot a unui număr mare de fișiere PHP, de exemplu, toate fișierele sursă în toate versiunile WordPress. Analiza returnează fie a un singur AST, un singur sistem sau mai multe sisteme, în funcție de intrare. Al doilea se bazează pe analizorul folosit de PHP Development Tools (PDT) în Eclipse. Când fișierele sunt analizate de către PDT, un document-obiect intern pentru fiecare fișier, similar cu un AST, este creat. Acest DOM este

apoi traversat de Eclipse PDT Extractor pentru a extrage AST-urile identice cu acele date de pe primul analizor pentru fiecare fișier. Extractorul PDT este destinat utilizării interactive în cadrul Eclipse și necesită toate fișierele sursă să facă parte dintr-un proiect Eclipse care este deschis în același spațiu de lucru. Fie fișiere individuale sau sistemele individuale (unul pentru fiecare proiect) sunt extrase.

Folosind aceste AST-uri și sisteme, PHP AiR oferă un număr de funcții Rascal și tipuri de date care vizează interogarea codului, analiza empirică a software-ului, evoluția software-ului, analiza programului și transformarea programului. Acestea sunt ambalate ca biblioteci Rascal, permițându-le să fie reutilizate în alt cod Rascal pentru a susține sarcini specifice de analiză sau investigații empirice. Deoarece sistemele și AST-urile sunt reprezentate ca tipuri Rascal, PHP AiR poate analiza AST-uri individuale, un singur sistem sau chiar mai multe sisteme în același timp, de exemplu pentru a compara cum sunt utilizate funcțiile în diferite produse (de exemplu, MediaWiki față de WordPress) sau pentru a vedea cum se utilizează funcțiile specifice s-a schimbat în diferite versiuni ale aceluiași produs.

Câteva exemple detaliate de instrumente existente care au fost create folosind PHP AiR ar fi: Analiza empirică a utilizării caracteristicilor PHP, Transformarea caracteristicilor dinamice în caracteristici statice, etc. ; alte exemplele includ o analiză de contaminare pentru detectarea periculoaselor utilizări de șiruri de caractere neverificate furnizate de utilizator în apelurile bibliotecii PHP, o refactorizare de la HTML, codificat manual la utilizări ale șablonului bibliotecii, o refactorizare similară din apelurile SQL, codificate manual la utilizările altor biblioteci, cu baze de date și mai multe proiecte de extras diverse valori din codul sursă PHP.

Odată cu extragerea de informații din sistemele PHP și modificând AST-urile, codul PHP transformat poate fi, de asemenea generat. Acest lucru se poate face într-unul din două moduri. În primul rând, PHP AST-urile AiR pot fi destul de imprimate pentru ași genera fundația reprezentării; limitarea este că aceasta nu se păstrează în spații albe, inclusiv, așa că acest lucru este cel mai bine în situații unde codul specific trebuie injectat în anumite locații a unui fișier existent sau de unde este creat un fișier nou de la zero. În al doilea rând, PHP AiR interfațează cu PDT-ul pentru a permite declarațiile sau expresiile existente care urmează să fie înlocuite cu noi instrucțiuni sau expresii, permițând injectarea unui cod arbitrar în locații specifice din fișierele PHP. Acest lucru este limitat la fișierele existente care trebuie să facă parte dintr-un proiect PHP în Eclipse.

2.4.2 Metoda Copy-on-write

Copy-on-write este o tehnică clasică de optimizare, bazată pe ideea de a întârzia copierea până când există o scriere a datelor. Denumirea tehnicii provine din copia datelor originale fiind forțat de momentul scrierii. Un exemplu de Copy-on-write se găsește în UNIX fork, unde memoria procesată local corespunde datelor locale, care ar trebui copiate din spațiul de adresă al procesului original în spațiul noului proces prin operarea UNIX fork. În sistemele moderne UNIX, această copie este de obicei întârziată prin copiere la scriere.

Un alt exemplu se găsește în limbajul PHP, un limbaj de scripting popular pentru aplicațiile Web de pe partea de server. Iată un exemplu cu tablourile asociative ale PHP

```
$r["box"] = "gizmo";  
$l = $r; // assignment from $r to $l  
$l["box"] = "gremlin";  
echo $r["box"]; // prints out gizmo
```

Fig. 2.4.1 Atribuirea variabilelor prin referință în PHP

Numai modificarea lui \$l la Linia 3, în urma atribuirii \$l = \$r are efecte locale asupra \$l care nu pot fi văzute de la \$r. Comportamentul sau semantica în PHP se numește Copy-on-assignment, deoarece valoarea de \$r pare a fi copiat înainte de a fi trecut la \$l. Putem lua în considerare tehnica Copy-on-write pentru a implementa acest comportament. Într-adevăr, cel mai dominant runtime PHP, numită Zend runtime, angajează copiere la scriere și amână copia de mai sus până la scrierea la linia 3. Pentru cititorii din comunitatea de limbi funcționale sau declarative, semantica matricelor PHP poate suna mai familiar, de exemplu, matricele PHP sunt similare cu matricele funcționale. în orice caz asemănarea lor devine mai puțin clară pe măsură ce aflăm cum putem împărtăși valori în PHP. În PHP, avem instrucțiunea de atribuire de referință, =&, cu care putem declara o partajare între două variabile. O astfel de împărțire rupe localitatea mutației.

3 Interfața mobile și baza de date

În cele ce urmează vom cuprinde întreg procesul de creare a interfeței mobile unei case inteligente de asemenea, etapele realizării unei baze de date. Se va lua în considerare fiecare pas al procesului.

3.1 Crearea interfeței mobile

Se va discuta despre o scurtă introducere a aplicației Blynk care va juca un rol substanțial în acest proiect, familiarizarea și folosirea template-urilor, pinilor virtuali și a widget-urilor ce ni le prezintă Blynk.

3.1.1 Blynk introducere

Aici, ne-am folosit de aplicația Blynk, o aplicație complet integrată de software IoT ce poate fi descărcată fie de pe telefon, pe Google Play sau IOS, fie să fie deschisă de pe browser, trebuind doar în facerea unui cont.

Totodată platforma Blynk alimentează producătorii de produse inteligente pentru casă inteligentă, sisteme HVAC complexe, echipamente agricole și toți cei care se află între ele. Obiectivul principal al platformei Blynk este de a face super-ușoară dezvoltarea aplicației pentru telefonul mobil. Dezvoltarea unei aplicații pentru mobil care poate vorbi cu Arduino este la fel de ușoară ca tragerea unui widget și configurarea unui pin.

Blynk poate fi folosit gratuit pentru uz personal și pentru crearea de prototipuri. Modelul lor de afaceri generează profituri prin vânzarea de abonamente companiilor care doresc să publice aplicații bazate pe Blynk pentru produsele sau serviciile lor hardware.

Fiecare proiect poate conține widget-uri grafice, cum ar fi LED-uri virtuale, butoane, afișaje de valori și chiar un terminal text și poate interacționa cu unul sau mai multe dispozitive. Cu ajutorul bibliotecii Blynk, este posibil să controlezi pinii Arduino sau ESP32 direct de pe telefon, fără a fi nevoie să scrii deloc cod.

3.1.2 Crearea device-ului

Acest subcapitol îl vom sectiona în mai mulți pași pentru a fi mai ușor de înțeles și totodată, vom prezenta una dintre metodele prin care, un device poate fi proiectat.

- Formatarea template-ului:

Prin ilustrarea template-ului, trebuie să luăm în considerare numele, tipul de conexiune pe care îl vom folosi, componenta hardware care se va conecta cu device-ul implementat pe parcurs și opțional, descrierea. În cazul nostru, vom lucra cu microcontroller-ul ESP8266 iar conexiunea fiind prin Wi-Fi

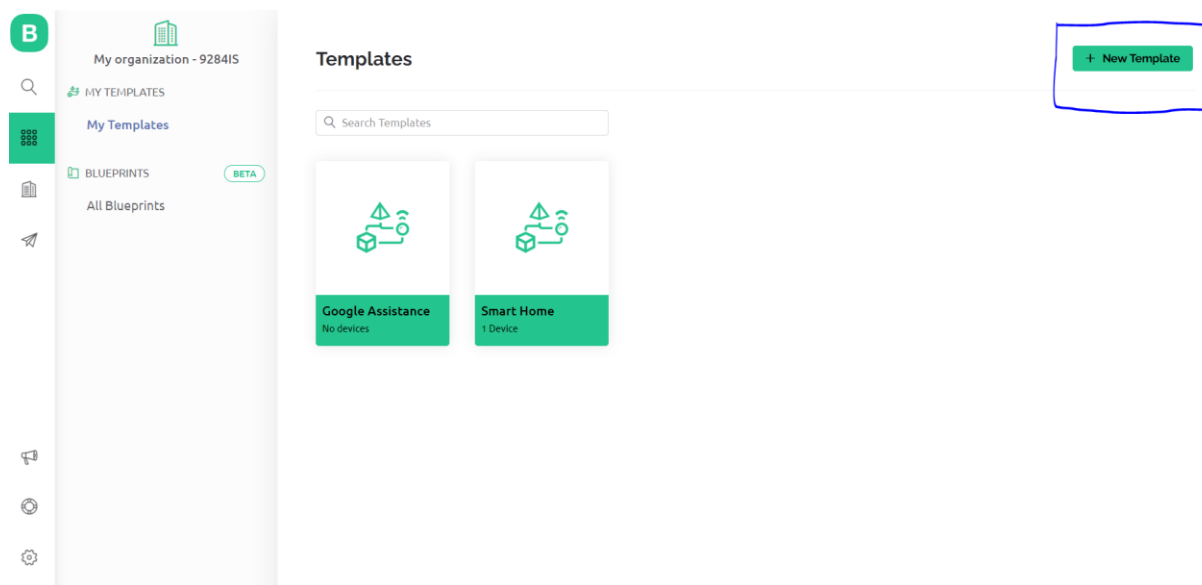


Fig. 3.1 Selectare New Template

Create New Template

NAME

HARDWARE

CONNECTION TYPE

DESCRIPTION

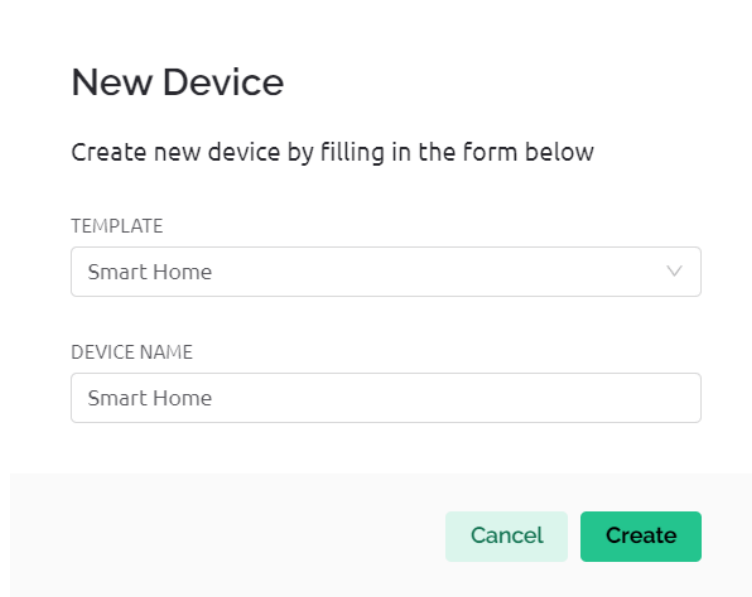
0 / 128

Cancel
Done

Fig. 3.1.1 Completare date template

- Crearea device-ului din template:

Din secțiunea *My Devices* , alegem opțiunea *New Device* și selectăm calea *From template*. Dupa care, completam cu template-ul care il avem si numele device-ului



New Device

Create new device by filling in the form below

TEMPLATE

Smart Home

DEVICE NAME

Smart Home

Cancel Create

Fig. 3.1.2 Formatare device dupa template

- Setare datastreams:

Atunci când vorbim de datastreams în Blynk, automat sunt incluși și pinii virtuali. Pinii virtuali (Virtual Pins) sunt o modalitate de a face schimb de date între hardware-ul nostru și aplicația Blynk. Comparăm pinii virtuali ca la canale pentru trimiterea oricăror date. Trebuie să ne asigurăm că diferențiem pinii virtuali de pinii GPIO fizici de pe hardware-ul nostru. Pinii virtuali nu au reprezentare fizică.

În cazul nostru, trebuie să luăm și în considerare numărul de funcționalități ce vor fi implementate în casa noastră. Așadar, dacă ne vom folosi de senzori pentru: măsurarea valorii temperaturii, a umidității și a gazului dintr-o cameră de asemenea, vom avea, un sistem în caz de incendiu și un sistem de iluminare automat, vom avea nevoie de 5 pini virtuali de integrat. Pinii virtuali sunt găsiți în template-ul nostru, în secțiunea *Datastreams*

Virtual Pin Datastream

View mode
 You are in a "View" mode. Tap on "Edit" button in the top right corner to make changes.

NAME

ALIAS

PIN

DATA TYPE

UNITS

MIN

MAX

DEFAULT VALUE

Close

Fig. 3.1.3 Setare date pin virtual

Așa cum este prezentat în fig. 3.4 , implementarea unui pin virtual este relativ ușor. Trebuie specificat numele pinul (în cazul fig. 3.4 , pinul este destinat sistemului în caz de incendiu, denumit astfel Fire_System) , alegerea pinului (Blynk oferă un total de 255 pini virtuali pe care îi putem folosi, limitarea numărului de funcționalități nu o să reprezinte o problemă) , tipul pinului (în marea majoritate va fi de tip întreg). Pentru această caracteristică, nu avem nevoie să memorăm o anumită variabilă (exemplu fiind temperatura sau umiditatea) deci valorile pentru max și min rămân neschimbate.

Id	Name	Alias	Color	Pin	Data Type	Units	Is Raw	Min	Max	Decimals	Default Value
1	Fire_System	Fire System		V1	Integer		false	0	1	--	0
2	LED	LED		V0	Integer		false	0	1	--	0
3	Temperature	Temperature		V4	Integer		false	0	100	--	0
4	Humidity	Humidity		V5	Integer		false	0	100	--	0
5	MQ2	MQ2		V7	Integer		false	0	2000	--	0

Fig. 3.1.4 Pinii virtuali creați

Pinii noștri virtuali, precum în fig. 3.5 , vor reprezenta fiecare funcționalitate pentru casa noastră inteligentă (diferența dintre pinul folosit pentru sistemul în caz de incendiu, sunt pinii pentru temperatură , umiditate și gaz unde, le-am impus și o maximă pentru ca, rezultatele să fie mai eficient vizualizate)

3.1.3 Adăugare widgets

În continuare realizării aplicației noastre, acum că am avea device-ul făcut și pinii virtuali realizați, trebuie să-I alocăm în widget-urile corespunzătoare. Widget-urile sunt piese pre-proiectate ale interfeței grafice cu utilizatorul. Fiecare widget îndeplinește o funcție specifică de intrare/ieșire atunci când comunică cu hardware-ul sau utilizatorul final. Există 4 tipuri majore de widget-uri: Controllers (Elemente UI utilizate pentru a trimite date către hardware. De exemplu: butoane, comutatoare, glisoare, joystick-uri) , Displays (Elemente UI utilizate pentru a vizualiza datele primite. LED-uri, diagrame, indicatori) , Interface Elements (Elemente pentru a construi o interfață de utilizator ușor de utilizat. File, meniuri drop-down, diverse intrări) și Misc (widget-uri speciale care nu aparțin niciunei categorii).

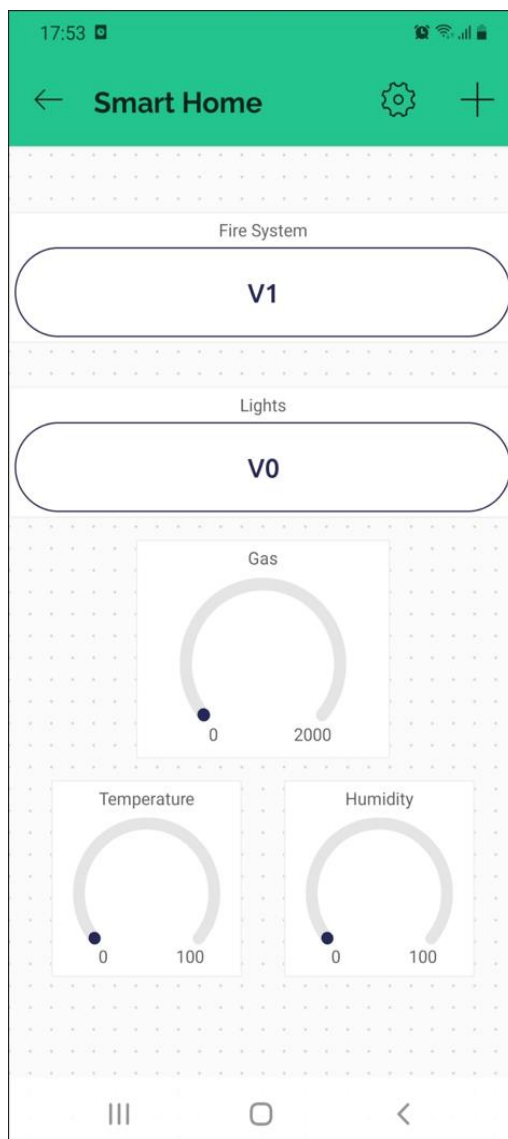


Fig. 3.1.5 Widgets

După cum se vede în fig. 3.6, am folosit 5 widgets pentru grafarea aplicației. Trei gauges pentru măsurarea temperaturii, umidității și a gazului, și două butoane pentru pornirea/oprirea sistemelor de incendiu și iluminat.

3.1.4 Adăugare notificări

Luând în considerare un caz în care, user-ul nu ar fi prezent la fața locului și o situație primejdioasă s-ar putea întâmpla, am luat în considerare implementarea notificărilor pentru funcționalitățile ce au ca rol, prevenirea unor cauze ce ar putea stârni situații amenințătoare.


Acestea pot fi ușor create mergând în opțiunea de templates în site-ul Blynk iar la setări, se merge pe opțiunea de Events.

După care, se setează datele pentru numele notificării , tipul de notificare(info, warning, etc.) și limita de timp la care să se activeze după prima inițializare. Pentru sistemul nostru în caz de incendiu, even-ul va arata precum în figura 3.7

Fire detected!!

General Notifications


EVENT NAME



Fire detected!!

EVENT CODE

fire_detected



Use letters, digits, underscores and hyphens o...

TYPE

Info

Warning

Critical

Content

DESCRIPTION (OPTIONAL)

Adu bidonul cu apa baaai!!!

27 / 300

Limit

Every

1

 message will trigger the event

Event will be sent to user only once per

1 second ▾

Fig. 3.1.6 Notificare sistem de incendiu

În final, interfața noastră mobilă va arată precum în figura 3.7

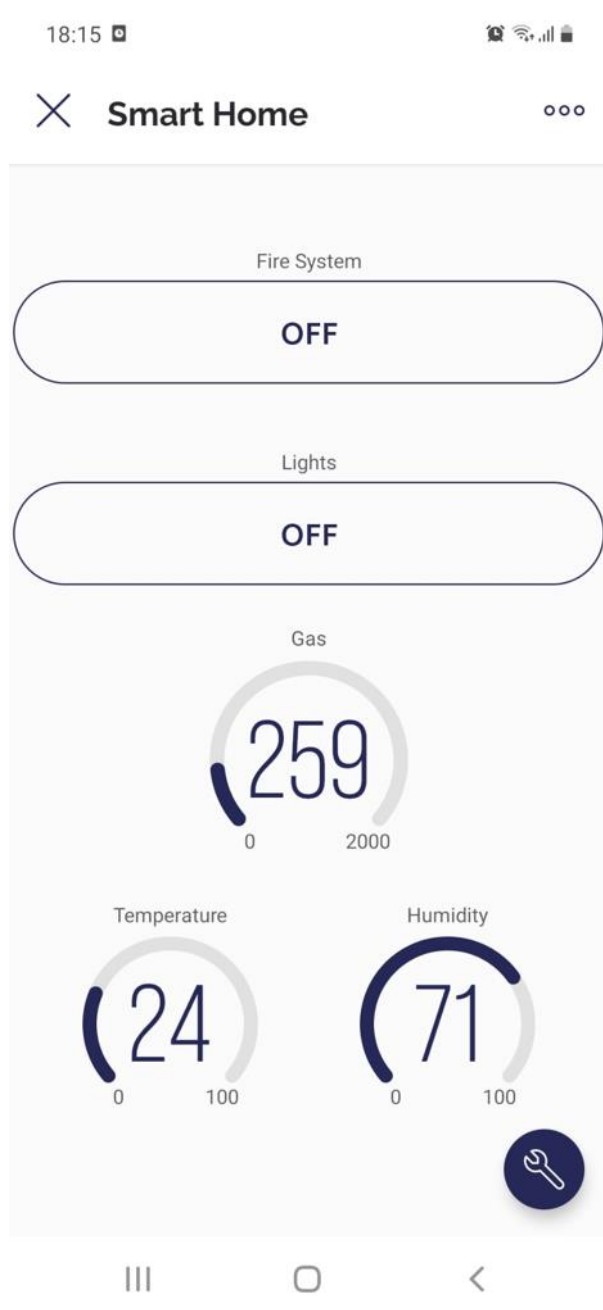


Fig. 3.1.7 Interfața mobilă

3.2 Crearea bazei de date

Pentru acest capitol, vom face o imagine de ansamblu în ceea ce privește phpMyAdmin (site-ul folosit pentru baza de date), formatarea bazei de date și

conștientizarea numărului de tabele necesar pentru a stoca datele casei noastre inteligente.

3.2.1 phpMyAdmin introducere

phpMyAdmin este un instrument software gratuit scris în PHP, destinat să se ocupe de administrarea MySQL pe Web. phpMyAdmin acceptă o gamă largă de operațiuni pe MySQL și MariaDB. Operațiunile utilizate frecvent (gestionarea bazelor de date, tabele, coloane, relații, indexuri, utilizatori, permisiuni etc) pot fi efectuate prin interfața cu utilizatorul, în timp ce aveți încă posibilitatea de a executa direct orice instrucțiune SQL.

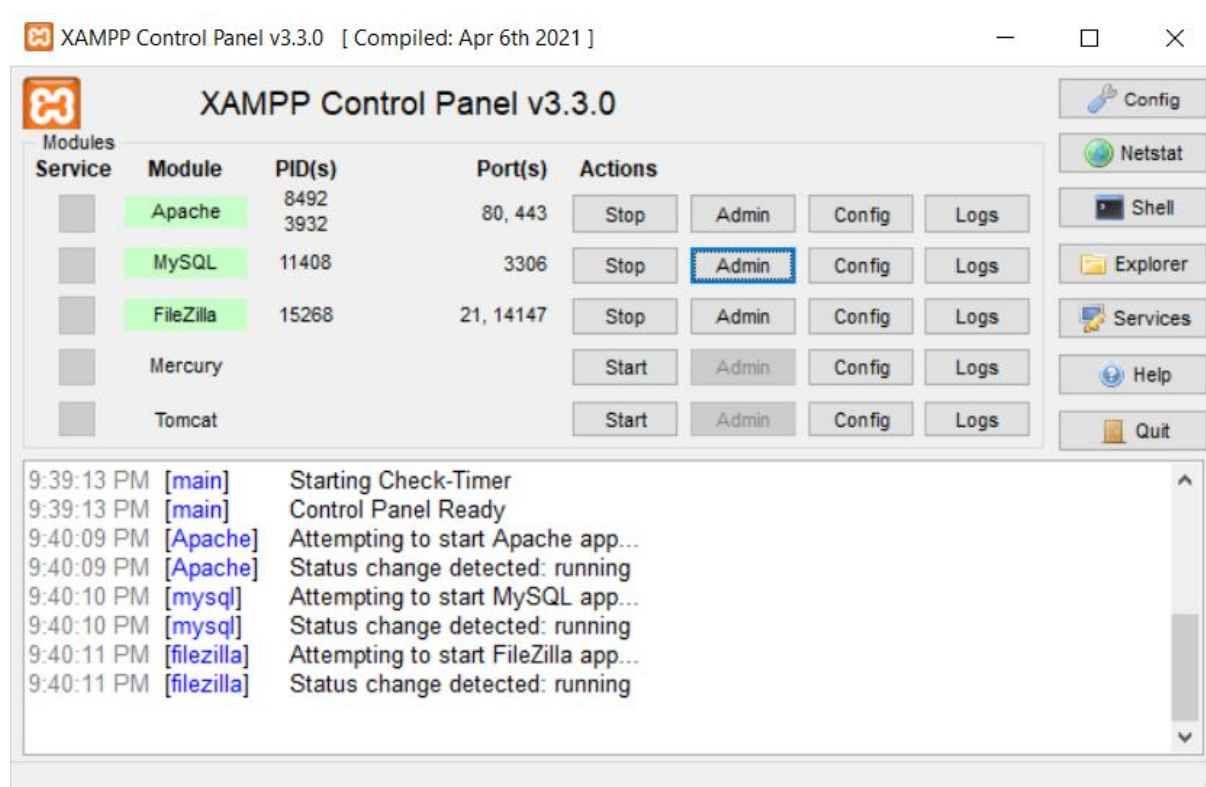


Fig. 3.2 Panoul de comandă XAMPP

Este cea mai populară aplicație pentru gestionarea bazelor de date MySQL. Putem crea, actualiza, elimina, modifica, șterge, importa și exporta tabele de baze de date MySQL utilizând acest software. phpMyAdmin acceptă, de asemenea, o gamă largă de operațiuni, cum ar fi gestionarea bazelor de date, relații, tabele, coloane, indici, permisiuni și utilizatori etc., pe MySQL și MariaDB. Aceste operațiuni pot fi efectuate prin interfața cu utilizatorul, în timp ce avem încă capacitatea de a executa orice instrucțiune SQL.

phpMyAdmin este o aplicație bazată pe GUI care este utilizată pentru a gestiona baza de date MySQL. Putem crea manual o bază de date și un tabel și să executăm interogarea pe ele. Oferă o interfață web și poate rula pe orice server. Deoarece este bazat pe web, îl putem accesa de pe orice computer.

3.2.2 Crearea tabelelor

Am construit două tabele destinate pentru senzorii de temperatură , umiditate și gaz (am specificat anterior că am folosit un senzor DHT11 pentru înregistrarea valorilor temperaturii și a umidității, din acest caz avem doar 2 tabele) cu scopul de a avea salvate toate schimbările valorice și de a observa evoluția datelor.

Prima tabelă este pentru senzorul DHT11

#	Nume	Tip	Colaționare	Atribute	Nul	Implicit	Comentarii	Suplimentar	Acțiune
<input type="checkbox"/> 1	id	int(11)			Nu	Niciuna		AUTO_INCREMENT	Modifică Elimină Mai mult
<input type="checkbox"/> 2	temperature	float			Da	NULL			Modifică Elimină Mai mult
<input type="checkbox"/> 3	humidity	float			Da	NULL			Modifică Elimină Mai mult
<input type="checkbox"/> 4	time_stamp	timestamp			Nu	current_timestamp()			Modifică Elimină Mai mult

Fig. 3.2.1 Tabela DHT11

A doua tabelă pentru senzorul MQ2

#	Nume	Tip	Colaționare	Atribute	Nul	Implicit	Comentarii	Suplimentar	Acțiune
<input type="checkbox"/> 1	id	int(11)			Nu	Niciuna		AUTO_INCREMENT	Modifică Elimină Mai mult
<input type="checkbox"/> 2	gas	float			Nu	Niciuna			Modifică Elimină Mai mult
<input type="checkbox"/> 3	time_stamp	timestamp			Nu	current_timestamp()			Modifică Elimină Mai mult

Fig. 3.2.2 Tabela MQ2

4 Partea hardware și crearea server-ului

Am conceput interfața mobile cu caracteristici pentru fiecare funcționalitate ce va fi implementată în casa inteligentă , am creat totodată și o bază de date pentru a analiza evoluția in timp a datelor fiecărui senzor. Iar acum , in acest capitol, vom ecraniza realizarea schemei electrice a casei inteligente, punând în evidență fiecare cablu conectat la fiecare pin utilizat de asemenea, implementarea codului server-ului cu care vom putea trimite informațiile din microcontroller în baza noastră de date.

4.1 Partea hardware

Acest subcapitol va include afișarea schemei electrice a casei inteligente ,asamblul tuturor componentelor folosite, familiarizarea cu platforma Arduino IDE și legătura între ESP8266 și Blynk.

4.1.1 Total componente utilizate

Pentru produsul final, am avut nevoie de următoarea lista de componente:

1. Fire mama-mamă:

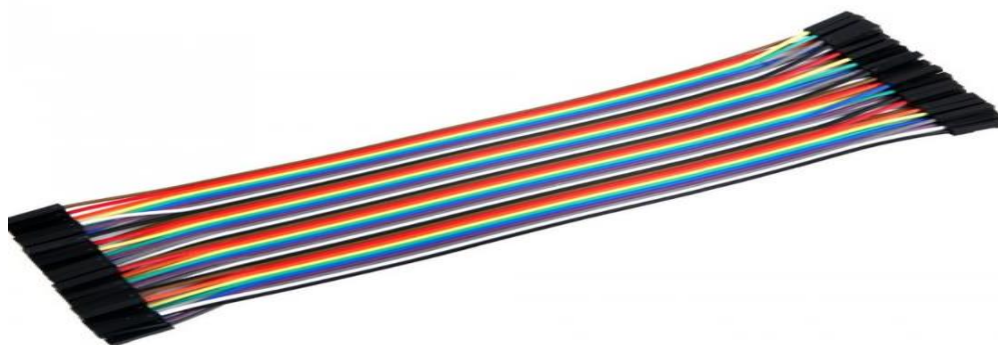


Fig. 4.1 Fire mamă-mamă

2. Fire tată-tată:

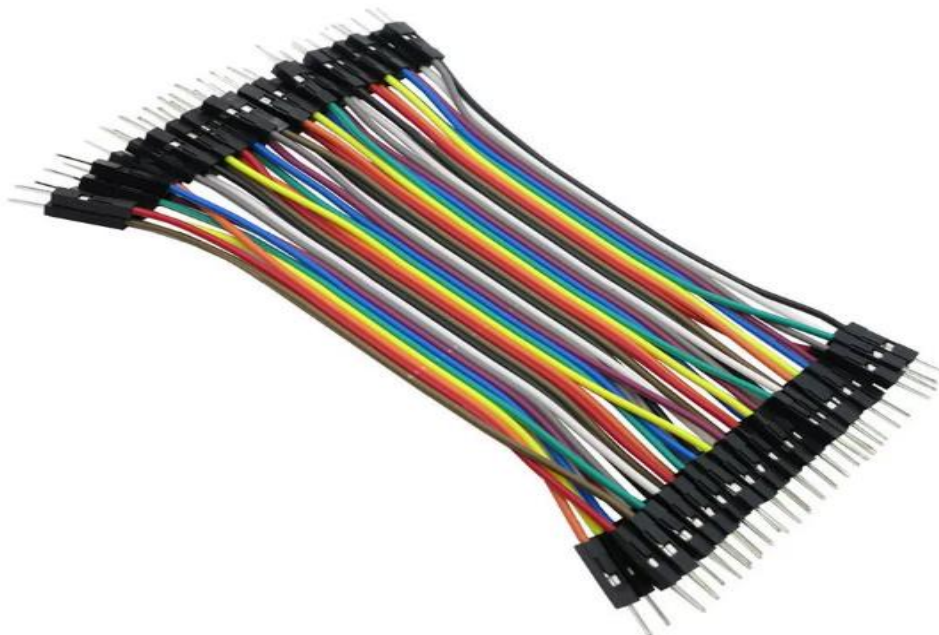


Fig. 4.1.2 Fire tată-tată

3. LED-uri multicolore:



Fig. 4.1.3 Led-uri

4. Buzzere:



Fig. 4.1.4 Buzzer

5. Rezistențe de 180 ohmi:



Fig. 4.1.5 Rezistență

6. Senzor DHT11 (vezi fig. 2.3.3)
7. Senzor MQ-2 (vezi fig. 2.3.4)
8. Senzor de detecție a flăcării (vezi fig. 2.3.5)
9. Breadboard:

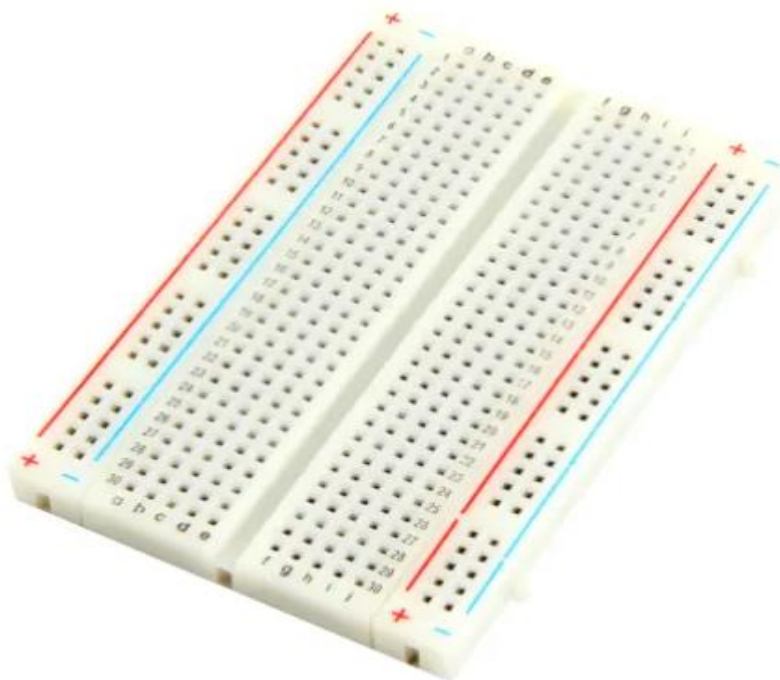


Fig. 4.1.6 Breadboard

10. ESP8266 (vezi fig. 2.2.1)

4.1.2 Schema electrică

Schema electrică ce va intera fiecare componentă menționată în subcapitolul mai sus.

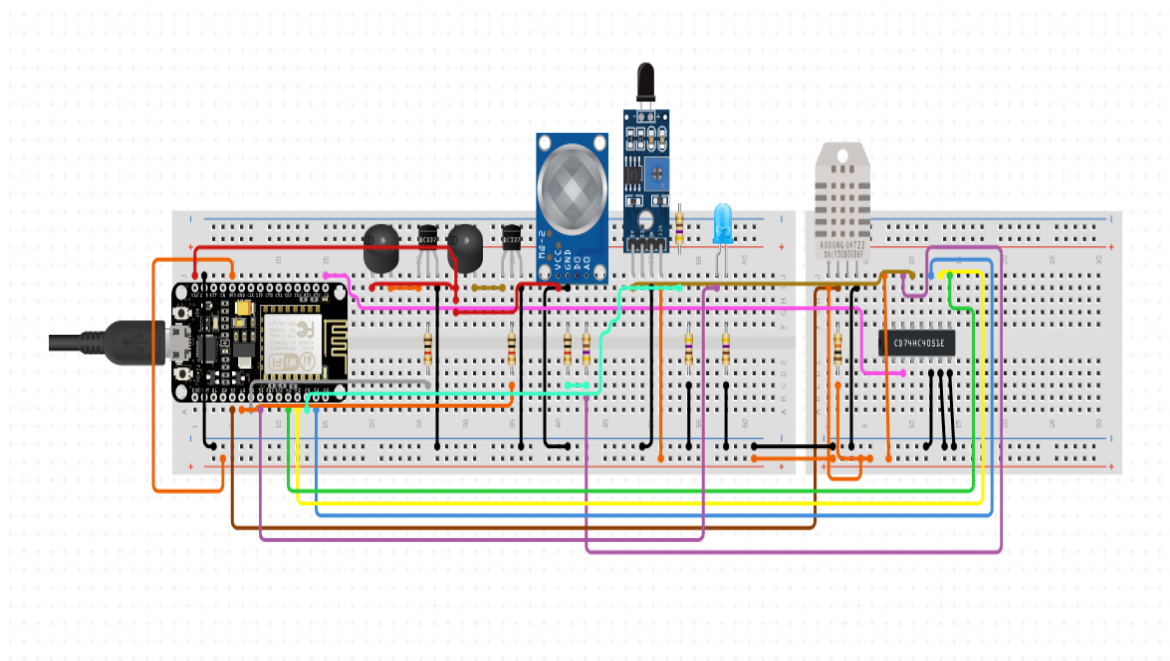


Fig. 4.1.7 Schema electrică a casei

4.1.3 Arduino IDE introducere

Pentru programarea casei și totodată pentru verificarea corectitudinii valorilor senzorilor, am folosit platforma de dezvoltare Arduino IDE.

Arduino Integrated Development Environment - sau Arduino Software (IDE) - conține un editor de text pentru scrierea codului, o zonă de mesaje, o consolă de text, o bară de instrumente cu butoane pentru funcții comune și o serie de meniuri. Se conectează la hardware-ul Arduino pentru a încărca programe și a comunica cu acestea.

Programele scrise folosind software-ul Arduino (IDE) se numesc schițe. Aceste schițe sunt scrise în editorul de text și sunt salvate cu extensia de fișier .ino. Editorul are funcții pentru tăierea/lipirea și căutarea/înlocuirea textului. Zona de mesaje oferă feedback în timpul salvării și exportului și, de asemenea, afișează erori. Consola afișează textul rezultat de către software-ul Arduino (IDE), inclusiv mesaje de eroare complete și

alte informații. Colțul din dreapta jos al ferestrei afișează placa configurată și portul serial. Butoanele din bara de instrumente vă permit să verificați și să încărcați programe, să creați, să deschideți și să salvați schițe și să deschideți monitorul serial.

4.1.4 Legătură între Arduino IDE și Blynk

4.1.4.1 Librării Blynk

Blynk dispune de o varietate largă de librării în care se pot conecta peste 400 de modele hardware (inclusiv ESP8266, ESP32, NodeMCU, toate Arduinos, Raspberry Pi, Particle, Texas Instruments etc.) la Blynk Cloud. În cazul de față, noi vom dispune doar de cea ce suporta microcontroller-ul ESP8266.

```
4  #include <ESP8266WiFi.h>
5  #include <BlynkSimpleEsp8266.h>
6  #include <DHT.h>
```

Fig. 4.1.4.1 Librăriile utilizate

4.1.4.2 Conectare device

În capitolul anterior, când am vorbit despre formatarea unui device pe platforma Blynk cu scopul de a crea pini virtuali și aceștia pe urmă fiind destinați unor widget-uri pentru a se putea realiza comandarea casei inteligente doar prin apăsarea anumitor butoane de pe telefon, au fost create automat un ID ce va reprezenta pe întregul, device-ul nostru și de altfel, un jeton de autentificare, ce face permisa conectarea device-ului de pe telefon cu codul generat în Arduino. În figura 4.2, se poate vedea unde găsiți ID-ul și jetonul de autentificare

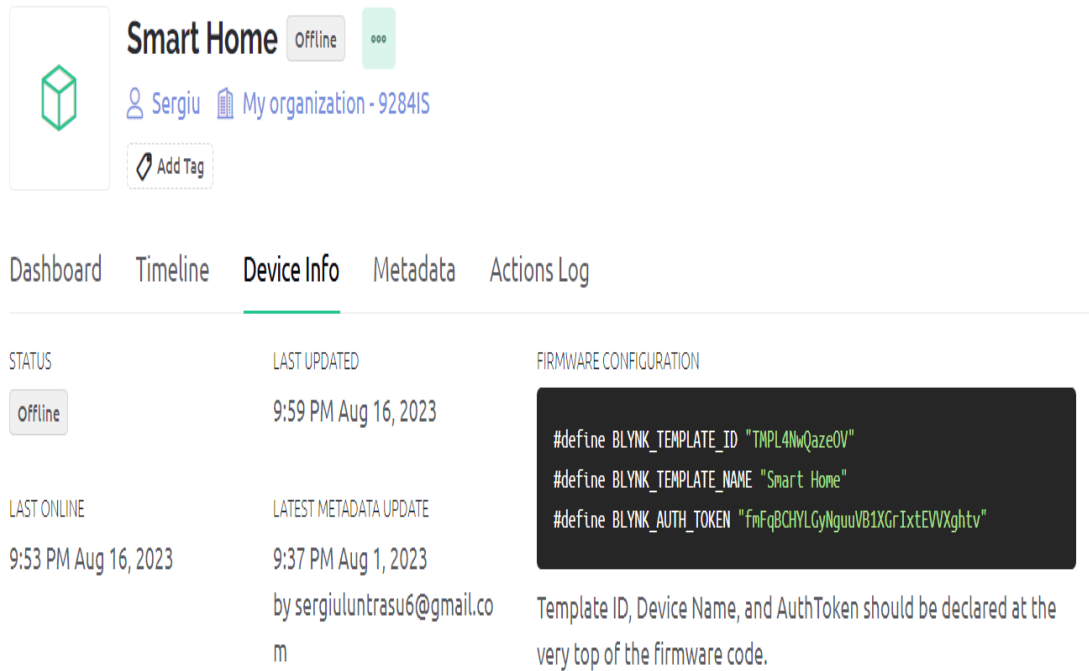


Fig. 4.1.4.2 Datele de conectare

În același timp, pentru ca conexiunea să se realizeze trebuie să integrăm în codul nostru numele rețelei Wi-Fi pe care o folosim în momentul rulării programului și dacă este necesar, parola rețelei corespunzătoare.

```
char auth[] = "fmFqBCHYLgyNguuVB1XGrIxtEWXghtv"; // Enter your Blynk auth token for the new Blynk app project
char ssid[] = "Slax Net"; // Enter your WIFI name
char pass[] = "viatabuna"; // Enter your WIFI password
```

Fig. 4.1.4.3 Datele integrate în Arduino IDE

4.2 Crearea server-ului

În capitolele anterioare am vorbit despre făurirea bazei de date dar acum, trebuie să realizăm și legarea sa de un server, cu scopul de a transmite informațiile din Arduino IDE în casa.

4.2.1 Visual Studio Code introducere

Visual Studio Code combină simplitatea unui editor de cod sursă cu instrumente puternice pentru dezvoltatori, cum ar fi completarea și depanarea codului IntelliSense.

În centrul său, Visual Studio Code are un editor de cod sursă foarte rapid, perfect pentru utilizarea de zi cu zi. Cu suport pentru sute de limbi, vă ajută să fiți instantaneu productiv cu evidențierea sintaxelor, potrivirea parantezelor, indentarea automată, selecția casetelor, fragmente și multe altele. Comenzile rapide intuitive de la tastatură, personalizarea ușoară și mapările de comenzi rapide de la tastatură contribuie de comunitate vă permit să navigați cu ușurință în cod.

Pentru o codare serioasă, Visual Studio Code beneficiază adesea de instrumente cu mai multă înțelegere a codului decât blocuri de text. Totodată include suport încorporat pentru completarea codului IntelliSense, înțelegerea și navigarea bogată a codului semantic și refactorizarea codului.

Iar când codarea devine grea, cei mai duri se depanează. Depanarea este adesea singura caracteristică pe care dezvoltatorii o ratează cel mai mult într-o experiență de codare mai simplă, așa că am realizat-o. Visual Studio Code include un depanator interactiv, astfel încât să poate parcurge codul sursă, să se inspecteze variabilele, să se vizualizeze stivele de apeluri și să se execute comenzi în consolă.

De asemenea, Visual Studi Code se integrează cu instrumente de creare și scriptare pentru a efectua sarcini obișnuite, făcând fluxurile de lucru de zi cu zi mai rapide. Are suport pentru Git, astfel încât să se poată lucra cu controlul sursei fără a părăsi editorul, inclusiv vizualizarea diferențelor de modificări în așteptare.

4.2.2 Scrierea server-ului

Server-ul a fost scris în limbaj PHP, prin care procesează datele primite de la un Arduino sau un dispozitiv similar printr-o solicitare HTTP POST. Se conectează la o bază de date MySQL, inserează date despre temperatură, umiditate și gaz în două tabele diferite și oferă feedback despre succesul sau eșecul procesului de inserare.

În primul rand, am început prin setarea parametrilor pentru host-ul ce va fi folosit, user-ul, parola în cazul în care avem (parolă inexistentă pentru proiectul nostru) și pentru baza noastră de date.

```
<?php
// Database credentials
$dbhost = 'localhost';
$dbuser = 'root';
$dbpass = '';
$dbname = 'database';
```

Fig. 4.2 Credențiale bază de date

După, trebuie totodată să ne salvăm valorile provenite de la senzorii casei inteligente în niscaiva variabile. Pentru asta am folosit metoda POST.

Metoda POST transferă informații prin anteturi HTTP. Informațiile sunt codificate așa cum este descris în cazul metodei GET și introduse într-un antet numit QUERY_STRING. Metoda POST nu are nicio restricție privind dimensiunea datelor care trebuie trimise. Metoda POST poate fi folosită pentru a trimite date ASCII și binare.

```
// Retrieve data from the HTTP request
$temperature = $_POST["temperature"];
$humidity = $_POST["humidity"];
$gas = $_POST["gas"];
```

Fig. 4.2.1 Variabile de la senzori

Precum în figura 4.6 , aceste linii preiau date dintr-o solicitare HTTP POST. Arduino IDE a trimis date cu tastele „temperatură”, „umiditate” și „gaz” în cerere. Valorile sunt stocate în variabilele corespunzătoare (\$temperatura, \$umiditatea și \$gaz).

```
// Check if data is not empty or NULL
if ($temperature === null || $humidity === null || $gas === null) {
    die("Error: Invalid data received from Arduino");
}
```

Fig. 4.2.2 Corectitudine date primite

În figura 4.7 verificăm dacă oricare dintre datele preluate (\$temperatură, \$umiditate sau \$gaz) este nulă. Dacă oricare dintre aceste variabile este nulă, înseamnă că datele trimise de la Arduino sunt incomplete. În acest caz, scriptul se încheie și afișează un mesaj de eroare.

În continuare, o nouă conexiune MySQLi (MySQL Îmbunătățită) este stabilită folosind acreditările bazei de date. Dacă există o problemă de conectare la baza de date, scriptul se încheie și afișează un mesaj de eroare care include detaliile erorii de conectare.

După care, codul pregătește și execută o interogare SQL pentru a insera date într-un tabel numit „dht11” (tabela corespunzătoare bazei de date pentru valorile temperaturi și umidități. Folosește substituenți (?) în interogarea SQL și apoi leagă datele reale (\$temperature și \$humidity) la substituenți folosind metoda bind_param.

Funcția bindParam() leagă un parametru la un substituent cu nume sau semn de întrebare într-o instrucțiune SQL. Funcția bindParam () este utilizată pentru a transmite variabila nu valoare. Funcția bindParam() este executată în timpul execuției. bindParam este o funcție încorporată în PHP.

```
// Prepare the SQL statement with placeholders
$sql = "INSERT INTO dht11 (temperature, humidity) VALUES (?, ?)";

// Create a prepared statement
$stmt = $conn->prepare($sql);

if (!$stmt) {
    die("Error in prepared statement: " . $conn->error);
}

// Bind the parameters and execute the statement
$stmt->bind_param("dd", $temperature, $humidity);

if ($stmt->execute()) {
    echo "Data inserted into the database successfully";
} else {
    echo "Error: " . $stmt->error;
}
```

Fig. 4.2.3 Inserare valori tabela "dht11"

Dacă execuția are succes, ecoa un mesaj de succes; în caz contrar, afișează un mesaj de eroare.

Asemănător pentru cazul tabelii "dht11", va trebui să procedăm și cu tabela "mq2". Se pregătește și execută o interogare SQL pentru a insera date într-un tabel numit „mq2”, care conține probabil informații legate de gaz. Metoda bind_param este folosită pentru a lega valoarea \$gas de substituent. Ecranizează un mesaj de succes sau de eroare bazat pe rezultatul execuției.


```
// Prepare the SQL statement for Gas_info table with placeholders
$sq2 = "INSERT INTO mq2 (gas) VALUES (?)";

// Create a prepared statement for Gas_info table
$stmt2 = $conn->prepare($sq2);

if (!$stmt2) {
    die("Error in prepared statement for Gas_info table: " . $conn->error);
}

// Bind the parameter and execute the statement for Gas_info table
$stmt2->bind_param("d", $gas);

if ($stmt2->execute()) {
    echo "Data inserted into the Gas_info table successfully<br>";
} else {
    echo "Error: " . $stmt2->error;
}

$stmt->close();
$stmt2->close();
$conn->close();
?>
```

Fig. 4.2.4 Inserare valori tabela "mq2"

4.2.3 Evaluarea funcționalităților

În final, avem baza de date creată , avem interfața mobile făcută , server-ul implementat cu partea hardware montată, este timpul pentru verificarea datelor.

```
16:28:06.790 -> [11908] Connected to WiFi
16:28:06.791 -> [11909] IP: 192.168.100.39
16:28:06.823 -> [11909]
16:28:06.857 ->
16:28:06.857 ->  / _ ) / / _ _ _ _ _ / / _
16:28:06.890 ->  / _ / / // / _ \ / ' _/
16:28:06.924 ->  / _ _ / _ \ _ , / _ // _ / _ \ _ \
16:28:06.957 ->           / _ _ / v1.3.0 on ESP8266
16:28:06.990 ->
16:28:06.990 -> #StandWithUkraine   https://bit.ly/swua
16:28:07.022 ->
16:28:07.022 ->
16:28:07.022 -> [12039] Connecting to blynk.cloud:80
16:28:07.099 -> [12163] Ready (ping: 36ms).
```

Fig. 4.2.5 Conexiune cu Blynk

Legătura a fost cu succes stabilită între Arduino IDE și Blynk de asemenea, fiind realizată și conexiunea la rețeaua Wi-Fi.

Iar după cum se poate vedea, datele au fost cu succes afișate în Serial Monitor și transmise bazei de date, server-ul fiind reușit apelat de către Arduino IDE.

```
16:28:18.457 -> Temperature: 27.20
16:28:18.457 -> Humidity: 73.00
16:28:18.502 -> Gas Level: 305
16:28:23.475 -> Data sending failed
16:28:23.645 -> Temperature: 27.20
16:28:23.645 -> Humidity: 73.00
16:28:23.645 -> Gas Level: 306
16:28:28.638 -> Data sending failed
16:28:28.824 -> Temperature: 27.20
16:28:28.824 -> Humidity: 73.00
16:28:28.824 -> Gas Level: 306
16:28:31.644 -> Data sent successfully
16:28:31.805 -> Temperature: 27.30
16:28:31.805 -> Humidity: 73.00
16:28:31.847 -> Gas Level: 307
16:28:31.847 -> Data sent successfully
16:28:32.112 -> Temperature: 27.30
16:28:32.112 -> Humidity: 73.00
16:28:32.159 -> Gas Level: 308
16:28:32.159 -> Data sent successfully
```

Fig. 4.2.6 Conexiunea cu server-ul

Într-un final, casa noastră inteligentă este proiectată și complet funcțională (vezi figura 4.12)

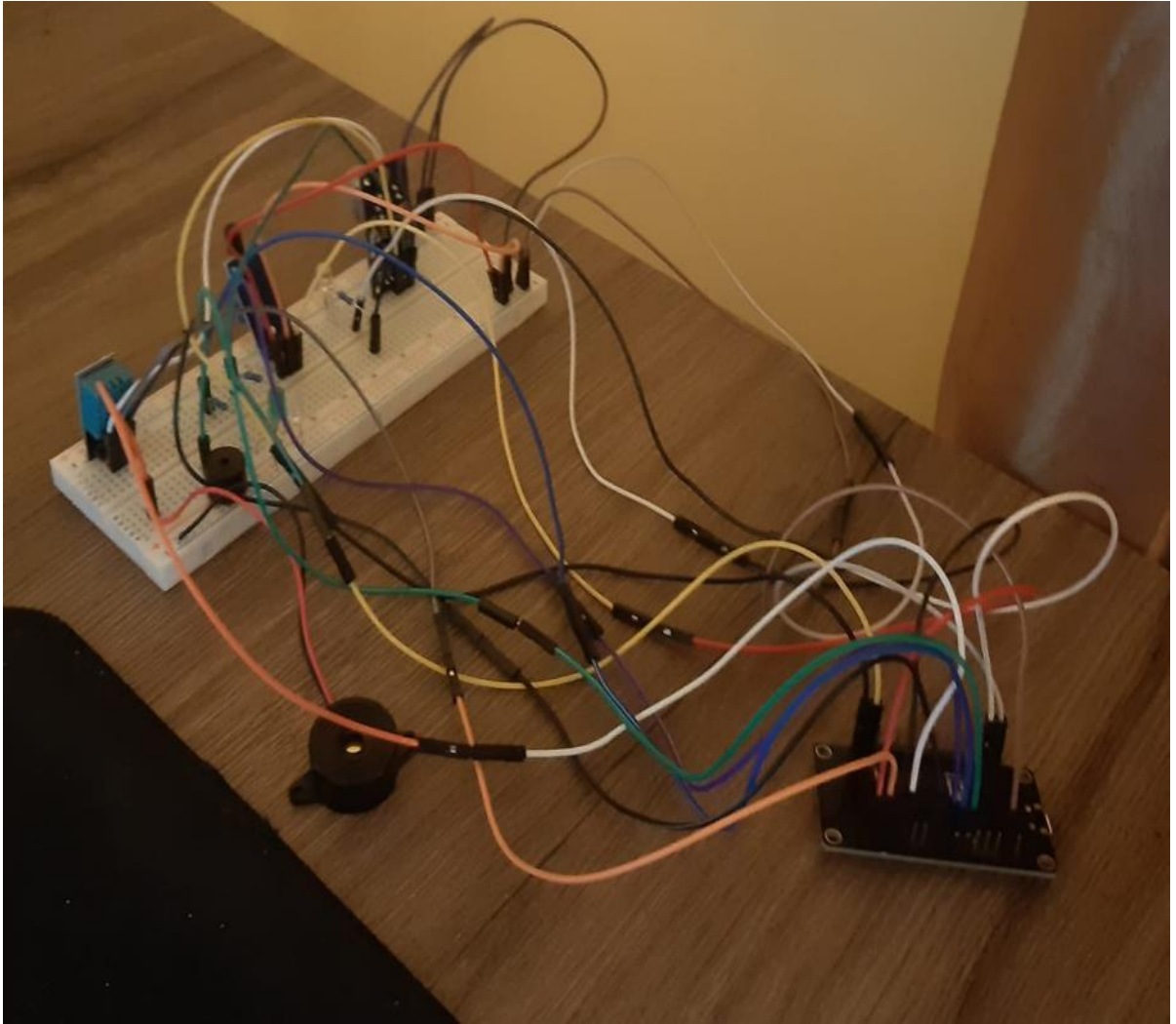


Fig. 4.2.7 Smart Home

5 Testare și validare

Nr. Crt.	Categorie	Test	Pasi	Rezultate	Status
1.	Interfață mobile	Verificarea butoanelor	Revizuirea setărilor datelor fiecărui buton, luând în vedere numele și , modul de apăsare	Butoanele sunt corect inițializate	Trecut
2.	Interfață mobile	Verificarea pinilor virtuali	Revizuirea setărilor create pentru toți	Fiecare pin virtual este	Trecut

			pinii virtuali folosiți, verificare să nu avem doi pini atribuiți aceluiași pin	corect implementat	
3.	Interfață mobile	Verificarea corectitudinii widget-urilor	Verificare dacă am ales widget-urile care trebuie și setarea acestora în conformitate	Fiecare widget este corect stabilit	Trecut
4.	Interfață mobile	Verificarea notificărilor	Revizuirea event-urilor, delay fiind setat pe durata care trebuie	Event-urile setate corect	Trecut
5.	Hardware	Verificarea integrității cablurilor și a componentelor	Revizuirea fiecărui cablu montan în breadboard și în microcontroller totodată, verificarea fiecărei componente fiind pusă și setată corespunzător	Fiecare cablu și componentă au fost setate corect	Trecut
6.	Hardware	Verificarea funcționalității microcontroller-ului	Se verifică dacă plăcuța ESP8266 este inițializată corespunzător	ESP8266 este complet funcțional	Trecut
7.	Interfață mobile + Hardware	Stabilirea legăturii între casa inteligentă și aplicația mobile	Se vizualizează formarea conexiunii dintre casă și aplicație, dacă	Conexiunea a fost realizată cu succes, Arduino IDE și Blynk	Trecut

			este realizată cu succes	confirmându-ne asta	
8.	Interfață mobile + Hardware	Verificarea corectitudinii datelor de la microcontroller în aplicație	Folosindu-ne de Arduino IDE și aplicația noastră mobile, verificăm dacă datele noastre coincid	Datele coincid și sunt precise (vezi fig. 5 și fi. 5.1)	Trecut
9.	Interfață mobile + Hardware	Verificarea butoanelor, acționand la comenzile date de utilizator	Verificăm dacă într-adevar aplicația noastră merge doar prin acționarea din aplicație	Casa este complet controlabilă de pe telefon	Trecut
10.	Interfață mobile + Hardware	Verificare primire notificări	Verificăm dacă pe baza cerințelor de siguranță (notificând user-ul cu privire la un posibil incendiu) impuse de noi se desfășoară conform așteptărilor noastre	Aplicația trimite notificări în eventualele situații de pericol	Trecut
11.	Server	Verificare primire date în baza de date	Să se vada dacă server-ul reușește cu succes să facă conexiune cu baza de date	Server-ul face rapid conexiunea cu baza de date, neexistând vreun delay	Trecut

```
15:06:31.564 -> Data sent successfully  
15:06:32.553 -> Temperature: 27.10  
15:06:32.553 -> Humidity: 66.00  
15:06:32.595 -> Gas Level: 232
```

Fig. 5 Date Serial Monitor

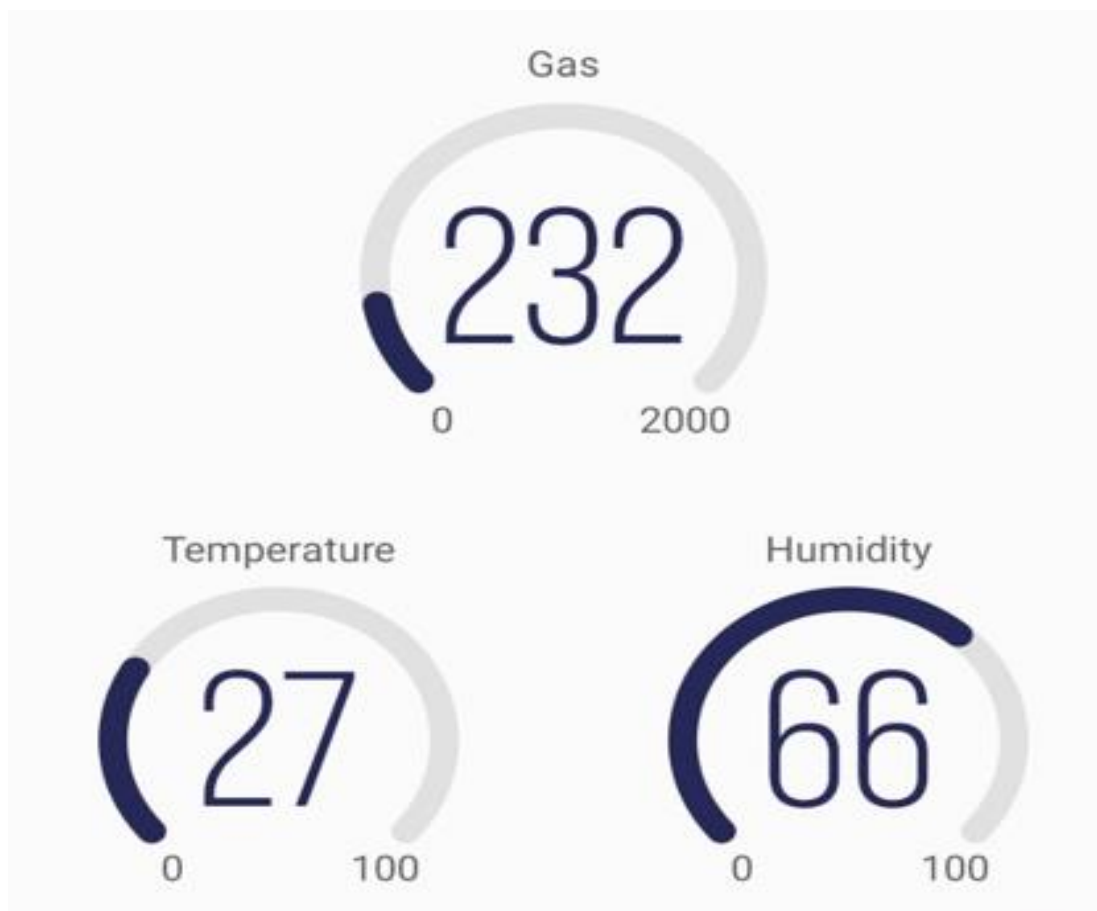


Fig. 5.1 Date aplicație mobile

6 Concluzii

6.1 Rezultate obținute și păreri finale

Odată cu dezvoltarea rapidă a Internetului și tehnologiile de comunicare și casele de astăzi au calcul și comunicare puternice abilități. O casă inteligentă bazată pe IoT este în curs de dezvoltare ca o parte importantă a celor deștepți și inteligenți orașe care sunt propuse și dezvoltate în jurul lumii. Scopul unei case inteligente este de a îmbunătăți nivelul de trai, securitatea și siguranța precum și economisiți energie și resurse. Cel inteligent acasă joacă un rol important în dezvoltarea societate.

Aplicația casei noastre inteligente poate fi accesibilă de pe orice tip de mobil cu acces la internet. Acest lucru adaugă flexibilitate și ușurează interacțiunea cu cererea prezentată. Întrucât interfața aplicației este ușor de înțeles, acest proiect poate fi folosit de ambele sisteme ingineri și oameni cu puține cunoștințe în tehnologie.

În ceea ce privește dispozitivele inteligente pe care le folosește, acest sistem are alt avantaj. Acesta creează o interfață unică pentru gestionarea dispozitivelor, vorbind în mare parte de sistemele de iluminat și de incendiu. De aceea utilizatorul trebuie să acceseze o singură aplicație pentru a putea controla casa acestuia.

Ca o concluzie, această lucrare arată cum să vă conectați inteligent dispozitive într-o singură entitate care utilizează platforma software IoT Blynk și microcontroller-ul cu Wi-Fi implementat Nodemcu ESP8266. Ideea prezentată are un aspect flexibil și extensibil configurație care poate fi ușor adaptată la noile aparate și caracteristici.

De ar fi să adaug, circumstanțele anevoioase ale proiectului cu scopul de a facilita, viitoarele proiecte bazate în IoT, ar fi formatarea server-ului în limbaj Php. În situația mea, nefiind experimentat cu acest limbaj, a fost necesar să-l învăț de la 0 iar, pe parcursul implementării codului server-ului, am întâmpinat funcții integrate acestui limbaj ce mi-au oferit anumite bătăi la cap pentru a putea fi familiarizat cu ele.

Pe de-o altă parte, cu toate că a fost dificil, a fost rezultat și un beneficiu, reușind să învăț pe cont propriu bazele unui limbaj nou de programare.

Într-un final, consider că prin ceea ce mi-am propus inițial că voi realiza în acest proiect a fost desăvârșit, fiecare componentă fiind cu succes implementată în schița finală de asemenea, menținându-ne la un buget corespunzător.

6.2 Direcții de dezvoltare

Dezvoltarea viitoare a acestei soluții de casă inteligentă poate fi concentrat pe mai multe domenii. Pe de o parte s-ar putea integra mai multe tipuri de aparate inteligente pentru a încorpora diverse tipuri de activități acasă. Pe lângă aceasta, pentru dispozitivul nostru poate fi dezvoltat mai multe caracteristici pentru dispozitivele deja adăugate. Acestea includ setarea scenelor și temelor implicite care automatizează sarcinile aparatelor inteligente de acasă pe baza configurațiile utilizatorului.

Pe lângă acestea, aș mai putea enumera următoarele îmbunătățiri:

- Pe partea de hardware, ar fi fost indicat implementarea unui multiplexor. Pe parcursul proiectului, nu m-am documentat îndeajuns încât să iau la cunoștință faptul că anumiți pini ai microcontroller-ului ESP8266 nu se pot utiliza. Încă nu am aflat motivul acestei incursiuni fapt ce m-a obligat să reduc numărul de pini folosiți. Integrarea unui multiplexor ar fi fost mult mai facilă adăugarea mai multor funcționalități
- Pe partea de software, în special pentru aplicația mobile, aș integra un sistem de securitate. Aplicația prezentă poate fi accesată de aproape oricine are în posesie, telefonul respectiv. Un sistem de log-in pentru fiecare user ar fi rezolvat acest impediment

7 References

- [1] A. V. M.-T. P. A. B. D. L. A. D. S. Tudose, ""Home automation design using 6LoWPAN wireless sensors networks"".
- [2] "Lombreglia, R. (2010) The Internet of Things, Boston Globe. Retrieved October."
- [3] "Reinhardt, A. (2004) A Machine-to-Machine Internet of Things."
- [4] "Cyber-physical Systems and Internet of Things".
- [5] "https://www.sciencedirect.com/science/article/pii/S0167739X13000241".
- [6] S. Kumar, "Interfacing Flame Sensor with Arduino to Build a Fire Alarm System," 2018.
- [7] "IoTivity, <http://www.iotivity.org/>, Last Access: April 24th 2016".
- [8] "Philips Hue Developer Program, <http://developers.meethue.com/>, Last".
- [9] "Lou Frenzel, Electronic Design, What's the difference between ZigBee".
- [10] "Apple HomeKit, <http://www.apple.com/ios/homekit/>, Last Access: April".
- [11] "Z-Wave Smart Lock, <http://danalock.com/>, Last Access: May 2016".
- [12] "The Oplink Connected Alarm Shield, <http://www.oplinkconnected.com/>,".
- [13] "Samsung SmartThings Home Monitoring Kit,".
- [14] "What is Grove Starter Kit Plus – Intel IoT Edition?".
- [15] "P. Klint, T. van der Storm, and J. J. Vinju, "RASCAL: A Domain Specific".
- [16] O. C. St. Catharine's, "DHT11 Sensor," vol. Advanced Raspberry Pi.
- [17] "Lee, D.-D., & Lee, D.-S. (2001). Environmental gas sensors. IEEE Sensors Journal, 1(3), 214–224."
- [18] "Lianos, M. and Douglas, M. (2000) Dangerization and the End of Deviance: The Institutional Environment. British Journal of Criminology, 40, 261-278."

- [19] "Ferguson, T. (2002) Have Your Objects Call My Object. Harvard Business Review, June, 1-7. [Citation Time(s):1]".
- [20] "Nunberg, G. (2012) The Advent of the Internet: 12th April, Courses. [Citation Time(s):1]".
- [21] "Kosmatos, E.A., Tselikas, N.D. and Boucouvalas, A.C. (2011) Integrating RFIDs and Smart Ob-jects into a Unified Internet of Things Architecture. Advances in Internet of Things: Scientific Research, 1, 5-12."
- [22] "Aggarwal, R. and Lal Das, M. (2012) RFID Security in the Context of "Internet of Things". First International Conference on Security of Internet of Things, Kerala, 17-19 August 2012, 51-56."
- [23] "Biddlecombe, E. (2009) UN Predicts "Internet of Things". Retrieved July 6. [Citation Time(s):1]".
- [24] "Butler, D. (2020) Computing: Everything, Everywhere. Nature, 440, 402-405."
- [25] "Gershenfeld, N., Krikorian, R. and Cohen, D. (2004) The Internet of Things. Scientific American, 291, 76-81."