

**SISTEM PORTABIL PENTRU MONITORIZAREA
FACTORILOR CARE INFLUENȚEAZĂ SĂNĂTATEA
PORTABLE SYSTEM FOR MONITORING FACTORS
INFLUENCING HEALTH**

Student:

**gr. TI-206,
Pleșu Cătălin**

Coordonator:

**Secrieru Adrian
asistent universitar**

Chișinău, 2024

MINISTERUL EDUCAȚIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA

Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică și Microelectronică

Departamentul Ingineria Software și Automatică

Admis la susținere

Șef departament:

FIODOROV Ion dr., conf.univ.

_____ 2024
„___”

SISTEM PORTABIL PENTRU MONITORIZAREA FACTORILOR CARE INFLUENȚEAZĂ SĂNĂTATEA

Proiect de licență

Student: _____ **Pleșu Cătălin, TI-206**

Coordonator: _____ **Secrieru Adrian, asist. univ.**

Consultant: _____ **Cojocaru Svetlana, asist.univ.**

Chișinău, 2024

Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică și Microelectronică

Departamentul Ingineria Software și Automatică

Programul de studii Tehnologia informației

Aprob
Șef departament:
Fiodorov Ion, dr., conf.univ.

„31” octombrie 2023

CAIET DE SARCINI pentru proiectul de licență al studentului

Pleșu Cătălin

(numele și prenumele studentului)

1. Tema proiectului de licență Sistem portabil pentru monitorizarea factorilor care influențează sănătatea

confirmată prin hotărârea Consiliului facultății nr. 2 din „31” octombrie 2023

2. Termenul limită de prezentare a proiectului de licență 20.05.2024

3. Date inițiale pentru elaborarea proiectului de licență Sarcina pentru elaborarea proiectului de diplomă.

4. Conținutul memoriului explicativ

Introducere

1 Analiza domeniului de studiu

2 Scopul, obiectivele și cerințele sistemului

3 Modelarea și proiectarea sistemul

4 Realizarea sistemului

5 Documentarea produsului realizat

6 Estimarea costurilor proiectului

Concluzii

Modelul paginii de pornire, principale și setări, Modelul paginii pentru alarme, configurări și detalii, Gestionarea sistemului, Stările dispozitivului, Tranzițiile protocolului de comunicare, Componentele aplicației mobile, Componentele softwareului dispozitivului, Implementarea sistemului, Arhitectura hardware a dispozitivului, Diagrama entităților și relațiilor bazei de date, Lista parametrilor configurabili.

Consultant	Capitol	Confirmarea realizării activității	
		Semnătura consultantului (data)	Semnătura studentului
Cojocarui Svetlana	Standarde tehnologice, Controlul calității, Estimarea costului proiectului		

Nr. crt.	Denumirea etapelor de proiectare	Termenul de realizare a etapelor	Nota
1	Elaborarea sarcinii, primirea datelor pentru sarcină	04.09.23– 29.09.23	10%
2	Analiza domeniului de studiu	03.10.23– 27.10.23	15%
3	Proiectarea sistemului	30.10.23 – 22.11.23	15%
4	Realizarea sistemului	05.01.24 – 10.04.24	35%
5	Descrierea sistemului	11.04.24– 19.04.24	10%
6	Estimarea costurilor sistemului	26.04.24– 30.04.24	15%
7	Finisarea proiectului	04.05.24– 14.05.24	5%

Coordonator de proiect de licență Secrieru Adrian ()

Declarația studentului

Subsemnatul Pleșu Cătălin, declar pe proprie răspundere că lucrarea de față este rezultatul muncii mele, pe baza propriilor cercetări și pe baza informațiilor obținute din surse care au fost citate și indicate, conform normelor etice, în note și bibliografie. Declar că lucrarea nu a mai fost prezentată sub această formă la nici o instituție de învățământ superior în vederea obținerii unui grad sau titlu științific ori didactic.

Pleșu

FACULTATEA CALCULATOARE, INFORMATICĂ ȘI MICROELECTRONICĂ
DEPARTAMENTUL INGINERIA SOFTWARE ȘI AUTOMATICĂ
PROGRAMUL DE STUDII TEHNOLOGIA INFORMAȚIEI

AVIZ

la proiectul de licență

Titlul: Sistem portabil pentru monitorizarea factorilor care influențează sănătatea.

Studentul Pleșu Cătălin gr. TI-206

1. Actualitatea temei: Factorii mediului influențează direct sănătatea umană, mulți dintre acești factori nici măcar nu pot fi identificați cu ajutorul simțurilor umane. Prin intermediul tehnologiilor IoT poate fi monitorizat mediul și înștiințat utilizatorul în caz de abateri de la norme.

2. Caracteristica proiectului de licență: Sistemul a fost creat pentru a facilita analiza parametrilor mediului din proximitatea utilizatorului. Și ca acesta să fie alertat în dependență de condițiile stabilite de el.

3. Analiza prototipului: Aplicația mobilă este utilizată pentru a afișa datele colectate de pe dispozitiv.

4. Estimarea rezultatelor obținute: Acest sistem este capabil de a monitoriza parametrii mediului. Sistemul poate emite și alarme.

5. Corectitudinea materialului expus: Materialul expus face referințe la sursele acestuia.

6. Calitatea materialului grafic: Proiectul este prezentat prin: imagini, diagrame și tabele.

7. Valoarea practică a proiectului: Aplicația poate fi instalată pe dispozitive cu sistem de operare Android cel puțin versiunea 11, ce dispun de adaptor Bluetooth.

8. Observații și recomandări: Cerințele față de teza de licență au fost îndeplinite.

9. Caracteristica studentului și titlul conferit: Studentul Pleșu Cătălin a dat dovadă de profesionalism în elaborarea lucrării, a respectat cerințele impuse și a elaborat calitativ teza de licență. Lucrarea de licență poate fi admisă spre susținere.

Lucrarea de licență poate fi admisă spre susținere, cu nota: _____

Lucrarea în forma electronică corespunde originalului prezentat către susținere publică.

Coordonatorul proiectului de licență _____ Secrieru Adrian,
asist.univ

semnătura, data

REZUMAT

Lucrarea de licență cu denumirea ‘Sistem portabil pentru monitorizarea factorilor care influențează sănătatea’ are ca scop prevenirea deteriorării sănătății umane ca urmare a expunerii la factori ai mediului cu valori nefavorabile.

Structura tezei de licență cuprinde introducere, șase capitole, concluzii, bibliografie din 37 de surse, 50 de pagini de text de bază, 7 tabele, 46 figuri și o anexă care conține un tabel cu estimarea costurilor.

Analiza domeniului de studiu, este primul capitol, în acesta este analizat domeniul de monitorizare a mediului și internetul lucrurilor, prezentat importanța temei, analizate trei sisteme similare cu sistemul realizat, analizate tehnologiile utilizate pentru crearea sistemelor IoT și identificate norme legate de parametrii care urmează a fi monitorizați.

Scopul, obiectivele și cerințele sistemului, este al doilea capitol în care au fost expuse scopul și obiectivele, și au fost specificate cerințele funcționale și non-funcționale ale sistemului.

Modelarea și proiectarea sistemului, capitolul trei descrie comportamentul și structura sistemului. Majoritatea modelelor sunt diagrame UML cu excepția arhitecturii hardware care nu corespunde nici unui standard.

Realizarea sistemului, a cuprins trei etape. Prima a fost realizarea aplicației mobile. A doua etapă a fost realizarea dispozitivului. Ultima etapă, testarea a fost efectuată în paralel cu celelalte două etape.

Documentarea produsului realizat, prezintă modul de a utiliza sistemul creat. Fiecare funcțional este descris în detaliu.

Estimarea costurilor proiectului constă din împărțirea proiectului pe etape și sarcini, până rămânem cu sarcini indivizibile. Și alocarea timpului și duratei de executare a acestor sarcini. Au fost identificate salariile pentru diverse roluri în domeniul IT și ca urmare a fost realizată diagrama WBS și Gantt.

ABSTRACT

The bachelor's thesis entitled "Portable system for monitoring factors influencing health" aims to prevent the deterioration of human health as a result of exposure to environmental factors with unfavorable values.

The structure of the bachelor's thesis includes an introduction, six chapters, conclusions, bibliography from 37 sources, 50 pages of basic text, 7 tables, 46 figures and an appendix containing a table with cost estimates.

The analysis of the field of study, is the first chapter, is presented the importance of the theme, and are analyzed the following things: field of environmental monitoring, Internet of things, three systems similar to the system created and the technologies used to create IoT systems. Are identified some norms related to the parameters that are to be monitored.

The purpose, objectives and requirements of the system, are exposed in the second chapter. The requirements of the system are made of functional and non-functional requirements.

System Modeling and Design, chapter describes system behavior and structure. Most of the models are UML diagrams except for the hardware architecture which does not conform to any standard.

The development of the system included three stages. The first was the creation of the mobile application. The second stage was making the device. The last stage, the testing was carried out in parallel with the other two stages.

The documentation of the created product, presents how to use the created system. Each function is described in detail.

Project cost estimation consists of dividing the project into stages and tasks, remaining with indivisible tasks. And allocating the time and duration of these tasks. Salaries for various IT roles were identified and as a result WBS and Gantt chart was produced.

CUPRINS

ABREVIERI.....	10
INTRODUCERE.....	12
1 ANALIZA DOMENIULUI DE STUDIU.....	13
1.1 Importanța temei.....	14
1.2 Sisteme similare cu proiectul realizat.....	14
1.3 Analiza tehnologiilor.....	16
1.4 Parametrii mediului.....	17
2 SCOPUL, OBIECTIVELE ȘI CERINȚELE SISTEMULUI.....	20
2.1 Cerințe funcționale.....	21
2.2 Cerințe non-funcționale.....	23
3 MODELAREA ȘI PROIECTAREA SISTEMULUI.....	24
3.1 Descrierea comportamentală a sistemului.....	24
3.1.1 Imaginea generală asupra sistemului.....	25
3.1.2 Modelarea vizuală a fluxurilor.....	28
3.1.3 Stările de tranzație a sistemului.....	30
3.1.4 Descrierea scenariilor de utilizare a aplicației.....	32
3.1.5 Fluxurile de mesaje și legăturile dintre componentele sistemului.....	33
3.2 Descrierea structurală a sistemului.....	34
3.2.1 Descrierea structurii statice a sistemului.....	35
3.2.2 Relațiile de dependență între componentele sistemului.....	36
3.2.3 Modelarea echipamentelor mediului de implementare.....	37
4 REALIZAREA SISTEMULUI.....	40
4.1 Descrierea aplicației mobile.....	41
4.2 Descrierea firmware-ului pentru dispozitiv.....	44
4.3 Testarea sistemului.....	45
5 DOCUMENTAREA PRODUSULUI REALIZAT.....	48
6 ESTIMAREA COSTURILOR PROIECTULUI.....	56
CONCLUZII.....	58
BIBLIOGRAFIE.....	59
ANEXA A.....	61

ABREVIERI

IoT (eng. Internet of Things) - internetul lucrurilor.

IT (eng. Information Technology) - tehnologia informației.

PM1 (eng. Particulate matter 1) - particule în suspensie cu diametrul de 1 micrometru.

PM2.5 (eng. Particulate Matter 2.5) - particule în suspensie cu diametrul de 2.5 micrometri.

PM10 (eng. Particulate Matter 10) - particule în suspensie cu diametrul de 10 micrometri.

VOCs (eng. Volatile Organic Compounds) - compuși organici volatili,.

ANT+ (eng. Advanced and Adaptive Network Technology) - tehnologie de rețea avansată și adaptivă.

Wi-Fi (eng. Wireless Fidelity) – fidelitate fără fir.

FPGA (eng. Field-Programmable Gate Array) - matrice de porți programabilă.

ASIC (eng. Application-Specific Integrated Circuit) - circuit integrat specific aplicației.

VHDL (eng. VHSIC (eng. Very High-Speed Integrated Circuit Hardware Description Language) Hardware Description Language) - limbaj de descriere a hardware-ului.

OS (eng. Operating System) - sistem de operare.

RTOS (eng. Real-Time Operating System) - sistem de operare în timp real.

OTA (eng. Over-the-Air) - actualizare prin aer.

LoRaWAN (eng. Long Range Wide Area Network) - rețea de arie largă cu distanțe mari.

AMQP (eng. Advanced Message Queuing Protocol) – protocol avansat de mesaje bazat pe cozi.

MQTT (eng. Message Queuing Telemetry Transport) - protocolul de transport al mesajelor bazat pe cozi.

HTTP (eng. Hypertext Transfer Protocol) - protocolul de transfer al hipertextului.

CoAP (eng. Constrained Application Protocol) - protocolul constrâns de aplicație.

DDS (eng. Data Distribution Service) - serviciul de distribuție a datelor.

IO (eng. Input/Output) – intrare/ieșire.

HG - Hotărâre Guvernamentală.

CSV (eng. Comma-Separated Values) - valori separate prin virgulă.

URL (eng. Uniform Resource Locator) - localizator uniform de resurse.

API (eng. Application Programming Interface) - interfață de programare a aplicației.

CI (eng. Continuous Integration) - integrare continuă.

CD (eng. Continuous Delivery) - livrare continuă.

UML (eng. Unified Modeling Language) - limbajul de modelare unificată.

SysML (eng. Systems Modeling Language) - limbajul de modelare a sistemelor.

AADL (eng. Architecture Analysis and Design Language) - limbajul de analiză și proiectare a arhitecturii.

OPM (eng. Object Process Methodology) – metodologia obiect proces.

RTC (eng. Real Time Counter/Clock) – contor în timp real.

RAM (eng. Random Acces Memory) – memorie cu acces aleatoriu.

Gnu (eng. GNU not Unix) – Gnu nu este Unix.

ARM (eng. Advanced RISC (eng. Reduced Instruction Set Computer) Machine) – mașină RISC avansată.

SSID (eng. Service Set Identifier) – identificator de set de servicii.

INTRODUCERE

În cadrul acestei lucrări este documentată fiecare etapă pentru realizarea sistemului de monitorizare a mediului. Sistemul are la bază tehnologiile IoT. Elementele sistemului sunt aplicația mobilă și dispozitivul. Avantajele utilizării unui astfel de sistem în comparație cu un termometru simplu spre exemplu sunt că acest sistem stochează datele pe termen lung și pot fi analizate ulterior. Un alt avantaj este că sistemul poate alerta utilizatorul în diverse cazuri.

Sistemul dat se încadrează atât în domeniul de monitorizare al mediului cât și în domeniul internetul lucrurilor. În cadrul acestei lucrări s-a făcut accent pe IoT.

O cerință de bază către sistem este că acesta trebuie să monitorizeze temperatura aerului, umiditatea relativă, calitatea aerului, intensitatea luminii și nivelul de zgomot. Iar o cerință benefică către dezvoltarea proiectului este ca acesta să fie Open Source și să fie găzduit pe GitHub, astfel cineva ar putea contribui la el în viitor.

Este simplu de a monitoriza acești parametri însă în ansamblu aceștia pot spune multe lucruri despre mediul în care se află utilizatorul, și cum acesta poate să îi influențeze sănătatea.

Printre cele mai utile artefacte rezultate din proiectarea sistemului se numără diagrama protocolului de comunicare, diagrama de plasare și imaginea conceptului hardware pentru dispozitiv.

Pentru realizarea aplicației mobile Canti Hub, s-a utilizat limbajul dart și framework-ul flutter însoțit de alte librării. Iar pentru realizarea software-ului dispozitivului Canti 1, s-a utilizat limbajul C++ și framework-ul arduino. Surprinzător este faptul că pentru partea de sistem încorporat s-au utilizat mai multe șabloane de proiectare de cât pentru realizarea aplicației mobile. Testele efectuate asupra elementelor sistemului și asupra sistemului în întregime au fost teste manuale.

În urma estimărilor costurilor pentru realizarea proiectului s-a ajuns la cifra de 91704 de lei. Durata de efectuare a sarcinilor în total este estimată la 1320 de ore, ceea ce pare a fi prea mult efort pentru un simplu proiect de licență.

1 ANALIZA DOMENIULUI DE STUDIU

Monitorizarea mediului cuprinde metode și strategii pentru găsirea, analizarea și stabilirea parametrilor mediului. Desfășurarea activității de monitorizare a mediului poate și în cele mai multe cazuri va găsi devieri ale parametrilor mediului. Apoi pot fi luate măsuri pentru a aduce parametrii mediului la valori normale [1]. Factorii de bază pe care se concentrează această disciplină sunt:

- aerul;
- apa;
- solul;
- gălăgia;
- biodiversitatea.

Dintre factorii enumerați mai sus, aerul și gălăgia sunt factorii de interes pentru acest proiect. Aceștia au un impact mare asupra sănătății umane și sunt mai ușor de monitorizat de ceilalți factori.

Dintre tendințele tehnologice în domeniul de monitorizare al mediului cele mai relevante pentru tehnologiile informaționale sunt teledetecția și internetul lucrurilor. Teledetecția presupune utilizarea de sateliți, drone, avioane sau alte dispozitive pentru a capta imagini sau a efectua măsurători ale suprafeței sau atmosferei pământului.

Internetul lucrurilor este rețeaua de obiecte fizice interconectate, cum ar fi senzori, dispozitive sau mașini, care pot comunica și schimba date [2]. Avantajul IoT față de teledetecție este că de obicei aceste dispozitive sunt mai ieftine și permit utilizarea lor pe o scară largă. Aceste dispozitive pot monitoriza în timp real mediul în care sunt amplasate. Internetul lucrurilor implementat pe scară largă permite colectarea unei cantități mari de date și luarea de decizii cu o precizie mare. Această tehnologie se axează în principal pe interacțiunea între dispozitive însă ca rezultat final se urmărește interacțiunea oamenilor cu aceste dispozitive. Problemele principale cu care se confruntă acest domeniu sunt satisfacerea cerințelor non funcționale și anume asigurarea interoperabilității, scalabilității, fiabilității, securității, confidențialității și eticii dispozitivelor și sistemelor IoT. Interoperabilitatea se referă la capacitatea diferitelor dispozitive și sisteme de a comunica și de a lucra împreună. Scalabilitatea se referă la capacitatea de a gestiona un număr mare de dispozitive și date fără a compromite performanța sau calitatea. Fiabilitatea se referă la capacitatea de a funcționa corect fără erori sau defecțiuni. Securitatea se referă la capacitatea de a proteja dispozitivele și datele împotriva accesului neautorizat sau atacurilor. Confidențialitatea se referă la capacitatea de a respecta drepturile și preferințele utilizatorilor cu privire la datele lor personale. Etica se referă la respectarea valorilor morale în timpul proiectării, realizării sau exploatării unui sistem IoT [3]. Conceptele fundamentale ale IoT sunt: date; dispozitive (lucruri); analiza și conectivitatea. Dispozitivele sunt utilizate pentru a colecta și posibil chiar și analiza datele. Dacă dispozitivul nu este responsabil pentru analizarea și luarea de decizii atunci prin intermediul unui protocol acesta va transmite datele și va aștepta ca decizia să fie luată pentru el.

1.1 Importanța temei

Ca consecință a evoluției, oamenii au devenit o specie vulnerabilă în fața mediului [4], pe lângă oameni, în cadrul acestei liste pot fi incluse o mulțime de animale domestice, dintre care cel mai penibil este considerat Chihuahua. Deși depășirea acestei condiții este necesară pentru bunăstarea populației umane, monitorizarea mediului rămâne un aspect important în prevenirea deteriorării sănătății. Calitatea parametrilor mediului influențează direct sănătatea oricărui organism viu. Spre exemplu, anual 9 milioane de oameni mor din cauza poluării [5]. Majoritatea țărilor nu depun eforturi considerabile pentru a rezolva această problemă [5]. Poluare nu poate fi evitată prin izolare de la lumea externă. Fiind că populația umană deja face acest lucru, populația globală petrece în jur de 90% în interior [6]. Prin urmare nivelul poluării în interior nu este mai mic ca nivelul de poluare în exterior, ba chiar poate fi mai mare așa cum morțile cauzate de poluare sunt în creștere [5] iar oamenii își petrec din ce în ce mai mult timp în interior.

O soluție evidentă, pentru a reduce efectele negative ale poluării asupra sănătății, este de a petrece mai mult timp în natură. Filosoful Friedrich Nietzsche a spus: “O viață sedentară este adevăratul păcat împotriva Duhului Sfânt.”. Acesta a ameliorat simptomele bolii de care suferea, făcând drumeții în natură.

Însă există posibilitatea ca mediul în care are loc odihna să fie poluat și prin urmare să nu aducă beneficii sănătății. Prin urmare monitorizarea mediului, este un instrument indispensabil pentru protejarea sănătății. Monitorizarea în timp real, cu feedback este și mai bună, așa cum nu este necesar de a investiga constant dacă starea mediului este satisfăcătoare. În caz că un parametru nu este corespunzător normelor, sistemul va înștiința utilizatorul.

1.2 Sisteme similare cu proiectul realizat

Pe piață există diferite sisteme de monitorizarea a mediului pentru diferite necesități. În această secțiune au fost identificate 3 sisteme cât mai portabile care să poată monitoriza parametrii de interes. Aceste 3 sisteme sunt:

- Atmotube PRO [7];
- Tempe [8];
- DT-8820 [9].

Calitățile sistemelor identificate care au fost analizate în detaliu, pentru a vedea care sunt punctele tari și slabe ale acestor sisteme sunt: numărul de parametri măsurați; conectivitatea; dimensiunile; durata de viață a bateriei și prețul.

Atmotube PRO, reprezentat în figura 1.1 este o stație meteo portabilă capabilă și să monitorizeze următorii parametri:

- particule PM1, PM2.5, PM10;
- indexul de calității al aerului (bazat pe compuși volatili organici);
- temperatura aerului;

- umiditatea relativă;
- presiunea barometrică.

Pentru a accesa datele colectate de acest dispozitiv, este nevoie de utilizat aplicația mobilă oferită de dezvoltator. Datele sunt transferate prin intermediul Bluetooth. Dimensiunile dispozitivului sunt 86 x 50 x 22 mm. Spațiul ocupat de acest sistem este prea mare pentru o purtare confortabilă, așa cum pe lângă dispozitivul în sine, va mai fi nevoie de purtat și un telefon mobil, care tinde să fie din ce în ce mai mare. Durata de viață a bateriei este de 10 zile, în ciuda faptului că dispozitivul are un volum mare. Dispozitivul poate fi procurat cu 3400 de lei. Ceea ce îl face inaccesibil pentru un număr mare de persoane, în special din teritoriul Republicii Moldova.



Figura 1.1 - Atmotube PRO

Garmin Tempe, reprezentat în figura 1.2 este un mic senzor de temperatură. Din păcate datele colectate de acesta pot fi accesate doar prin protocolul ANT+, de dispozitive care susțin acest protocol. În special aceste dispozitive sunt ceasuri fitness. Dimensiunile acestui senzor sunt surprinzător de mici 18 x 12 x 18 mm. Durata de viață a bateriei este estimată a fi un an. Acest dispozitiv costă 800 de lei, însă pentru a îl putea utiliza apar costuri suplimentare, fiind că este nevoie de un dispozitiv aparte care să poată afișa datele colectate.



Figura 1.2 - Garmin Tempe

DT-8820, reprezentat în figura 1.3 este un multimetru capabil să măsoare intensitatea luminii, intensitatea sunetului, temperatura aerului și umiditatea relativă. Spre deosebire de celelalte dispozitive, DT-8820 dispune de un ecran pentru a afișa datele măsurate. Și chiar are și butoane fizice pentru a selecta modul al acestuia de lucru, așa cum acesta nu dispune de conectivitate cu alte dispozitive. Dimensiunea acestui dispozitiv 251 x 64 x 40 mm, ceea ce îl face cel mai puțin portabil dispozitiv din această listă. Acest dispozitiv utilizează baterii Kroma de 9V, iar durata de viață este estimată a fi de un an. Prețul acestui dispozitiv este de 1200 de lei. Pentru numărul de parametri care îi poate măsura pare a fi un preț mai rezonabil decât celelalte dispozitive.



Figura 1.3 - DT-8820

În urma comparării acestor dispozitive s-a stabilit că, caracteristicile unui sistem dorit ar fi ca acesta să poată măsura câți mai mulți parametri posibil, dar cel puțin temperatura, umiditatea relativă, calitatea aerului, nivelul de gălăgie și nivelul de iluminare. Ideal prețul acestui sistem ar trebui să fie cât mai accesibil. Cel puțin să nu depășească prețul de 600 de lei.

De la un astfel de sistem se urmărește în special ca dimensiunile acestuia să fie mici, cât mai aproape posibil de dimensiunile a Garmin Tempe de 18 x 12 x 18 mm. Și de asemenea cu o durată de viață a bateriei mare, de cel puțin 30 de zile. Dispozitivul trebuie să posede conectivitate prin Bluetooth pentru conectarea la un telefon mobil și chiar și suport pentru Wi-Fi pentru a transmite datele către un server MQTT.

1.3 Analiza tehnologiilor

Un sistem IoT tipic are mai multe niveluri. Aceste niveluri depind unul de altul, importanța lor pentru sistem fiind în ordinea enumerării [10]:

- hardware dispozitiv;
- software-ul dispozitivului;
- protocoale de comunicare;
- platformă în nori.

La nivelul de hardware pe lângă mulțimea de senzori și mecanisme care vor fi utilizate, la baza dispozitivului se află un procesor. Opțiunile disponibile sunt: FPGA, Microcontroler sau ASIC [11]. Ce ține de dezvoltarea dispozitivului în caz că se optează pentru FPGA sau ASIC, va fi nevoie de scris cod în Verilog sau VHDL ceea ce este diferit de programarea clasică fiind că acest cod nu se execută secvențial pe un procesor dar este transformat în circuite care vor funcționa sincron sau asincron, după necesitate. Dacă se optează pentru microcontroler, atunci acesta poate fi programat cu limbaje precum C sau C++ dar și altele. Compararea acestor platforme hardware este prezentată în tabelul 1.1, fiecare tehnologie are avantajele și dezavantajele sale. Pentru un sistem care va fi produs pe scara mică, cel mai potrivit în majoritatea cazurilor este un microcontroler, așa cum acesta este mai ieftin de procurat în cantități mici, oferă suficientă flexibilitate în utilizarea acestuia însă este mai ușor de utilizat ca celelalte platforme.

Tabelul 1.1: Comparare FPGA, microcontroler, ASIC [11]

Criteriu	FPGA	Microcontroler	ASIC
Cost (în cantități mici)	moderat	mic	mare
Cost (în cantități mari)	moderat	mic	mic
Viteza de procesare	rapid	moderat	foarte rapid
Consumul de energie	moderat	mic	mic
Flexibilitatea	mare	mică	zero
Ușurința de utilizare	moderat	ușor	dificil

Software-ul dispozitivului poate fi scris pentru a rula direct pe hardware, sau pentru a utiliza un sistem de operare, de cele mai multe ori, un sistem de operare în timp real [12]. Fiecare abordare are avantajele și dezavantajele sale. Avantajul scrierii codului fără OS este că performanța acestuia va fi mai mare. Avantajul utilizării unui RTOS este că acesta va garanta procesarea evenimentelor în timpul dorit. O funcționalitate importantă pe care o poate avea software-ul dispozitivului este programarea OTA, care permite actualizarea software-ului dispozitivului la distanță [10]. Ce ține de limbajele de programare care pot fi utilizate pentru a scrie acest software, nu există nici o limitare, pot fi utilizate atât limbaje compilate cât și limbaje interpretate. Din limbajele de programare compilate C este cel mai popular, estimându-se că acesta este utilizat în 80% din cazuri. C++ este utilizat din cauza că compilatorul suportă un standard mai nou de cât compilatorul de C. Dintre limbajele mai tinere care au început să fie populare și în domeniul sistemelor încorporate, însă de obicei nu în producție, sunt Rust și Zig [13]. Dintre limbajele interpretate, cele mai utilizate sunt Python și Lua.

Protocoalele de comunicare sunt aceleași ca și în cadrul altor sisteme. Pot fi identificate două categorii principale, protocoale de rețea și protocoale de date [14]. Dintre protocoalele de rețea cele mai utilizate sunt: Wi-Fi; Bluetooth; ZigBee; LoRaWAN. Dintre protocoalele de date cele mai utilizate sunt: AMQP; MQTT; HTTP; CoAP; DDS; LwM2M. La acest nivel nu este dificil de creat protocoale de date proprii.

Platforma în nori este utilizată pentru a prelua, prelucra, analiza și stoca datele. Printre platformele în nori populare se numără: Amazon Web Services; Microsoft Azure. Pe acest tip de platformă dezvoltatorii pot să ruleze aplicațiile lor proprii. Însă există și platforme în nori care oferă servicii specializate, cum ar fi platforma Adafruit IO care oferă servicii de broker MQTT.

1.4 Parametrii mediului

Valorile de referință pentru parametrii mediului au fost extrase din HG353 din 2010 [15] și regulamentul sanitar privind normativele de emitere a zgomotului [16], însă acesta nu a fost încă aprobat. Aceste regulamente au ca scop asigurarea condițiilor de trai inofensive pentru populație și se axează pe

parametrii în interior, unde oamenii își petrec ce-a mia mare parte a vieții. În continuare vor fi prezentate normele extrase din aceste surse în formă de tabel.

În tabelul 1.2 sunt prezentate limitele termice minime și maxime admise pentru diferite rate metabolice. Pentru a obține metabolismul în wați este nevoie de a converti cantitatea de Cal utilizată în Joule, acest lucru se face înmulțind Cal cu 4184, apoi pentru a obține wați este nevoie de împărțit energia la numărul de secunde în care au fost consumate kaloriile [17]. De asemenea din acest regulament mai poate fi evidențiat faptul că în perioada rece a anului, se recomandă ca temperatura în interior să fie cu, cel puțin 3 °C mai mare ca în perioada caldă.

Tabelul 1.2 – Limitele termice minime și maxime admise la posturile de lucru [15]

Metabolismul, (M) W	Temperatura aerului minimă, °C	Temperatura aerului maximă, °C
$M \leq 117$	18	32
$117 < M \leq 234$	16	29
$234 < M \leq 360$	15	26
$360 < M \leq 468$	12	22
$M > 468$	12	18

Indiferent de anotimp se recomandă ca umiditatea relativă în încăperi să fie menținută între 30 % și 70 % [15]. Umiditatea ridicată amplifică sensibilitatea la temperaturi reci, de aceea se recomandă ca dacă umiditatea aerului este mare, temperatura aerului să menținută la o valoarea mai ridicată.

În tabelul 1.3 sunt prezentate nivelurile de iluminare recomandate, pentru lucrărilor cu diferite caracteristici. Aceste recomandări sunt potrivite atunci când contrastul între detaliul observat și fond este mediu și când fondul nu este nici luminos nici întunecat. În timpul somnului se recomandă să nu fie lumină, aceasta poate cauza probleme cu vederea pe termen lung

Tabelul 1.3 – Nivelul de iluminare recomandat, pentru contrastul mediu între detaliu și fond [15]

Caracteristicile lucrărilor vizuale, pentru distanța de privire de 344 mm	Nivelul normat de iluminare, lx
Lucrări de precizie deosebită cu detalii sub 0,1 mm	1500
Lucrări de precizie foarte mare cu detalii detalii între 0,1 mm și 0,3 mm	750
Lucrări de precizie mare cu detalii între 0,3 mm și 0,5 mm	500
Lucrări de precizie medie cu detalii între 0,5 mm și 0,8 mm	300
Lucrări de precizie mică cu detalii între 0,8 mm și 1,2 mm	200

În tabelul 1.4 sunt prezentate nivelurile sonore maxime admisibile în încăperile locative, conform legii, după ora 22 este nevoie ca zgomotele emise să nu depășească 45 dBA. Aceste date sunt caracteristice pentru apartamente, pentru cămine spre exemplu este permis ca nivelul zgomot să fie mai mare cu 5 dBA în ambele cazuri.

Tabelul 1.4 – Nivelurile sonore maxime ale zgomotului admise în încăperile locative [16]

Intervalul timpului de evaluare	Nivelurile sonore maxime, dBA
7.00-22.00	55
22.00-7.00	45

În cazul când vor fi monitorizați și alți parametri este important de identificat normele sau legile care reglementează acești parametri, și a le utiliza ca referințe în sistem. Așa cum pare imposibil ca oricare parametru să nu joace un rol în bunăstarea organismului uman, indiferent cât de nesemnificativ pare acesta.

2 SCOPUL, OBIECTIVELE ȘI CERINȚELE SISTEMULUI

Problema întâmpinată în ceea ce privește monitorizarea mediului constă în faptul că majoritatea sistemelor existente pentru măsurarea unei varietăți mari de parametri mai degrabă sunt niște multimetre decât niște dispozitive portabile, care să funcționeze autonom și doar să înștiințeze utilizatorii în caz de pericol pentru sănătate. Deși există sisteme care ar face parțial acest lucru, ele oferă un număr limitat de parametri pe care îi pot monitoriza, sau nu oferă suficiente funcționalități.

Prin dezvoltarea unui sistem portabil pentru monitorizarea factorilor mediului se urmărește scopul de a identifica dacă mediul din proximitatea utilizatorului prezintă un risc pentru sănătatea acestuia. După cum a fost stabilit în capitolul precedent, temperatura, umiditatea, gălăgia, iluminarea dar și alți parametri poate afecta sănătatea umană, de aceea este recomandat de a îi menține în normele stabilite. Pe lângă normele identificate există și alte standarde ce definesc valorile optime pentru majoritatea parametrilor, în special cei care pot fi controlați în interior. Este râvnită monitorizarea unui număr cât mai mare de parametri ai mediului. Însoțit de o portabilitate ridicată și de o autonomie mare a bateriei. Însă din cauza constrângerilor de resurse materiale, dimensiuni ale viitorului sistem și autonomiei acestuia se va opta ca acest sistem să poată monitoriza cel puțin:

- temperatura;
- calitatea aerului;
- nivelul de iluminare;
- nivelul de gălăgie.

Pentru a realiza acest proiect, este nevoie de realizat o mulțime de sarcini, aceste sarcini pot fi grupate în următoarele obiective:

- analiza și validarea cerințelor sistemului;
- dezvoltarea dispozitivului de monitorizare a mediului;
- monitorizarea parametrilor doriți;
- asigurarea portabilității dispozitivului;
- dezvoltarea aplicației mobile;
- stabilirea conexiunii între dispozitiv și aplicația mobilă;
- implementarea alarmelor;
- crearea unei interfețe de utilizator minimalistă;
- monitorizarea mediului în timp real;
- minimizarea costurilor de producție;
- testarea sistemului în condiții reale;
- estimarea costurilor;
- documentarea și raportarea rezultatelor.

După cum a fost stabilit sistemul de interes este constituit din două elemente, dispozitivul și aplicația mobilă. Aceste elemente la rândul lor pot fi considerate sisteme, așa cum se dorește ca ele să poată funcționa independent unul față de altul într-o oarecare măsură. Un aspect important al sistemului este autonomia față de conexiunea la internet, sistemul nu trebuie să depindă de aceasta pentru a funcționa, această problemă a fost identificată în cadrul aplicației mobile de la Garmin. În subcapitolele ce urmează, vor fi expuse cerințele funcționale ce descriu comportamentul și cerințele non funcționale ce descriu calitățile, pentru sistemul în ansamblu. Cerințele pentru fiecare element în parte pot fi identificate din aceste cerințe, însă specificarea lor în parte nu oferă multe beneficii proiectului, așa cum acesta nu este foarte complex.

2.1 Cerințe funcționale

CF1: Sistemul trebuie să fie capabil să monitorizeze temperatura aerului, umiditatea relativă a aerului, calitatea aerului, intensitatea luminii și nivelul de zgomot. Pe lângă acești parametri trebuie să se monitorizeze și nivelul bateriei.

CF2: Eșantionarea datelor trebuie să respecte o recurență predefinită, implicită de 1 minută dar configurabilă. Configurarea recurenței trebuie să poată fi efectuată în timpul operației sistemului. Limita de jos admisă a recurenței este de 30 secunde iar limita de sus echivalentul a 24 de ore în secunde.

CF3: Persistența datelor trebuie să fie asigurată local, fără servicii de la al furnizori terți. Baza de date utilizată trebuie să fie SQLite. Sistemul trebuie să ofere posibilitatea de a exporta datele, fie în CSV sau altele.

CF4: Sistemul trebuie să ofere posibilitatea de a seta alarme. În figura 2.2 este prezentată interfața de utilizator care permite configurarea alarmei selectate. Interfața de utilizator trebuie să permită selectare unui interval, iar alarma va fie activată dacă valoarea actuală este în interiorul sau exteriorul intervalului în dependență de setare.

CF5: Sistemul trebuie să ofere posibilitatea de a vizualiza a datelor prin mai multe reprezentări grafice. Este necesară reprezentarea fiecărui parametru într-o listă sau grilă. Este binevenită prezentarea evoluției parametrilor în timp pe grafic.

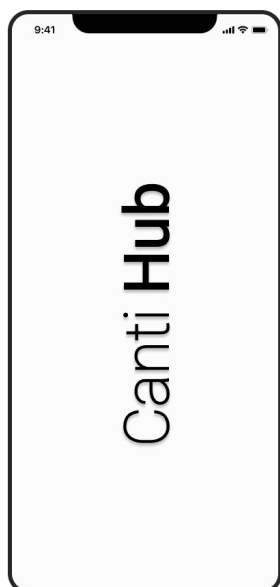
CF6: Componentele sistemului, adică dispozitivul și aplicația mobilă trebuie să comunice printr-un protocol propriu, care să funcționeze cu ajutorul tehnologiei Bluetooth.

CF7: Sistemul trebuie să fie inter-operabil cu alte sisteme existente. Pentru realizarea acestei funcționalități sistemul va trebui să ofere o interfață compatibilă cu protocolul MQTT. În configurarea conexiunii.

CF8: Interfața de utilizator trebuie să fie minimalistă și să nu distragă utilizatorii din scopurile lor pentru a utiliza sistemul. Prin urmare nu se admit reclame sau alte distracții. În figura 2.1 și 2.2 este prezentat un prototip inițial al paginilor aplicației mobile, realizat în Figma. La lansarea aplicației, va fi afișată pagina de pornire pe care este scrisă denumirea aplicației mobile – Canti Hub. În timp ce această

pagina este afișată în fundal vor fi încărcate datele din baza de date. Când această activitate se finisează aplicația va afișa pagina principală.

Pagina de pornire



Pagina principala



Setari

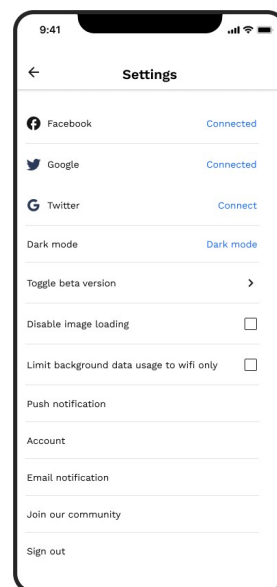
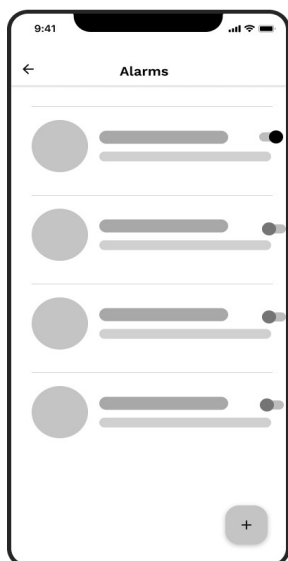


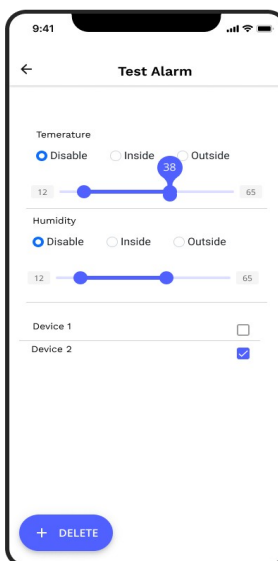
Figura 2.1 - Modelul paginii de pornire, principale și setări

Pe pagina principală în funcție numărul de dispozitive conectate la aplicația mobilă vor fi afișate diverse lucruri. În cazul în care există mai multe dispozitive conectate, aplicația va prezenta o pagină de pornire cu o vizualizare succintă pentru toate dispozitivele conectate, oferind astfel o imagine de ansamblu a stării acestora. La apăsarea pe un parametru, trebuie să fie afișat un grafic liniar cu evoluția acestuia în timp. Pe pagina principală, utilizatorii vor avea posibilitatea de a conecta noi dispozitive, facilitând astfel extinderea rețelei de monitorizare. De asemenea, în partea de sus stângă a interfeței va exista butonul de setări care deschide pagina respectivă.

Alarmer



configurare alarma



Pagina cu detalii

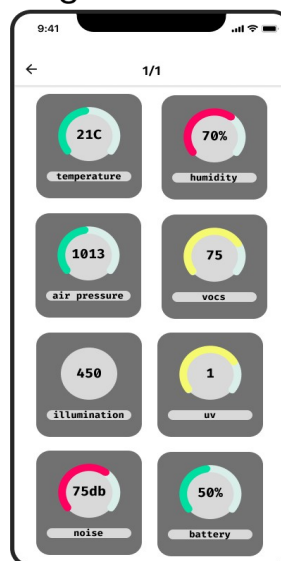


Figura 2.2 - Modelul paginii pentru alarme, configurări și detalii

Pe pagina de setări vor putea fi adăugate conexiuni MQTT sau WiFi. De asemenea va putea fi configurată recurența parametrilor. De pe pagina principală poate fi accesată și pagina de alarme, în care vor putea fi create, activate, dezactivate, editate și șterse alarmele. În cadrul fiecărei alarme poate fi selectat unul sau mai mulți parametri, iar pentru fiecare parametru poate fi selectat un interval de activare al alarme fie acesta interior sau exterior.

CF9: Aplicația mobilă trebuie să ofere atât tema luminoasă cât și tema întunecată, iar utilizatorul să poată să o configureze independent de tema sistemului dacă dorește.

CF10: Aplicația mobilă trebuie să fie tradusă în limba Română, Engleză și Rusă.

2.2 Cerințe non-funcționale

CNF1: Sistemul trebuie să aibă o autonomie ridicată. Funcționarea sistemului nu trebuie să depindă de conexiunea la internet. Sistemul trebuie să aibă o durată de viață a bateriei de 30 de zile fără a necesita reîncărcarea dispozitivului. În plus, sistemul trebuie să poată detecta și trata potențialele erori pentru a rămâne funcțional.

CNF2: Sistemul trebuie să fie adaptabil. Trebuie să poată suporta diferite versiuni de hardware și software pentru dispozitivele conectate, iar acestea trebuie identificate și gestionate corespunzător.

CNF3: Codul trebuie să fie cât ușor de portat pe alte platforme.

CNF4: Prețul de producție al unui dispozitiv trebuie să fie minim, prețul unei unități nu ar trebui să depășească 400 de lei.

CNF5: Dimensiunile dispozitivului nu trebuie să depășească dimensiunile de 86 x 50 x 22 mm.

CNF6: Realizarea sistemului trebuie să fie versionată cu ajutorul instrumentului Git. Să se respecte convenția de mesaje **conventional commit**.

CNF7: Sistemul trebuie să fie Open Source și Open Hardware. Fișierele sursă trebuie să fie găzduite pe platforma Github. Dacă este posibil, să se implementeze pipeline-uri de CI și CD pentru firmware-ul dispozitivului și aplicația mobilă.

CNF8: Realizarea sistemului va avea loc după modelul de dezvoltare iterativ – agile.

CNF9: Sistemul va fi inter-operabil cu platforme MQTT. Datele vor putea fi preluate și transmise către platforme MQTT introduse de utilizator.

3 MODELAREA ȘI PROIECTAREA SISTEMUL

Pentru modelarea sistemului, au fost considerate mai multe limbaje de modelare. Acestea fiind: UML, SysML, Archimate, AADL și OPM. La prima vedere AADL [18] părea a fi o alegere mai bună, considerând că sistemul va avea și partea hardware. Acesta ar fi permis abordare de inginerie a sistemelor bazată pe modele, care permite efectuarea simulărilor și identificarea posibilelor erori înainte de realizarea sistemului. Totodată acest limbaj oferă atât limbaj textual cât și grafic. Însă din cauza că pe internet este un număr restrâns de resurse pentru a învăța nu s-a optat pentru a îl utiliza. A doua ce-a mai bună abordare ar fi de a modela sistemul pe foi de hârtie utilizând un pix, așa cum mediul digital nu se poate compara cu mediul analog, însă această abordare nu ar fi acceptabilă în ipostaza proiectului respectiv. Ca urmare a acestor fapte s-a optat pentru utilizarea limbajului UML [19] – limbaj de modelare unificat. Deși inițial acesta era prevăzut doar pentru sistemele intensive de software, în următoarele reviziuni a fost introdusă posibilitatea de a modela sisteme complete. Deși unii încă au rămas nemulțumiți și au creat limbajul derivat SysML care se concentrează pe sisteme dar nu software.

Ca instrument pentru modelare sistemului a fost ales Modelio [20], acesta a fost ales datorită faptului că este gratis, cu, codul sursă deschis. Se instalează și rulează pe mașina locală fără a avea nevoie de acces la internet. Deși acest instrument nu este perfect și are probleme cu reprezentarea unor diagrame, faptul că oferă o experiență de utilizare intuitivă îi oferă un avantaj față de soluțiile comerciale.

În specificațiile v2.5.1 de UML [19] sunt definite 14 diagrame [21], care sunt împărțite în două tipuri, diagrame comportamentale și diagrame structurale. Aceste diagrame sunt prevăzute pentru diverse cazuri și utilizarea tuturor poate fi justificată doar pentru cele mai complexe proiecte.

Informația ce-a mai relevantă despre sistem este în diagrama de amplasare, așa cum acest sistem are și o parte hardware. Din această poate fi dedus faptul că topologia sistemului va fi stea. Însă din cauza limitărilor limbajului UML, acest lucru ar putea fi sărit cu vederea.

Deși modelarea are loc înainte de realizare, în timpul dezvoltării pot fi identificate probleme în model, sau limității tehnice. Ca urmare modelul va suferi schimbări, pentru a ilustra structura și comportamentul sistemului în realitate.

3.1 Descrierea comportamentală a sistemului

Așa cum a fost menționat, limbajul UML are un set de diagrame care descriu comportamentul sistemului. Aceste diagrame sunt:

- diagrama cazurilor de utilizare;
- diagrama de activitate;
- diagrama de secvență;
- diagrama mașină de stări;
- diagramă de comunicare;

- diagramă de interacțiune;
- diagramă de sincronizare.

Diagrama cazurilor de utilizare, descrie ce face un sistem, însă nu descrie modul de implementare. În această diagramă este prezentat un set de evenimente care pot fi declanșate de către actor. Actorul în această diagramă poate fi o persoană, un rol, sau chiar și un sistem. Prin această diagramă poate fi prezentată imaginea generală a sistemului.

Diagrama de activitate prezintă fiecare pas pentru a atinge un obiectiv în cadrul unui sistem. Această diagramă este potrivită pentru reprezentarea algoritmilor. Fiecare activitate are un punct de intrare și un punct de ieșire. Iar fiecare acțiune din interior, poate fi la rândul ei un proces.

Diagrama de secvență este potrivită pentru a arăta interacțiunea dintre mai multe obiecte sau actori. Aceasta are linii de viață din care sunt trase mesaje către alte linii. Această diagramă prezintă în mod cronologic mesajele care au avut loc între obiecte și actori.

Diagrama mașină de stări este utilizată pentru a modela comportamentul unui obiect complex. Aceasta prezintă care sunt evenimentele care pot schimba starea obiectului și ce acțiuni sunt întreprinse când o stare este activă.

Diagrama de comunicare pune accent pe comunicarea dintre obiecte. Aici sunt trase asocieri între obiecte, iar mesajele și direcțiile acestora sunt puse lângă linie. Conține aceeași informație ca o diagramă de secvență.

Diagrama de interacțiune adesea este complexă și este evitat de nespecialiști și amatori. Aceasta pe lângă elementele diagramei de activitate conține elemente cum ar fi interacțiunea, constrângeri de timp sau de durată.

Diagrama de sincronizare cunoscută și cu numele de diagramă de evenimente prezintă cronologia în care obiectele și actorii acționează. Se pune accentul pe durata evenimentelor și modificările care se produc. Însă din cauza că Modelio nu le suportă. Această diagramă nu a fost utilizată în modelarea sistemului.

3.1.1 Imaginea generală asupra sistemului

Pentru a utiliza sistemul, utilizatorul trebuie să dețină cel puțin un dispozitiv. Apoi acesta va trebui să îl înregistreze în sistem. Odată ce dispozitivul este înregistrat în sistem utilizatorul poate vizualiza datele monitorizate pe acesta. Poate export datele respective. De asemenea mai există și posibilitatea de a seta alarme. Aceasta este imaginea generală a sistemului, reprezentarea ei grafică poate fi vizualizată în figura 3.1.

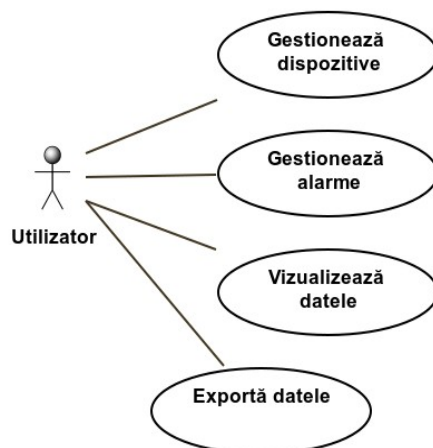


Figura 3.1 - Utilizarea sistemului

Sistemul va permite vizualizarea datelor în mai multe moduri. Cel mai de bază mod va fi fie o listă fie o grilă cu fiecare parametru monitorizat, va fi o pagină dedicată fiecărui dispozitiv. Altă reprezentare va fi o diagramă de tip arie, normalizată astfel în cât să poată fi identificat cu ușurință dacă un parametru este prea mare sau prea mic. În ultimul rând va fi prezentă posibilitatea de a vizualiza fiecare parametru în parte pe un grafic cu linii, pentru a putea vedea cum a evoluat acest parametru. Reprezentarea acestor cazuri de utilizare poate fi vizualizată în figura 3.2.

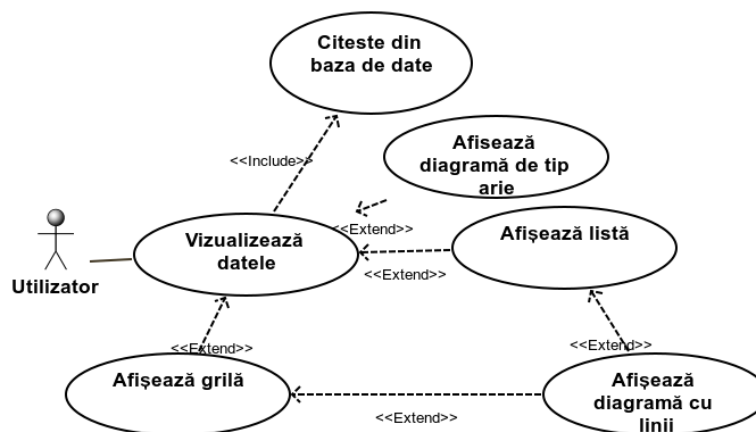


Figura 3.2 - Vizualizarea datelor

În figura 3.3 este prezentat mai detaliat modul de gestionare al sistemului. În primul rând, așa cum a fost menționat anterior, utilizatorul poate adăuga dispozitive în sistem. Totodată utilizatorul le poate șterge. Un alt aspect important al gestionării sistemului este configurarea recurenței. Utilizatorul poate configura recurența pentru fiecare parametru în parte. Respectând condiția că aceasta să se încadreze în intervalul 29 secunde, 24 ore. În cadrul acestui sistem, utilizatorul poate gestiona alarme, asta presupune crearea, ștergerea, dezactiva și activa alarmele. De asemenea el le poate edita. Toate aceste setări sunt transmise și către dispozitiv dacă Bluetooth-ul este activat și dispozitivul conectat este în raza de acțiune.

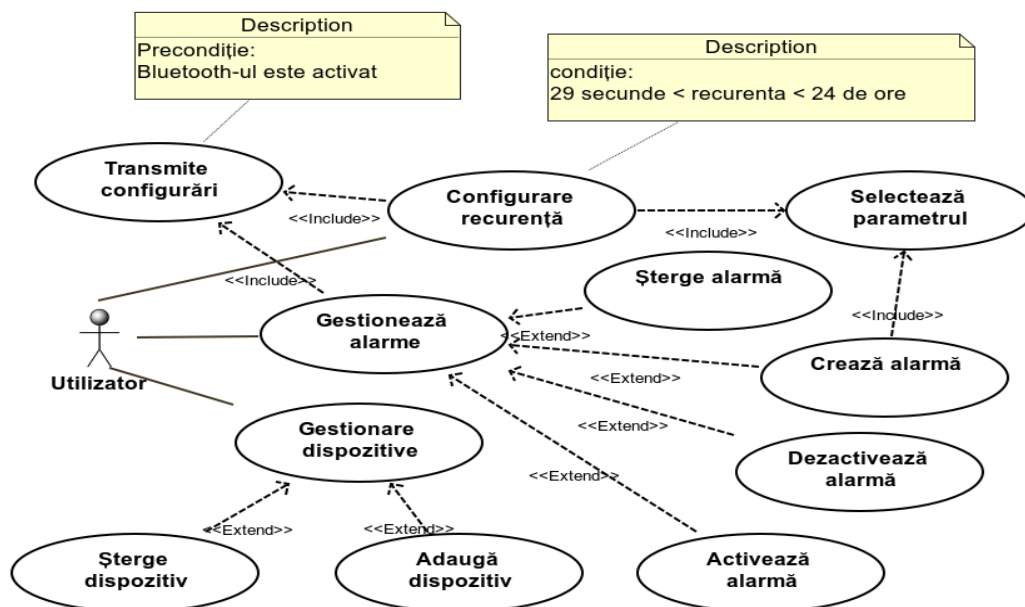


Figura 3.3 - Gestionarea sistemului

Inspectând figura 3.4, poate fi dedus că sistemul este unul inter-operabil. Există posibilitatea de a conecta sistemul la o platformă externă. Acest lucru va avea loc prin intermediul protocolului de comunicare MQTT. Aplicația va fi principalul publicator către această platformă externă. Pentru ca dispozitivul să poată transmite datele prin MQTT, acesta va trebui să fie în regim staționar. Regimul staționar presupune că dispozitivul este conectat la o sursă de alimentare.

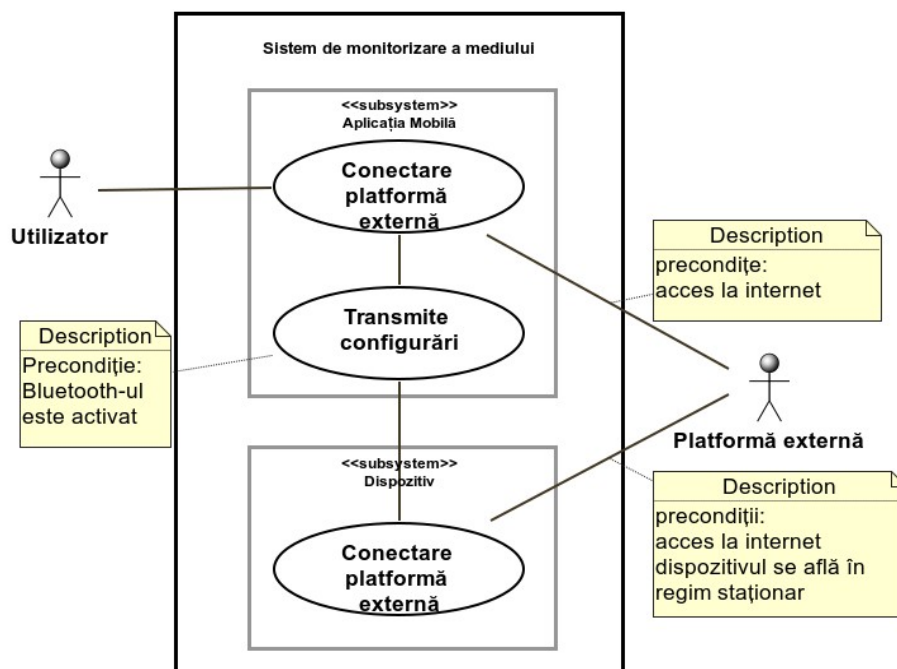


Figura 3.4 - Conectarea la o platformă externă

În figura 3.5 este prezentat modul în care software-ul sistemului poate fi actualizat. În primul rând dezvoltatorul trebuie să publice o versiune nouă de software și să o marcheze corespunzător. Apoi GitHub va compila software-ul fie că acesta este pentru dispozitiv sau telefon mobil. Odată ce utilizatorul va

verifica dacă există o nouă versiune software, aplicația îl va înștiința de acest lucru. Versiunea nouă de software va fi descărcată. Utilizatorul va trebui să instaleze manual noua versiune de aplicație mobilă, fiind că pe dispozitivele care nu sunt rulate, instalarea aplicațiilor în fundal este posibilă doar prin intermediul serviciilor google. Noua versiune de software pentru dispozitiv va fi transmisă prin Bluetooth către dispozitiv. Dacă acesta avea cel puțin 70% de baterie, actualizarea va avea loc. Dacă nu utilizatorul va fi înștiințat de acest lucru. Și îi va fi propus să pună dispozitivul la încărcat.

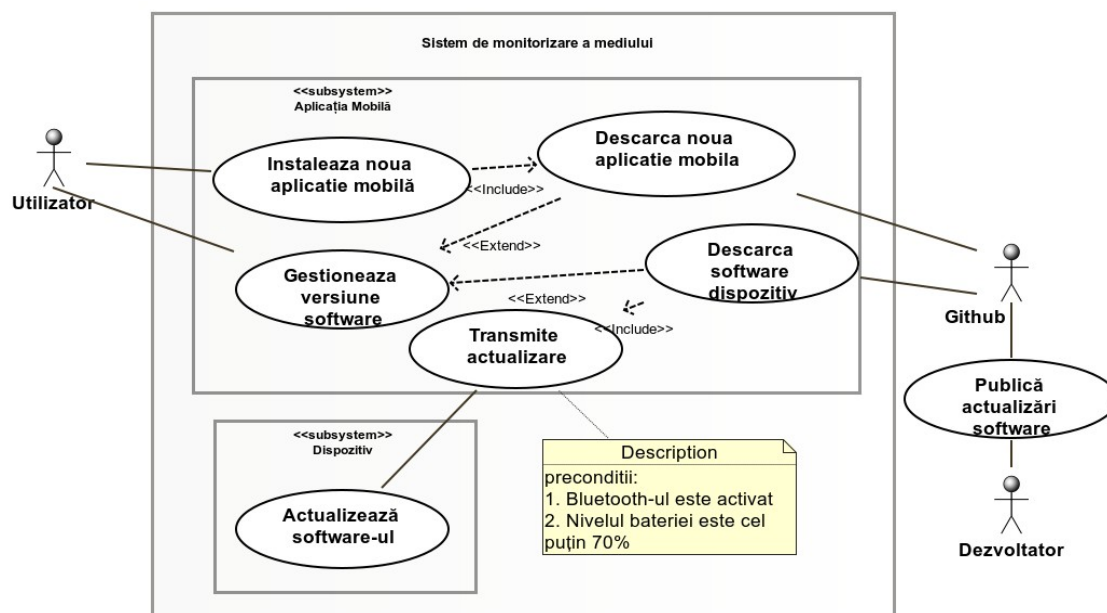


Figura 3.5 - Actualizarea versiunii software

3.1.2 Modelarea vizuală a fluxurilor

Fiecare senzor cu, care este dotat dispozitivul va fi utilizat pentru a citi unul sau mai mulți parametri. Primul pas este să fie inițializat senzorul. Aici există posibilitatea ca senzorul să fie defect, ca rezultat nu se va primi nici un răspuns, acest caz va trebui tratat ca o eroare. În cazul când senzorul este funcțional, se începe măsurarea parametrului. Se așteaptă finisarea operației. Însă în acest timp pot fi efectuate și alte operațiuni. Spre exemplu pot fi inițializați și alți senzori. Odată ce parametrul a fost convertit, software-ul îl poate citi. În figura 3.6 este reprezentat acest proces.

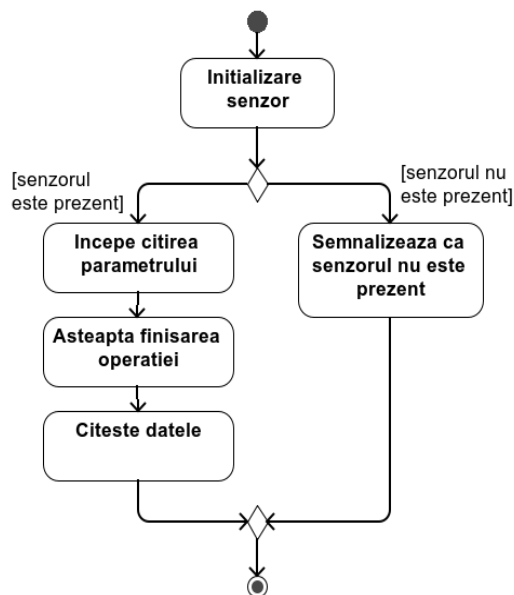


Figura 3.6 - Citirea parametrului

În figura 3.7 sunt prezentați pașii pentru a evalua alarmele. În primul rând sunt determinate care alarme sunt activate, pentru a nu genera alarme false. Asta este pe partea de aplicație mobilă, fiind că pe partea de dispozitiv va fi permis de setat doar 3 alarme, pentru a nu utiliza prea multă memorie. Următorul pas este de a identifica dacă condițiile pentru măcar o alarmă sunt îndeplinite. Condiția prezintă un interval în care valoarea parametrului curent trebuie să se încadreze sau să nu se încadreze pentru a genera alarma. Acest comportament este specificat de utilizator. Apoi dacă există măcar o alarmă care să fie activă, va fi generat un semnal de alarmă. Pe aplicația mobilă poate fi prezentată numele alarmei.

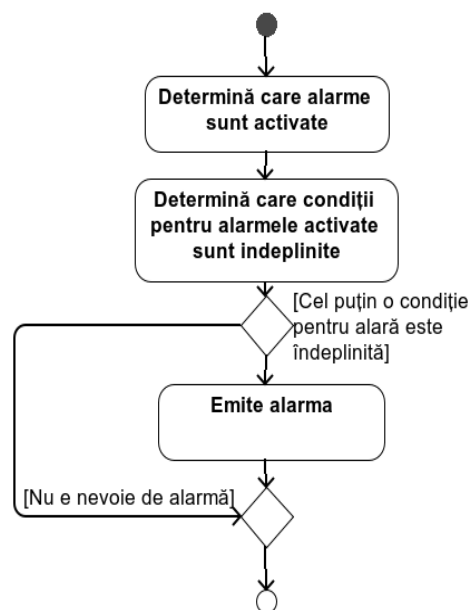


Figura 3.7 - Evaluare alarme

Exportarea datelor este un proces simplu. În primul rând datele sunt extrase din baza de date. Apoi în dependență de formatul selectat de către utilizator, datele sunt transformate într-un fișier de acest

format. Formatele necesare de implementat sunt CSV și SQLite, însă se poate de implementat și alte formate. În ultimul rând utilizatorul este întrebat unde dorește să salveze fișierul. Acest proces poate fi vizualizat în figura 3.8.

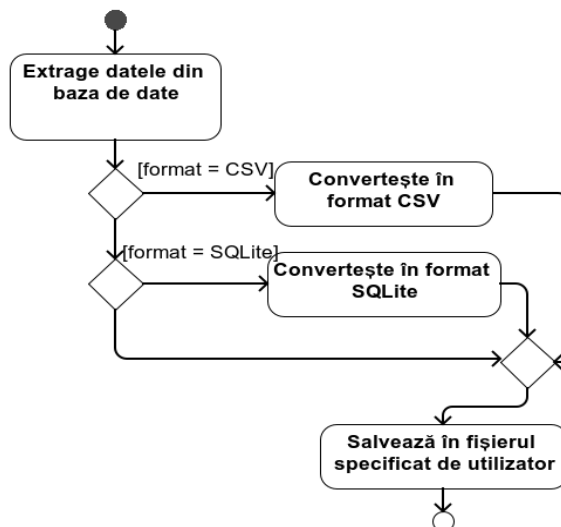


Figura 3.8 - Exportarea datelor

3.1.3 Stările de tranzație a sistemului

Dispozitivul se poate afla în patru stări principale. Stările sunt Inițializare, AutoTestare, Operație și Somn. Dispozitivul ajunge în cadrul stării de Inițializare odată ce acesta este pornit. Pornirea este fizică, prin conectarea alimentării. În cadrul acestei stări microcontrolerul va inițializa periferialele sale și senzorii. Dacă inițializarea va avea loc cu succes dispozitivul va trece în starea de AutoTestare, dacă nu acesta se va stinge. Când se află în starea de AutoTestare microcontrolerul va testa periferialele sale, integritatea senzorilor și a altor componente. În caz că testarea rezultă în erori dispozitivul se va duce înapoi în starea de Inițializare. Însă dacă mai mult de trei cicluri au fost cu eroare, dispozitivul se va stinge. Dacă testarea s-a finisat cu succes, atunci dispozitivul trece în starea de Operațiune în cadrul căreia el va colecta datele și le va transmite după posibilitate. Totodată pentru a reduce din consumul de energie, așa cum de regulă citirea unui parametru durează câteva milisecunde iar restul, cel puțin 29 de secunde nu va fi nimic de făcut, dispozitivul se va duce în starea de Somn. În aceasta majoritatea periferialelor dispozitivului sunt deconectate. Apoi după ce a expirat timpul de inactivitate, dispozitivul se întoarce în starea de Operațiune. În figura 3.9 pot fi vizualizate aceste stări în mai puține cuvinte.

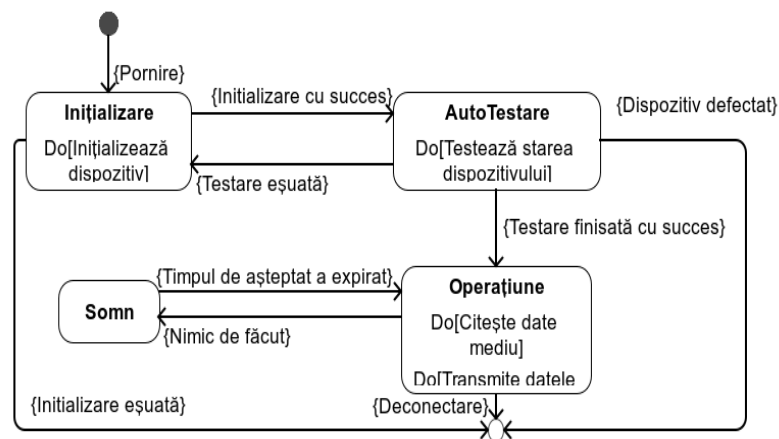


Figura 3.9 - Stările dispozitivului

În figura 3.10 sunt prezentate stările parcurse atunci când dispozitivul se află în starea de Operațiune. În primul rând în starea de Configurare se începe măsurarea datelor pe senzori. Apoi se trece în starea de așteptare, în care se așteaptă finisarea convertirilor. După ce convertirea a fost efectuată, datele pot fi citite. Apoi se trece la starea de Procesare, asta ar presupune filtrarea acestora. În caz că este posibil datele sunt transmise. Apoi dispozitivul va trece în starea de Somn.

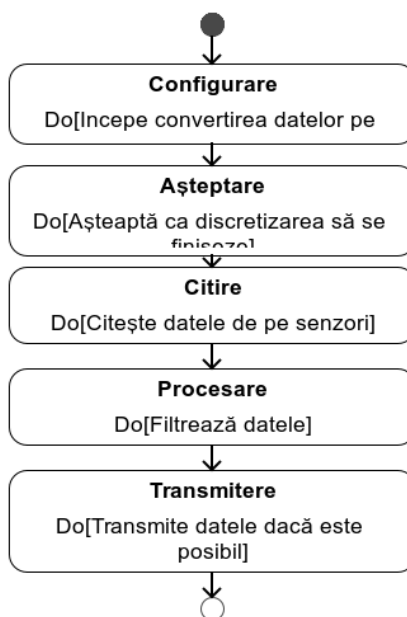


Figura 3.10 - Sub-stările etapei de operațiune

Protocolul de comunicare proiectat pentru sistem poate fi reprezentat ca o mașină de stări. Acest protocol de comunicare este unidirecțional. Aplicația mobilă este cea care va iniția comunicarea. Comunicarea va avea la bază trei servicii Bluetooth, cu câte o caracteristică. În figura 3.11 este reprezentată doar partea ce ține de serviciul și caracteristica de configurare. Celelalte două servicii sunt mai simple. În primul rând, comunicarea se începe cu o comandă. Această comandă încapă într-un octet. Dacă comanda primită fie este mai lungă, fie nu corespunde comenzilor cunoscute aceasta este ignorată. Acest protocol permite setarea și citirea recurenței parametrului dorit sau resetarea acesteia. Comanda 0xF1 inițiază accesul la

recurență. Urmată de indicele parametrului tot într-un octet. Apoi în dependență dacă următoarea acțiune va fi o scriere sau o citire a caracteristicii, asta va reprezenta și citirea sau scrierea valorii recurenței. Comanda 0xF1 urmată de indicele parametrului va reseta recurența acestui parametru, dacă acesta există. Comanda 0xA0, 0xA1 și 0xA2 inițiază setarea unei alarme. După înscrierea acesteia se scrie indicele parametrului care iar este un octet. În continuare urmează 3 valori, prima întreg pe 4 octeți, iar următoarele două numere cu virgulă mobilă pe 4 octeți. Acestea reprezintă tipul intervalului, valoarea minimă și maximă. În dependență de scrierea sau citirea caracteristicii se vor accesa acești parametri. Însă nu este posibil de făcut operații diferite. Comanda 0xEE este responsabilă pentru accesarea timpului. Timpul este în format unix pe 64 de biți. Acesta poate fi setat sau citit. Comenzile pentru activarea, dezactivarea și ștergerea alarmelor sunt 0xE{0..2}, 0xB{0..2} și 0xD{0..2}.

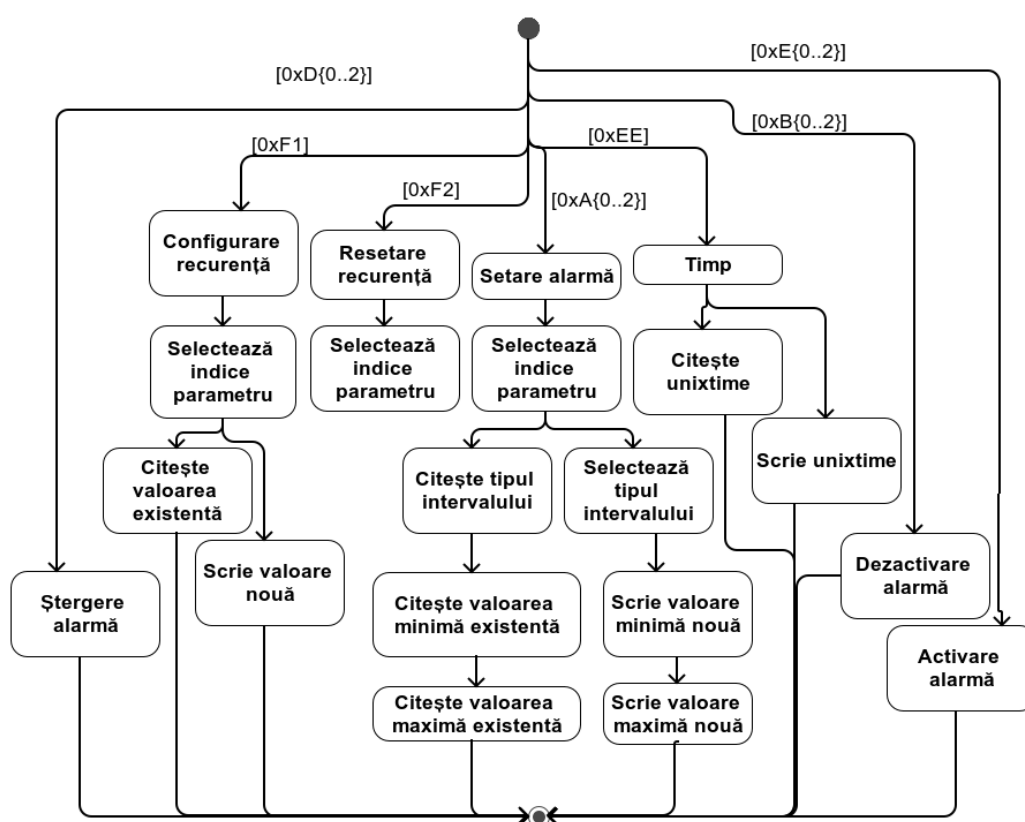


Figura 3.11 - Tranzițiile protocolului de comunicare

3.1.4 Descrierea scenariilor de utilizare a aplicației

Dispozitivul este un element autonom al sistemului. Odată ce acesta este alimentat, el inițiază senzorii și apoi începe procesul de colectare al datelor. Acest proces repetitiv presupune inițializarea măsurării parametrilor, citirea parametrilor și salvarea acestora în caz că ei nu pot fi transmiși. Când utilizatorul are nevoie de aceste date, el va accesa aplicația mobilă. Aceasta la rândul ei va obține datele de la dispozitiv și le va afișa pentru utilizator. Acest scenariu poate fi vizualizat în figura 3.11.

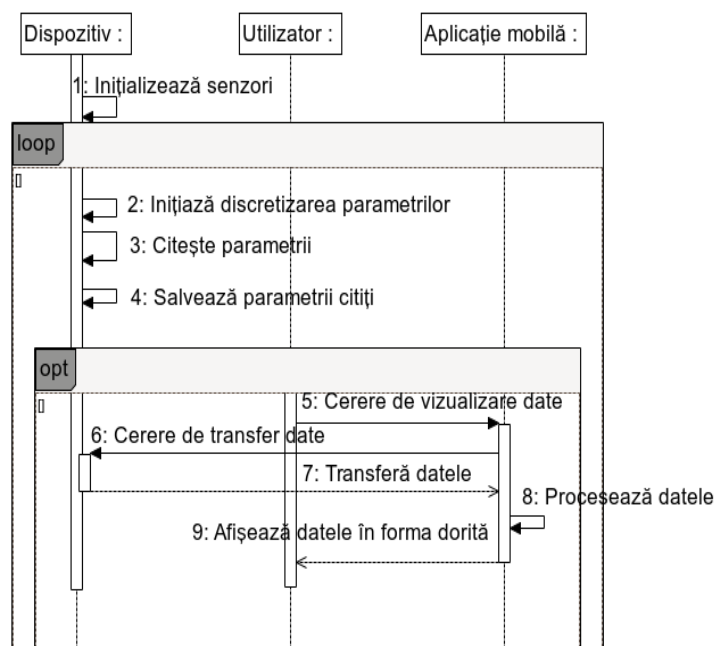


Figura 3.12 - Colectarea datelor

În figura 3.13 este ilustrat modul în care un dispozitiv nou este adăugat în sistem. Această interacțiune este inițiată de către utilizator. Utilizatorul va trebui să selecteze ce dispozitiv dorește să adauge în caz că sunt prezente mai multe dispozitive pornite dar care nu se află în nici o rețea personală. Apoi aplicația va cere, ca utilizatorul să introducă cheia de acces către dispozitiv. Aplicația mobilă va transmite cererea de conectare împreună cu cheia de acces către dispozitiv. Acesta va evalua cheia și în dependență de corectitudinea acesteia va accepta sau refuza conexiunea. Utilizatorul va fi înștiințat de rezultatul acestei operații.

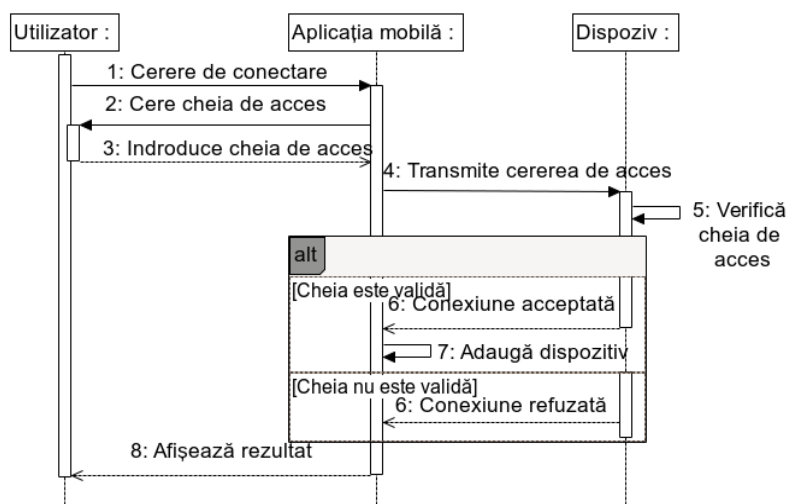


Figura 3.13 - Conectarea unui nou dispozitiv

3.1.5 Fluxurile de mesaje și legăturile dintre componentele sistemului

În figura 3.14 este prezentată colaborarea dintre componentele fizice ale dispozitivului pentru a colecta datele. În primul rând, microcontrolerul va inițializa senzorii digitali BME680 și VEML7700. Apoi când măsurarea datelor va fi finalizată va citi parametrii respectivi. De pe senzorul BME680 vor fi

citite temperatura, umiditatea, presiunea și compușii volatili organici. De pe senzorul VEML7700 va fi citit nivelul de iluminare. În ultimul rând, microfonul MAX9814 care este un senzor analog, va modifica tensiunea pe un pin, iar microcontrolerul o va citi. Apoi în urma unor calcule va fi identificat nivelul de gălăgie. Pentru a păstra datele, în cazul când nu există o conexiune Bluetooth, microcontrolerul va înscrie aceste date în regiunea de ram a RTC-ului, această regiune de RAM nu pierde alimentarea nici când microcontrolerul este în regim de somn, deci datele nu vor fi pierdute. Când va fi prezentă o conexiune, microcontrolerul va putea citi aceste date și să le transmită către aplicația mobilă.

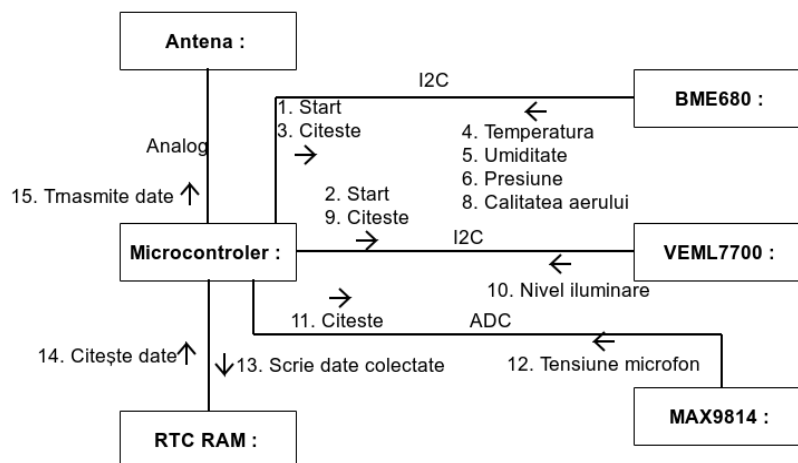


Figura 3.14 - Colectarea datelor

Pentru a adăuga un dispozitiv nou, utilizatorul trebuie să selecteze ce dispozitiv dorește să adauge în caz că sunt prezente mai multe dispozitive. Apoi aplicația va cere, ca utilizatorul să introducă cheia de acces către dispozitiv. Aplicația mobilă va transmite cererea de conectare împreună cu cheia de acces către dispozitiv. Acesta va evalua cheia și în dependență de corectitudinea acesteia va accepta sau refuza conexiunea. Utilizatorul va fi înștiințat de rezultatul acestei operații. Acest proces poate fi vizualizat sub formă de colaborare în figura 3.15.

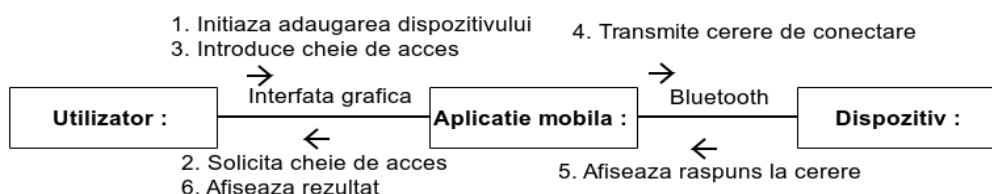


Figura 3.15 - Conectarea unui nou dispozitiv

3.2 Descrierea structurală a sistemului

Pentru a modela structura unui sistem utilizând limbajul de modelare UML, pot fi utilizate următoarele diagrame:

- diagramă de clasă;
- diagramă de obiecte;
- diagramă de componente;

- diagramă de implementare.
- diagramă de structură compozită;
- diagramă de pachete;
- diagramă de profiluri.

Diagrama de clasă este cea mai utilizată diagramă de structură în dezvoltarea software. Deși în principal este utilizată pentru a reprezenta clase, poate fi utilizată și pentru a reprezenta structuri și enumerații pentru limbajele care nu au clase cum ar fi C. Cu acest tip de diagramă poate fi prezentat modelul logic și fizic al sistemului. Reprezentarea grafică a clasei este un dreptunghi cu trei secțiuni. În prima se regăsește numele și stereotipul clasei. În următoarea secțiune sunt prezentate atributele. În ultima secțiune sunt prezentate metodele clasei.

Diagrama de obiecte este utilizată pentru a instala un exemplu al diagramei de clasă. Aceasta are ca scop să valideze corectitudinea diagramei de clasă.

Diagrama de componente prezintă grupările logice ale elementelor. Aici pot fi prezentate modulele software ale sistemului. Dar și componentele obiectelor fizice. Fiecare component este reprezentat cu ajutorul unei casete rectangulare. Conectorii sunt interfețe oferite sau cerute.

Diagrama de structură compozită este utilizată rareori în afara domeniului de dezvoltare software. Deoarece este mai detaliată decât diagrama de clase, descriind mai multe amănunte despre clase și interacțiunile dintre acestea. Pentru majoritatea cazurilor această diagramă este una în plus.

Diagrama de implementare prezintă componentele hardware și software ale sistemului. Componentele hardware sunt prezentate ca noduri iar componentele software sunt prezentate ca artefacte. Această diagramă prezintă modul de instalare al sistemului și cum componentele sistemului sunt legate.

Diagrama de pachete este utilizată pentru a prezenta dependențele dintre pachetele care compun un model. Scopul principal este de a prezenta relația dintre diferitele componente mari care formează un sistem complex.

Diagrama de profiluri este la fel de complexă ca un limbaj. Pentru această diagramă sunt create noi proprietăți și semnificații pentru diagramele UML existente. Aici se definesc noi stereotipuri. Profilurile sunt utile pentru a particulariza un model UML pentru o platformă specifică, cum ar fi Java sau .Net Framework.

3.2.1 Descrierea structurii statice a sistemului

Pentru implementarea senzorilor la nivel de driver, indiferent de limbajul care va fi utilizat, poate fi utilizat șablonul de proiectare hardware adaptor [22]. În figura 3.16 este prezentat faptul că toți senzorii vor moșteni de la clasa senzor. Aceștia vor trebui să ofere o singură metodă pentru citirea parametrilor, însă aceasta va primi tipul parametrului care va fi citit. Astfel se va respecta și principiul substituirii Liskov [23].

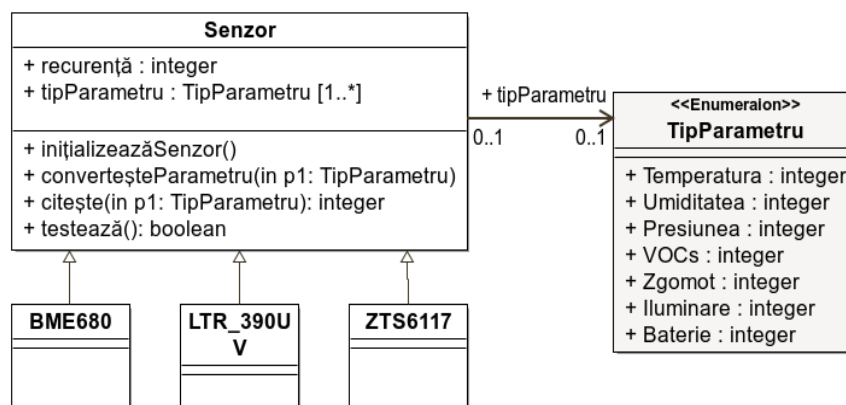


Figura 3.16 - Implementarea senzorilor

În cod va fi prezentă o clasă pentru generarea stărilor dispozitivului. În metodele acesteia se va scrie codul pentru inițializare, auto testare, operațiune și somn. Acestea vor fi implementate cu ajutorul mașinii de stări prezentată în figura 3.9. Cum poate fi observat în figura 3.17 această clasă va interacționa și cu alte clase cum ar fi clasa responsabilă de colectarea datelor, care va avea un șir de senzori. Și clasa de auto testare care va avea metode pentru a testa toate părțile componente ale dispozitivului.

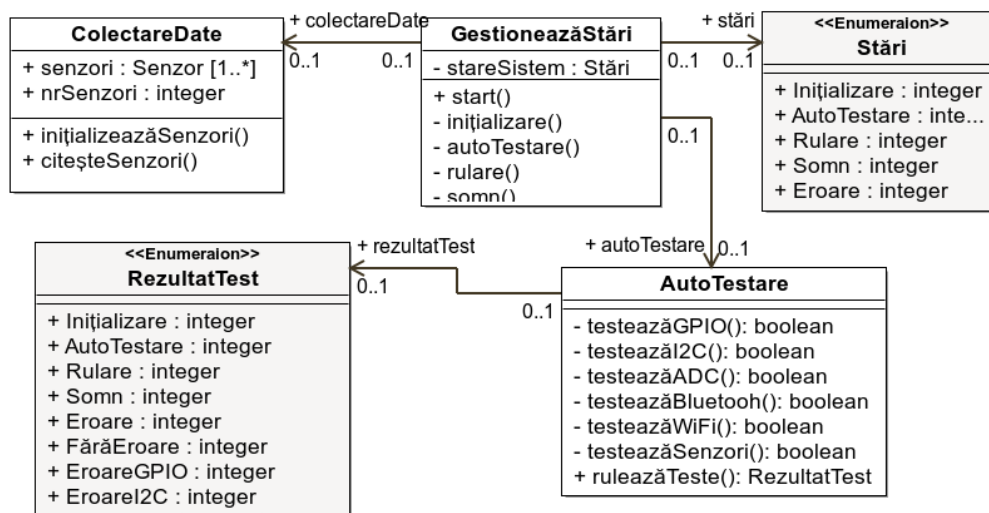


Figura 3.17 - Gestionarea stărilor dispozitivului

3.2.2 Relațiile de dependență între componentele sistemului

Componentele software ale aplicației mobile sunt prezentate în figura 3.18. Acestea sunt logica de aplicație, baza de date, setări, prezentarea și comunicarea. Nivelul de prezentare va oferi o interfață pentru afișarea datelor. Nivelul de comunicare va permite stabilirea conexiunii cu dispozitivul, transmiterea de comenzi și preluarea datelor. Baza de date va permite stocarea datelor colectate pe termen lung. Componenta de setări va permite de a salva setările utilizatorului ca acestea să nu se piardă.

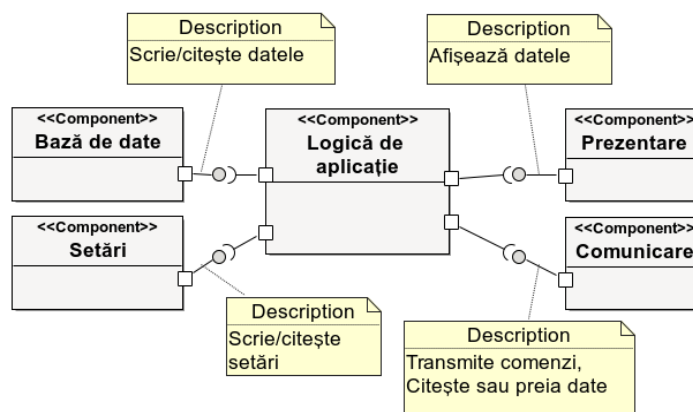


Figura 3.18 - Componentele aplicației mobile

În figura 3.19 sunt prezentate componentele software ale dispozitivului. Acestea includ următoarele componente: aplicație, persistență, comunicare, auto testare, driver VEML7700, driver BME680, driver MAX9814. Componenta de persistență va avea rolul de a stoca setările, în principal cele ce țin de recurența eșantionării parametrilor, conexiuni Wi-Fi și MQTT, dar și stocarea datelor pentru recuperarea lor ulterioară. Componenta de comunicare va permite conectarea la rețele Wi-Fi și Bluetooth, dar și transmiterea datelor prin aceste canale. Pentru transmiterea datelor prin WiFi se va folosi protocolul MQTT iar pentru transmiterea datelor prin Bluetooth se va implementa un protocol propriu. Componenta de auto testare va testa funcționalitatea dispozitivului și integritatea fiecărui senzor. Drivererele pentru fiecare senzor va permite citirea datelor oferite de aceștia dar și identificarea dacă senzorul este în stare funcțională și nu are defecte.

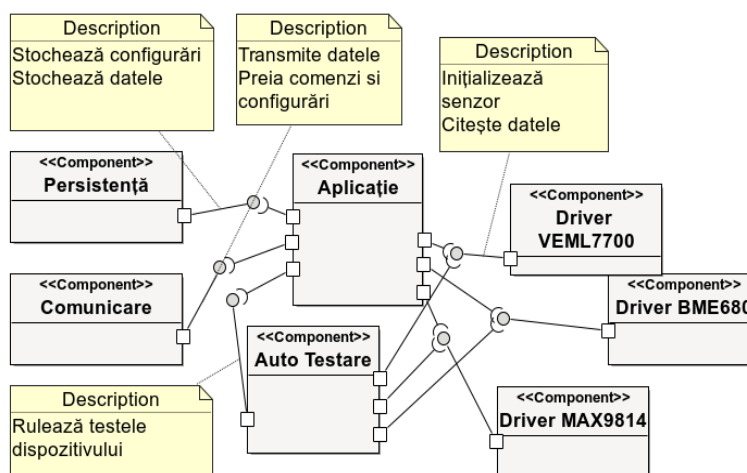


Figura 3.19 - Componentele software-ului dispozitivului

3.2.3 Modelarea echipamentelor mediului de implementare

În figura 3.20 este prezentată implementarea sistemului în ansamblu. O instanță a sistemului este constituită din două tipuri de elemente: dispozitiv și aplicație mobilă. În cadrul acestei instanțe, pot fi conectate mai multe dispozitive la o singură aplicație mobilă. Nucleul dispozitivului este microcontrolerul ESP32-C3, care va rula firmware-ul scris pentru acesta. La microcontroler vor fi conectați senzorii respectivi, BME680, MAX9814 și VEML7700 de pe care se vor citi parametrii de interes. Pe telefonul

mobil, spre deosebire de dispozitiv este instalat și un sistem de operare, care va constitui mediul de execuție al aplicației. Acest sistem de operare va permite accesul aplicației la rețeaua Bluetooth, dar și la alte funcții, cum ar fi conexiunea la internet sau afișarea unei interfețe grafice. În cadrul acestui sistem pot fi conectate mai multe dispozitive. Totodată acest sistem ar putea să transmită sau să preia date de la un broker MQTT.

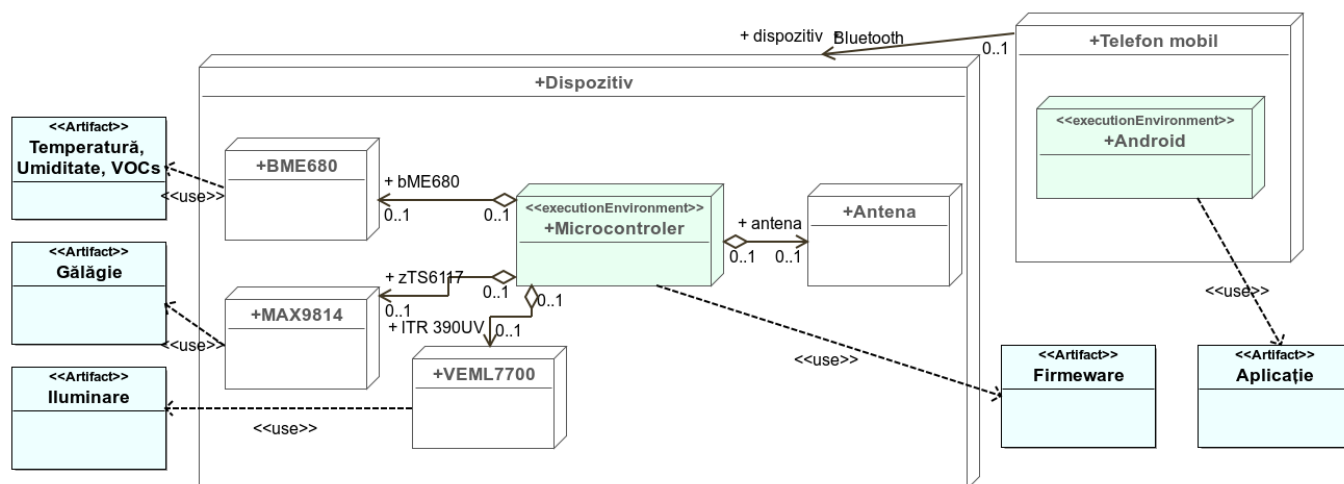


Figura 3.20 - Implementarea sistemului

În figura 3.21 este prezentată arhitectura hardware a dispozitivului. Acesta are la bază placa ESP32-C3 Super Mini de culoare roz. La aceasta, pe magistrala I2C, reprezentată cu culoarea portocalie, sunt conectați următorii senzori BME680, VEML7700 și SHT40. BME680 are capacitatea de a măsura temperatura, umiditatea relativă, presiunea atmosferică și rezistența aerului la încălzire, ceea ce poate determina compuşii volatili organici sau indexul de calitate al aerului. Totodată din presiunea atmosferică poate fi dedusă altitudinea. VEML7700 este capabil să măsoare intensitatea luminii. SHT40 este capabil să măsoare temperatura și umiditatea relativă. Acest senzor este utilizat deoarece în urma măsurării rezistenței aerului, senzorul BME680 se încălzește iar temperatura măsurată de pe acesta nu este corectă. În ultimul rând la placă este conectat și microfonul MAX9814 pe unul din pinii analogi, conectat cu linia verde. Acesta va fi utilizat pentru identificarea nivelului de gălăgie. Pentru a emite alerte va fi utilizat un buzzer, acesta în figură este conectat prin intermediul liniei negre.

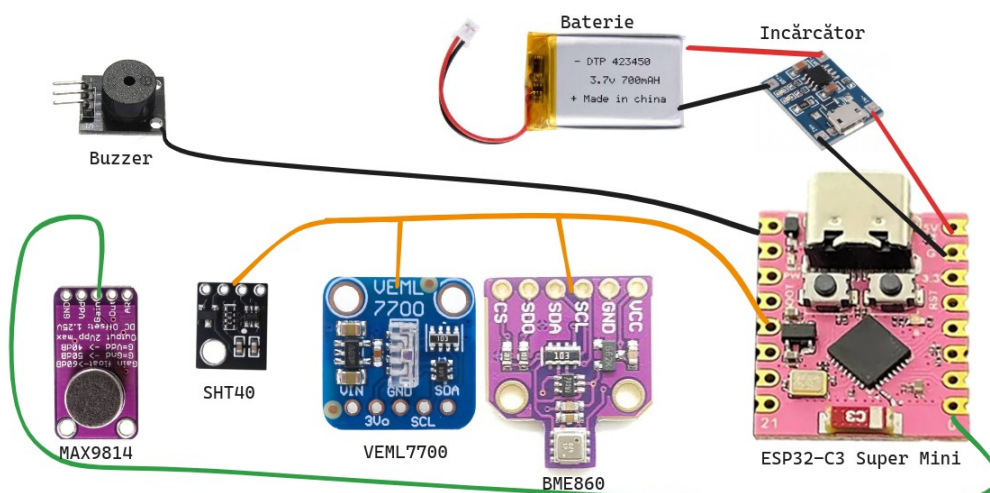


Figura 3.21 - Arhitectura hardware a dispozitivului

Dispozitivul este alimentat de o baterie de 700mAh care este alimentată prin intermediul unui modul de încărcare a bateriilor de polimer de litiu. În dependență de recurența cu care va fi setat să măsoare dispozitivul mediul, bateria poate ține o durată foarte mare. Așa cum dispozitivele ce utilizează baterii de tip CR2032 ce are capacitatea în jur de 200 mAh, pot funcționa de pe o baterie și un an.

4 REALIZAREA SISTEMULUI

Pentru realizarea sistemului au fost nevoie de mai multe instrumente software. S-a optat ca toate aceste instrumente să fie libere și open source pentru a nu fi nevoie de a le pirata. În continuare este prezentată lista de software utilizată pentru a realiza sistemul:

- Gnu/Linux [24] – sistem de operare;
- terminal – pentru a accesa editorul de text, compilatoarele și alte programe;
- Neovim [25] – editor de text hiper-extensibil bazat pe Vim [26];
- flutter-tools.nvim [27] – extensie Neovim pentru crearea aplicațiilor flutter;
- Visual Studio Code – editor de text;
- platform.io – plugin Visual Studio Code pentru crearea aplicațiilor pentru microcontrolere;
- Android Studio [28] – a fost utilizat emulatorul de android oferit de acesta;
- Git [29] – instrument pentru versionarea codului sursă;
- Github [30] – platformă online de găzduire a codului sursă.

Neovim este al doilea cel mai popular editor de text care rulează în terminal din anul 2023 [31]. Printre avantajele acestui editor este modul eficient de a edita textul fiind prezente mai multe moduri de editare, cum ar fi normal, insert, selectare vizuală și altele. Totodată acest editor are o comunitate activă care creează și mențin o mulțime de extensii utile. Acest editor poate fi adaptat pentru o mulțime de cazuri de utilizare, iar configurarea acestuia este realizată prin editarea unui fișier, care poate fi copiat și utilizat și pe alte calculatoare. Pentru a facilita dezvoltarea unei aplicații Flutter, a fost instalată extensia flutter-tools.nvim, care oferă colorarea sintaxei și alte funcționalități.

Fiind că aplicația mobilă este destinată de a rula pe telefoane mobile care de cele mai multe ori au procesoare ARM însă majoritatea calculatoarelor sunt pe platforma x86, a fost nevoie de a instala un emulator android pentru a permite testarea mai rapidă a schimbărilor de cod. Acest emulator a fost obținut instalând aplicația Android Studio.

Git este cel mai popular instrument pentru versionarea codului sursă [32]. Totodată el poate versiona și alte lucruri, nu doar codul sursă, spre exemplu documente sau imagini care vor fi modificate. Totuși este mult mai ușor de a lucra cu fișiere textuale decât cu fișiere binare în versionarea acestora, fiind posibilă compararea a două versiuni.

Github este o platformă online care permite găzduirea codului sursă, atât în repozitorii private cât și în repozitorii publice. Totodată oferă posibilitatea de a crea pipeline-uri pentru integrarea continuă și plasarea continuă, însă repozitoriile private au niște limitări în acest domeniu.

Pentru realizarea sistemului au fost utilizate mai multe limbaje de programare, compilatoare și librării. Limbajul de programare C++ și librăria arduino, pentru partea de sistem încorporat. Limbajul Dart împreună cu framework-ul Flutter pentru crearea aplicației mobile.

Pentru dezvoltarea proiectului a fost utilizat un laptop Lenovo Thinkpad E595 cu procesor AMD Ryzen 7 3700U cu 24 GB de RAM. Dintre perifericele utilizate în-afară de monitorul, tastatura și touchpad-ul laptopului au mai fost folosite încă 2 monitoare externe. O tastatură ergonomică, despicată și ortolinară Dactyl Manuform [33] și un trackball asemănător cu Logitech MX ERGO [34].

Codul sursă al aplicației mobile și al firmware-ului pentru dispozitiv poate fi accesat pe paginile respective ale acestora de GitHub. Pentru aceasta a fost creată o organizație numită Canti IoT, poate fi găsită la adresa: <https://github.com/orgs/Canti-IoT/repositories>. Aici este prezentă repozitoria pentru aplicație Canti-Hub și repozitoria pentru dispozitiv Canti-1.

4.1 Descrierea aplicației mobile

Primul dosar al proiectului aplicației mobile care a fost creat pentru realizarea aplicației este dosarul **assets**. Scopul acestui dosar este de a conține obiecte precum imagini sau alte fișiere care nu sunt cod în limbajul **dart**. În acest dosar se află alt dosar cu denumirea **devices** și un fișier numit **parameters.json**. Dosarul devices este destinat de a stoca pictograma pentru dispozitivul creat, în viitor în cazul că vor exista mai multe dispozitive pictogramele acestora vor fi adăugate în acest dosar. În fișierul parameters.json sunt enumerați parametrii care pot fi monitorizați de sistem, indexul acestora, recurența de eșantionare implicită în secunde, o valoarea normală, optimală a acestor parametri, și limita minimă și maximă recomandată, aceasta nu reprezintă o limită fizică. Conținutul acestui dosar și conținutul fișierului parameters.json este prezentat în figura 3.21.

```

1  {
2    "parameters": [
3      { "name": "battery", "index": 1, "recurrence": 60, "normal": 50.0, "max": 80.0, "min": 20.0, "units": "%", },
4      { "name": "temperature", "index": 10, "recurrence": 60, "normal": 21.0, "max": 32.0, "min": 12.0, "units": "°C", },
5      { "name": "humidity", "index": 20, "recurrence": 60, "normal": 45.0, "max": 70.0, "min": 30.0, "units": "%", },
6      { "name": "pressure", "index": 30, "recurrence": 60, "normal": 1013.0, "max": 1100.0, "min": 900.0, "units": "hPa", },
7      { "name": "altitude", "index": 31, "recurrence": 60, "normal": 0.0, "max": 5000.0, "min": 0.0, "units": "m", },
8      { "name": "vocs", "index": 40, "recurrence": 60, "normal": 75.0, "max": 500.0, "min": 0.0, "units": "IAQ", },
9      { "name": "noise", "index": 50, "recurrence": 60, "normal": 45.0, "max": 55.0, "min": 0.0, "units": "dB", },
10     { "name": "illumination", "index": 60, "recurrence": 60, "normal": 500.0, "max": 1500.0, "min": 0.0, "units": "lux", },
11     { "name": "uv", "index": 70, "recurrence": 60, "normal": 4.0, "max": 6.0, "min": 0.0, "units": "UV", },
12   ]
13 }

```

Figura 4.1 - Structura fișierului parameters.json

Următorul fișier care a fost direct creat în mapa proiectului se numește **l10n.yaml**, acesta conține setările pentru localizarea aplicației. Pentru ca localizarea să funcționeze, în fișierul **pubspec.yaml** au fost adăugate următoarele dependențe:

- flutter_localizations – traducerea aplicației;
- provider – librărie pentru gestionarea stării aplicației;
- url_launcher – lansarea unui browser cu URL selectat;
- flutter_octicons – librărie cu pictograme în format SVG;
- syncfusion_flutter_sliders – librărie pentru desenarea unui interval cu două cursoare;
- drift – librărie pentru accesarea unei baze de date SQLite locală;
- shared_preferences – stocarea setărilor sub formă de cheie – valoare;

- `permission_handler` – cererea permisiunilor de la utilizator;
- `flutter_blue_plus` – accesul la adaptorul Bluetooth.

Pentru ca localizarea să funcționeze este nevoie de generat fișierul `app_localisation.g.dart` cu ajutorul comenzii: **flutter gen-l10n**. Pentru a traduce aplicația în mai orice limbă este nevoie, se va copia fișierul **app_en.arb** din directoria **lib/l10n** cu extensia limbii dorite și se va traduce conținutul acestui fișier. După generarea fișierelor de cod, în timpul rulării aplicației, dacă limba setată în setările telefonului va fi schimbată, atunci și limba utilizată în aplicație va reflecta acest lucru.

Toate modulele de cod **dart** se află în dosarul **lib**. În continuare va fi expusă necesitatea fișierelor mai importante. În dosarul **lib/common** se află două fișiere. Primul este **files.dart** care conține o clasă cu attribute statice ce iau valori către fișierele utilizate în acest proiect, pentru a evita necesitatea de a scrie denumirea unui fișier în mai multe locuri, și pentru a evita viitoare erori de scriere. Fișierul **parameters.dart** conține definirea clasei `Parameter` cu attributele necesare și metoda `fromJson` pentru a deserializa fișierul **parameters.json**. Dosarul **lib/database** conține fișierele necesare pentru gestionarea unei baze de date SQLite locală. În fișierul **database.dart** este creată clasa `Database`, metoda de a crea o conexiune cu fișierul bazei de date, și sunt înregistrate tabelele bazei de date. Definirea tabelor are loc în fișierul **tables.dart**, iar în fișierul **custom_types.dart** sunt definite două enumerări, una pentru tipul dispozitivului iar cealaltă pentru metoda de eluare a parametrilor pentru alarme. Schema bazei de date poate fi vizualizată în figura 4.2, aceasta este generată automat atunci când fișierul bazei de date este deschis în DBeaver. În ultimul rând fișierul **database.g.dart** este fișierul generat de pluginul **drift** în urma executării comenzii **dart run build_runner build**. Acest fișier conține metode și clase pentru a interacționa cu baza de date și este utilizat în restul aplicației. Dosarul **lib/pages** conține dosare și fișiere necesare pentru afișare interfeței de utilizator, ele vor fi expuse pe scurt, iar reprezentarea acestor pagini va fi în capitolul cu documentarea. În primul rând prima pagină a aplicației este definită în fișierul **start_page.dart** această pagină afișează denumirea aplicației în timp ce în fundal sunt efectuate mai multe acțiuni definite în **Providers**. Printre aceste acțiuni se numără inițializarea bazei de date, încărcarea setărilor și temei din memorie și încărcarea parametrilor din fișierul json în caz că aceștia nu sunt în baza de date. Odată ce aceste acțiuni au fost finisate, pagina de start este înlocuită cu pagina principală, definită în fișierul **main_page.dart**. De pe această pagină se poate naviga pe pagina de setări sau pagina de configurat alarme.

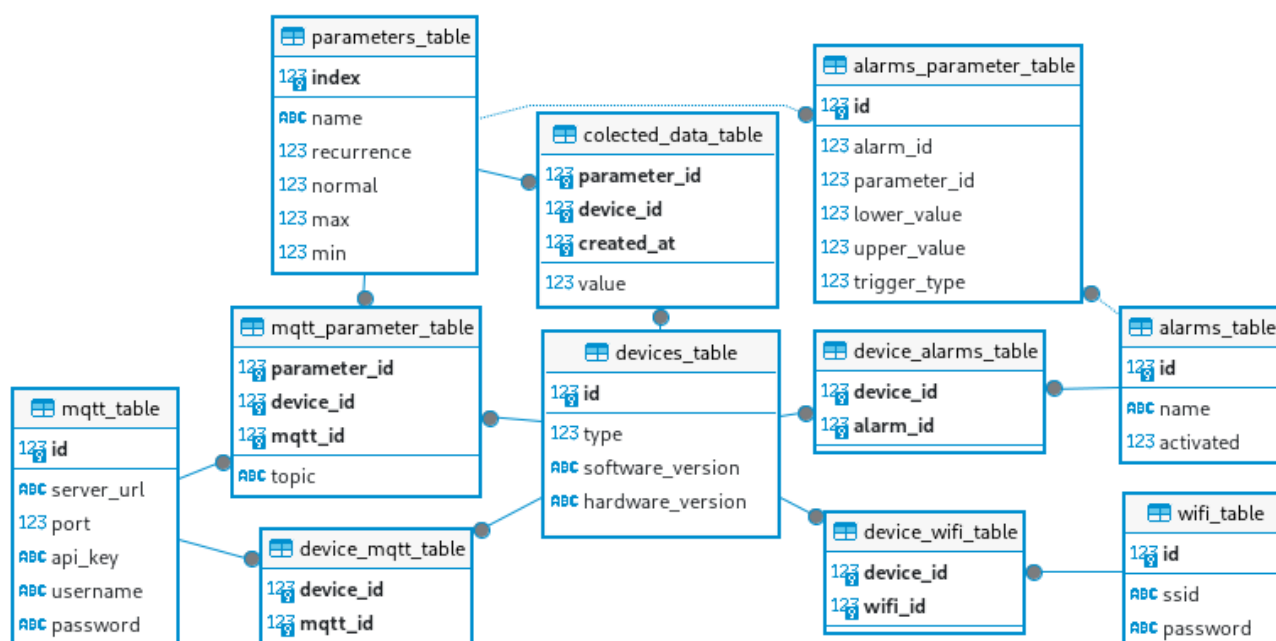


Figura 4.2 - Diagrama entităților și relațiilor bazei de date

În dosarul **lib/providers** sunt definite fișierele necesare pentru a lucra cu pluginul **provider** care a fost inclus în fișierul **pubspec.yaml** ca dependență. Provider este un plugin pentru gestionarea stării aplicației, în flutter elementele grafice nu au acces unele la altele iar asta face dificil de accesat date în diferite părți ale aplicației. Utilizând acest plugin este posibil de avut o stare globală pentru aplicație și de actualizat interfața grafică fără dificultăți. În fișierul **database_provider.dart** sunt definite mai multe metode, ce-a mai importantă este cea de inițializat baza de date, care presupune crearea unui obiect de tip bază de date, conectarea la acesta, și obținerea tuturor datelor din baza de date, pentru a le putea afișa. Celelalte metode sunt metode standard de creat, actualizat și șters datele din baza de date, este important de menționat că, în fiecare metodă, la sfârșit interfața de utilizator este notificată că datele au fost schimbate și necesită să facă o actualizare.

Providerul definit în **settings_provider.dart** permite salvarea setărilor ce țin de aplicație, care ar fi mai dificil de stocat în baza de date. Pentru aceasta a fost utilizat pluginul **shared_preferences**, acesta permite salvarea în memorie a datelor bazată pe chei. Datele stocate prin intermediul acestui modul sunt: actualizarea aplicației; actualizarea firmware dispozitiv; respectarea temei sistemului; respectarea temei întunecate; modul de afișare al parametrilor.

Providerul pentru temă, definit în fișierul **theme_provider.dart** accesează providerul pentru setări, și în dependență de valorile atributelor **systemTheme** și **darkTheme** care sunt valori binare, setează tema aplicației în modul corespunzător.

ParametersProvider definit în fișierul **parameters_provider.dart** este utilizat pentru a de-serializa datele din fișierul json și pentru a le insera în baza de date. Iar DeviceProvider din **device_provider.dart** este utilizat pentru a adăuga în sistem noi dispozitive, și pentru a afișa datele despre acestea.

Providerul de bluetooth este responsabil pentru oferirea datelor și metodelor pentru identificarea dispozitivelor pe rețeaua Bluetooth, conectarea la acestea, și schimbul de date cu ele.

4.2 Descrierea firmware-ului pentru dispozitiv

Firmware-ul pentru dispozitiv a fost creat cu ajutorul ecosistemului Platform.IO. Proiectul acestuia conține un fișier de configurare platformio.ini în care sunt toate configurările proiectului. Dintre flagurile de compilare mai interesante este BUILD_TIMESTAMP=\$UNIX_TIME care salvează timpul curent într-un macro, pentru a fi utilizat în cod. Dacă dispozitivul nu va mai fi alimentat după timpul se va desincroniza, totuși este mai bine ca acesta să aibă o valoare mai recentă. În acest fișier sunt declarate și librăriile utilizate, cu excepția arduino care este indicată ca framework. Librăriile incluse sunt:

- adafruit/Adafruit BME680 Library – driver pentru BME680;
- adafruit/Adafruit VEML7700 Library – driver pentru VEML7700;
- sensirion/arduino-sht – driver pentru SHT40;
- fbiego/ESP32Time – abstractizarea accesului la RTC-ul controlerului.

Execuția programului se începe în fișierul main.cpp. Ca orice program arduino, aici sunt prezente două funcții de abză: setup și loop. În setup a fost inițializat adaptorul Bluetooth, magistrala I2C, RTC-ul, managerul de senzori și managerul de alarme. În funcția loop, sunt executate ciclic funcțiile de intrare ale managerului de Bluetooth, managerului de senzori și managerului de alarme.

Pentru a accesa aceeași instanță de RTC, senzori și alarme, clasele respective au fost implementate după șablonul de proiectare singleton. Denumirea acestor clase sunt: RTCSingleton, SensorManager și AlarmManager.

SensorManager are o listă de senzori pe care îi gestionează. Acești senzori sunt implementați după șablonul adaptor, indiferent de clasa inițială oferită de librării, aceștia au fost adaptați pentru a putea fi puși într-un singur array. Care este parcurs, și operațiile sunt efectuate asupra tuturor elementelor. Pentru că senzorii citesc valori diferite și pentru a identifica acest lucru, metoda de citire a parametrului ia indexul parametrului, iar în această metodă se returnează valoarea conform indexului. Prin urmare se poate citi un parametru, fără țină cont care este obiectul senzorului dorit.

AlarmManager conține un 3 șiruri de elemente ce în sumă formează câte o alarmă. A fost prevăzut că pe un dispozitiv vor putea fi stocate doar 3 alarme, din cauza spațiului limitat. Pe lângă metodele de gestionare a alarmelor și ce-a în care este evaluată alarma. Această clasă mai are și o funcție prin intermediul căreia, AlarmManager este înștiințat de către SensorManager că a fost citită o valoare nouă, și că trebuie să evalueze alarmele din nou.

În clasa BLEServerManager sunt definite metodele de conectare și deconectare a dispozitivului. Sunt înregistrate cele trei servicii ale dispozitivului, serviciul de index, de valori și de configurări. Fiecare serviciu este implementat într-un fișier aparte. În serviciul de configurări este implementat protocolul de comunicare cum a fost prezentat în figura 3.11. Cu ajutorul serviciului de index, telefonul mobil, poate

afla ce indecși suportă dispozitivul dat. Indecșii sunt citați în mod repetat până când nu încep a se repeta. Serviciul de valori este similar în similitudine cu serviciul de indecși. Aplicația mobilă scrie în caracteristică indexul parametrului dorit și apoi citește valoarea respectivă.

4.3 Testarea sistemului

Sistemul a fost testat utilizând teste manuale, acestea au fost efectuate în timpul realizării. Atât flutter cât și platform.io oferă mecanisme de scriere a testelor unitare, însă din cauza timpului limitat s-a optat pentru utilizarea doar a testelor manuale. Aceste teste manuale presupun fie inspectarea elementelor grafice ale aplicației, fie printarea unor părți legate de starea aplicației în consolă și analizarea acestora. Testarea pentru dispozitiv de asemenea inițial a implicat analizarea datelor citite de pe portul serial, iar mai apoi când s-a ajuns la testarea comunicării Bluetooth, s-a utilizat aplicația nrf Connect. În continuare vor fi prezentate câteva cazuri de testare care au fost utilizate în timpul realizării.

În tabelul 4.1 sunt prezentați pașii pentru testarea, dacă o alarmă a fost creată. În starea inițială ne aflăm pe pagina cu alarme, apoi apăsăm pe butonul de creat o alarmă. După asta se va deschide pagina cu configurările noii alarme. Dacă apăsăm pe săgeata înapoi și ieșim de pe această pagină, ne întoarcem pe pagina cu alarme. Pe aceasta putem apăsa pe comutatorul care inițial este dezactivat pentru a activa alarma și schimba starea comutatorului.

Tabelul 4.1 - Testarea creării alarmelor

Stare inițială	Pagina cu alarme este deschisă
Pasul 1	Apăsarea butonului cu semnul +
Rezultatul așteptat	Crearea unei alarme noi
Rezultatul primit	Suntem redirecționați pe pagina, cu setări pentru noua alarmă creată
Pasul 2	Apăsăm pe săgeata înapoi
Rezultatul așteptat	Să fim redirecționați pe pagina cu alarme
Rezultatul primit	Suntem redirecționați pe pagina cu alarme
Pasul 3	Apăsarea comutatorului, în dreptul alarmei
Rezultatul așteptat	Comutatorul să își schimbe stare în activ
Rezultatul primit	Comutatorul arată activ

Acest test efectuat la moment a rezulta că aplicația funcționează în modul dorit, însă în momentul dezvoltării au fost întâmpinate o mulțime de probleme.








În tabelul 4.2 sunt prezentați pașii de testare comunicării prin Bluetooth, serviciul de indecși. Scopul acestui serviciu este de a transmite dispozitivelor conectate ce parametri poate monitoriza acest dispozitiv. Dacă caracteristica este citită, ea este înlocuită cu următorul index al parametrilor. Prin urmare se poate obține toată lista de indecși. În pasul 1 a fost apăsă odată și s-a obținut valoarea 0x0A, care corespunde indexului 10, pentru temperatura, la pasul 2 a fost apăsă din nou butonul de citire și s-a obținut 0x14 – 20, indexului parametrului de umiditate.

Tabelul 4.2 - Testarea transmițerii indecșilor

Stare inițială	Valoarea caracteristicii nu este afișată
Pasul 1	Apăsăm pe butonul de citit caracteristica
Rezultatul așteptat	Să primim prima valoare, a indexului - 0x0A
Rezultatul primit	<p>Unknown Service UUID: 4fafc201-1fb5-459e-8fcc-c5c9c331914b PRIMARY SERVICE</p> <p>Unknown Characteristic ↓ ↑ ⇅ ⇅ UUID: beb5483e-36e1-4688-b7f5-ea07361b26a8 Properties: INDICATE, NOTIFY, READ, WRITE Value: (0x) 0A-00-00-00</p> <p>Descriptors: Characteristic User Description ↓ ↑ UUID: 0x2901 Value: index Client Characteristic Configuration ↓ UUID: 0x2902</p>
Pasul 2	Apăsăm pe butonul de citit caracteristica
Rezultatul așteptat	Să primim valoarea 20 în hexazecimal - 0x14
Rezultatul primit	<p>Unknown Service UUID: 4fafc201-1fb5-459e-8fcc-c5c9c331914b PRIMARY SERVICE</p> <p>Unknown Characteristic ↓ ↑ ⇅ ⇅ UUID: beb5483e-36e1-4688-b7f5-ea07361b26a8 Properties: INDICATE, NOTIFY, READ, WRITE Value: (0x) 14-00-00-00</p> <p>Descriptors: Characteristic User Description ↓ ↑ UUID: 0x2901 Value: index Client Characteristic Configuration ↓ UUID: 0x2902</p>
Pasul 3	Apăsăm pe butonul de scriere apoi citire
Rezultatul așteptat	Să primim din nou valoarea 0x0A

Apoi în pasul 3, mai întâi a fost scris în caracteristică o valoare nulă, pentru a reseta starea caracteristicii. Apoi a fost citită valoarea caracteristicii. Valoarea citită a fost 0x0A, ceea ce reprezintă iarăși parametrul de temperatură. Însă dacă starea caracteristicii nu ar fi fost resetată ar fi trebuit să citim valoarea 0x1E.

Continuare tabelul 4.2 - Testarea transmiției indecșilor

Rezultatul primit	<p>Unknown Service UUID: 4fafc201-1fb5-459e-8fcc-c5c9c331914b PRIMARY SERVICE</p> <p>Unknown Characteristic     UUID: beb5483e-36e1-4688-b7f5-ea07361b26a8 Properties: INDICATE, NOTIFY, READ, WRITE Value: (0x) 0A-00-00-00</p> <p>Descriptors: Characteristic User Description   UUID: 0x2901 Value: index Client Characteristic Configuration  UUID: 0x2902</p>
-------------------	---

Testarea acestui funcțional a rezultat în succes chiar și din prima încercare fără necesitatea de a corecta codul, așa cum implementarea acestui funcțional este foarte simplă.

5 DOCUMENTAREA PRODUSULUI REALIZAT

Aplicația mobilă poate fi descărcată de pe pagina de GitHub, secțiunea de publicări <https://github.com/Canti-IoT/Canti-Hub/releases>. Dispozitivul poate fi asamblat conform schematicii figura 3.21, iar software-ul pentru acesta de asemenea poate fi descărcat de pe pagina de GitHub, <https://github.com/Canti-IoT/Canti-1>. Apoi este nevoie de a conecta dispozitivul prin portul usb, și de a îl programa din visual stuido code.

La moment dispozitivul este la etapa de machetă, pentru a fi mai ușor programat acesta este conectat prin cablu, însă cablul poate fi deconectat, și să fie conectată bateria în loc. Singura metodă cu care utilizatorul poate interacționa cu dispozitivul direct este de a îl alimenta sau a îi opri alimentarea.

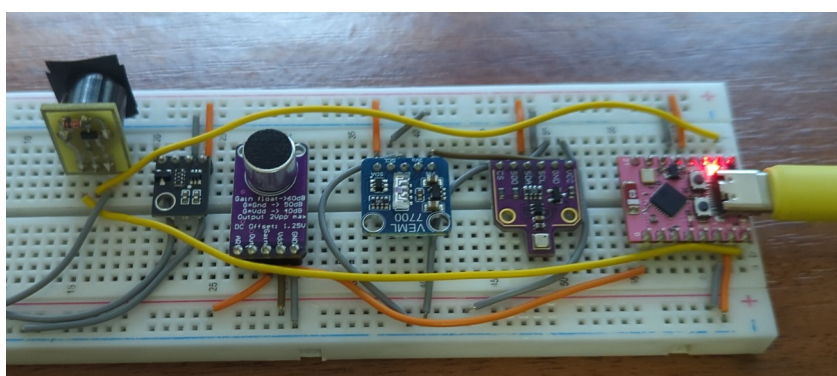


Figura 5.1 - Pagina de pornire

Pentru aplicația respectivă nu există noțiune de administrator sau utilizator simplu, utilizatorul este administratorul rețelei sale personale de dispozitive, nimeni în-afară de el nu are acces la aceste dispozitive. Aplicația funcționează fără de conexiunea la internet, însă pentru funcționalul de MQTT când acesta va fi realizat, va avea nevoie de acces la internet.

Odată ce aplicația este lansată, pe ecran va fi afișată pagina de pornire atât timp cât resursele aplicației sunt încărcate în memorie. Aspectul aceste pagini este reprezentat în figura 5.2. Această pagină nu ar trebui să fie afișată mai mult de câteva secunde iar după ce încărcarea este finisată, automat se va trece la pagina principală a aplicației.

Canti Hub

Figura 5.2 - Pagina de pornire

În figura 5.3 este prezentată pagina principală, cu elementele principale marcate. Pe bara aplicației sunt prezente două pictograme, setări marcată cu 1, apăsarea acesteia va deschide pagina cu setări reprezentată în figura 5.4 și alarme marcată cu 2, apăsarea acesteia va deschide pagina cu alarme afișată în figura 5.10. Marcat cu cifra 3 este lista de dispozitive conectate, această listă poate fi derulată orizontal.

Apăsarea pictogramei unui dispozitiv va afișa detaliile acestui dispozitiv, în special parametri monitorizați de acesta, acești parametri sunt afișați în secțiunea marcată cu 5. Pictograma marcată cu cifra 4 va deschide pagina de adăugat dispozitive reprezentată în figura 5.13.

Aceasta are o bară pe care în partea stânga se află pictograma setărilor iar în partea dreaptă se află pictograma alarmelor. Acest lucru va face atât setările cât și alarmele accesibile. Dedesubtul acestei bare se află o listă de dispozitive. Acestea vor fi dispozitivele înregistrate în sistem. Dispozitivele active vor fi prezentate cu un disc de culoarea verde iar cele inactive cu un disc de culoarea roșie.

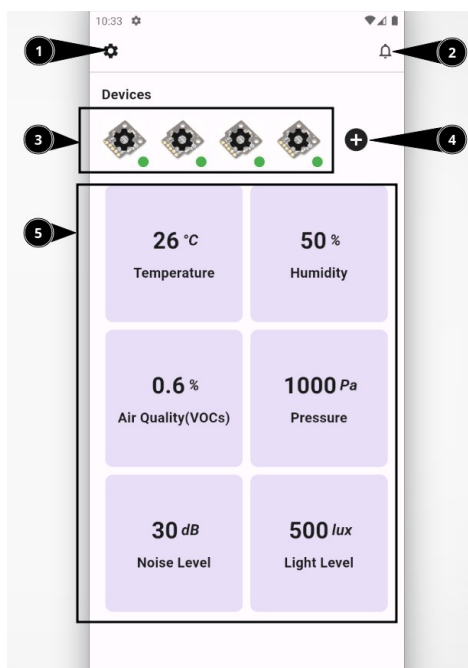


Figura 5.3 - Pagina principală

Așa cum a fost menționat mai sus, pagina cu setări este reprezentată în figura 5.4, pe aceasta fiecare element este marcat cu o cifră pentru a descrie rostul fiecăruia mai clar. Marcat cu 1 este pictograma unei săgeți care întoarce aplicația la pagina principală. Marcat cu 2 este linkul către pagina de GitHub a proiectului. Marcat cu cifra 3 este un comutator care activează sau dezactivează verificarea actualizărilor pentru aplicația mobilă. Mai jos, marcat cu 4 este comutatorul pentru verificarea în mod similar dacă există actualizări pentru firmware-ul dispozitivului. Activarea comutatorului marcat cu 5 va face ca aplicația să respecte tema de sistem iar elementul din setări marcat cu 6 să fie ascuns. Dacă comutatorul 5 este dezactivat atunci avem posibilitatea de a seta tema aplicației să fie întunecată prin intermediul comutatorului marcat cu 6. Modul de afișare al parametrilor pe pagina principală poate fi setat prin intermediul listei de setări marcată cu 7, opțiunile prezente sunt ca parametrii să fie afișați în listă, sau în grilă cu un element, două elemente sau 3 elemente pe rând. Pentru a configura setările parametrilor, poate fi apăsat câmpul marcat cu cifra 8, în urma acestei acțiuni va fi deschisă pagina reprezentată în figura 5.5. Pentru a adăuga conexiuni WiFi pentru a fi setate pe dispozitiv, poate fi accesată pagina reprezentată în figura 5.7 apăsând elementul marcat cu cifra 9. În ultimul rând pentru a

adăuga credențiale pentru conexiuni MQTT, poate fi apăsat elementul cu cifra 10, ca rezultat va fi deschisă pagina cu lista de conexiuni MQTT, asemănătoare cu pagina de setări WiFi.

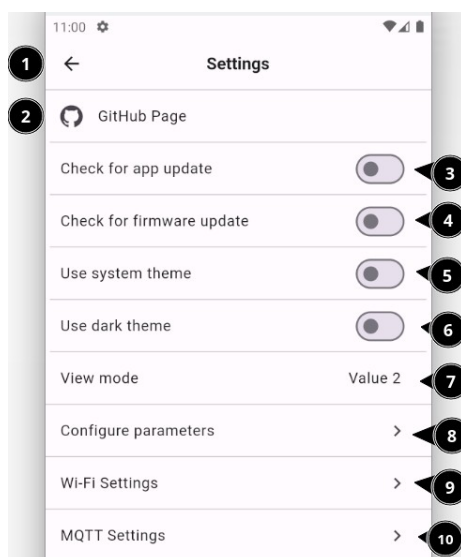


Figura 5.4 - Setările aplicației

În figura 5.5 este prezentat meniul de configurare al parametrilor, acesta poate fi accesat din pagina de setări. Pe această pagină, dacă apăsăm pe pictograma cu un creion, marcată cu cifra 1 va fi deschis un meniu, care permite editarea atributelor acestui parametru. Reprezentarea acestui meniu este în figura 5.6.

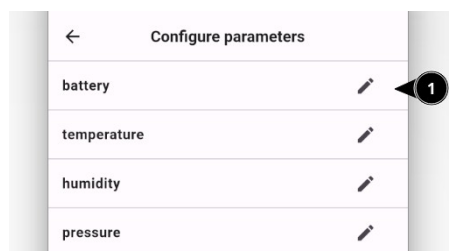


Figura 5.5 - Lista parametrilor configurabili

În figura 5.6 este prezentat meniul pentru modificarea atributelor unui parametru. Pentru fiecare parametru putem modifica recurența cu care se eșantionează acesta, valoarea normală a acestui parametru, valoarea admisibilă maximă și minimă.

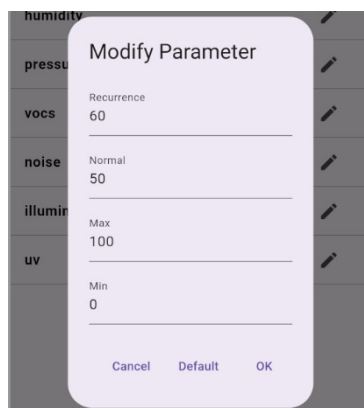


Figura 5.6 - Configurarea unui parametru

În figura 5.7 este prezentată o porțiune a paginii ce conține lista de rețele WiFi, cu marcasele corespunzătoare. Marcat cu numărul 1 este denumirea rețelei, mai exact identificatorul SSID al acesteia. În partea dreaptă a câmpului cu fiecare rețea WiFi sunt două iconițe, pictograma care reprezintă un creion, marcată cu cifra 2 permite deschiderea meniului de editare a acestei rețele, acest meniu este prezentat în figura 5.8, merită de menționat că acest meniu este identic atât pentru editare cât și pentru crearea unei noi rețele. Marcat cu cifra 3, este o pictogramă ce reprezintă un coș de gunoi, dacă acesta este apăsat, rețeaua va fi ștearsă din baza de date.



Figura 5.7 - Lista rețelelor WiFi

În figura 5.8 este prezentat meniul de editare sau creare al rețelei WiFi. Prin intermediul acestui meniu poate fi introdus SSID-ul și parola rețelei.

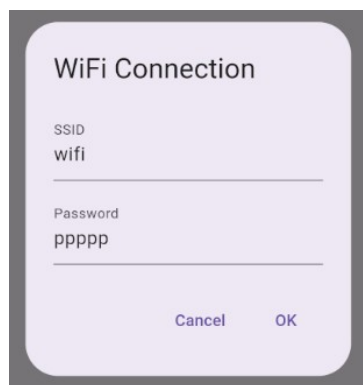


Figura 5.8 - Editarea/Adăugarea unei noi rețele WiFi

Similar cu meniul pentru editarea rețelei WiFi este și meniul pentru crearea și editarea unei conexiuni MQTT, acest meniu poate fi vizualizat în figura 5.9. Câmpurile acestui meniu sunt:

- adresa URL a serverului;
- portul brokerului MQTT;
- cheia api;
- numele de utilizator;
- parola.

Merită de menționat că în majoritatea cazurilor va fi nevoie de introdus fie cheia api fie parola.

Figura 5.9 - Editarea/Adăugarea unei conexiuni MQTT

În figura 5.10 este prezentată pagina cu lista de alarme, care poate fi accesată din pagina principală a aplicației. Marcat cu 1 este pictograma unei săgeți care întoarce aplicația pe pagina precedentă. Cu cifra 2 este marcat numele alarmei, acesta implicit va lua valoarea de Alarmă, dar poate fi editat. Pentru a edita proprietățile alarmei trebuie să fie apăsată pictograma cu un creion marcată cu cifra 3, rezultatul acestei acțiuni va fi deschiderea paginii de configurare a alarmei, vizibilă în figura 5.11. Alarma poate fi activată sau dezactivată utilizând comutatorul marcat cu cifra 4. În ultimul rând în partea de jos a paginii, marcat cu cifra 5 este un buton care permite crearea noilor alarme. Pagina care va fi deschisă în urma apăsării pe acest buton este identică cu cea pentru editare, cu diferența că inițial va fi goală.

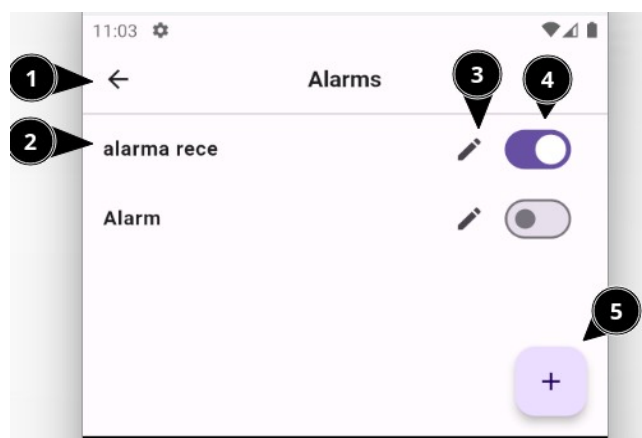


Figura 5.10 - Pagina cu alarme

Pagina de configurare a unei alarme este prezentată în figura 5.11, elementele acestei pagini sunt marcate pentru a fi identificate mai ușor. Marcat cu cifra 1 este denumirea alarmei, acesta este singurul câmp care este prezent când o alarmă este creată, și poate fi modificat. Marcat cu 2 este un dreptunghi care conține mai multe detalii despre configurarea pentru fiecare parametru în parte. Cu cifra 3 sunt grupate 3 butoane radio care permit dezactivarea parametrului sau setarea tipului de interval dorit, intervalul în interior va activa alarma atunci când valoarea parametrului este mai mare ca valoarea minimă și mai mică ca valoarea maximă, iar intervalul în exterior va activa alarma atunci când valoarea parametrului este mai mică de cât valoarea minimă sau mai mare de cât valoarea maximă. Pentru selectarea acestui interval sunt prezente 2 mânere de glisare, marcat cu 4 este mânerul pentru valoarea

minimă iar cu 5 este mânerul pentru valoarea maximă. În caz că este dificil de tras aceste mânere pentru a obține valorile dorite, se poate de apăsat pictograma cu creion, marcată cu cifra 6 pentru a deschide un meniu pentru introducerea de la tastatură a acestor valori. Marcat cu cifra 7 este pictograma unei cutii de gunoi, apăsând-o parametrul respectiv va fi șters din configurarea alarmei. Pentru a adăuga parametri noi, este utilizat butonul marcat cu cifra 8, în urma apăsării acestuia va fi deschis meniul reprezentat în figura 5.12, unde poate fi selectat un parametru dorit care încă nu se află în configurarea alarmei. Trebuie luat în considerare faptul că alarma nu va fi activată dacă cel puțin condiția pentru un parametru nu va fi îndeplinită. În caz că se dorește ștergerea alarmei, acest lucru poate fi realizat prin apăsarea butonului marcat cu cifra 9.

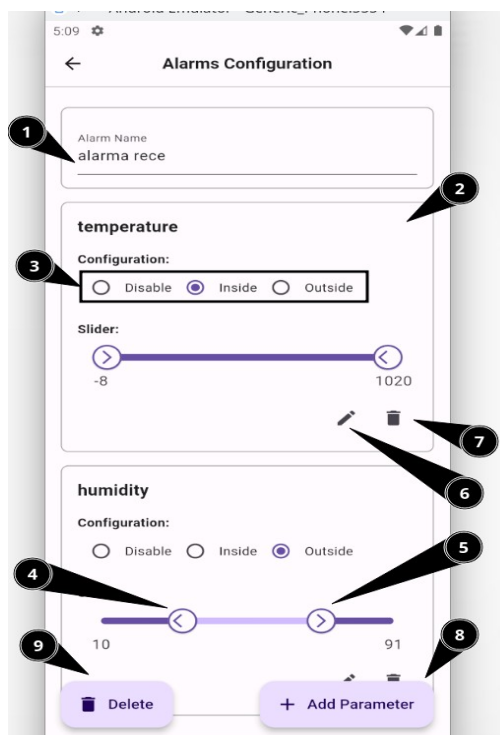


Figura 5.11 - Configurarea unei alarme

În figura 5.12 este prezentat meniul pentru selectarea parametrului care să fie adăugat în cadrul alarmei. Acesta are două butoane, primul este de anulare a acțiunii iar al doilea este de a adăuga parametrul.

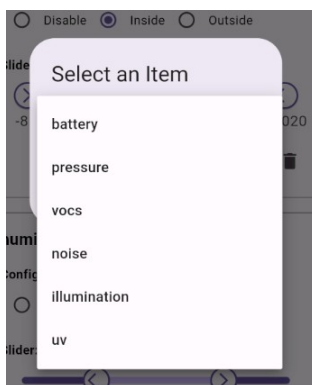


Figura 5.12 - Selectarea parametrului pentru a fi adăugat

În figura 5.13 este reprezentat meniul de adăugare a unui dispozitiv pentru cazul când se dorește adăugarea unui dispozitiv MQTT (virtual). La moment este posibil de adăugat dispozitive fizice sau virtuale, diferența este că conexiunea cu dispozitivul fizic va fi efectuată prin intermediul Bluetooth iar conexiunea dispozitivului virtual va fi efectuată prin intermediul MQTT. Tipul dispozitivului dorit poate fi selectat prin intermediul meniului marcat cu cifra 1. Dacă se optează pentru dispozitiv fizic, conținutul paginii va fi diferit. Apoi prin intermediul meniului marcat cu cifra 2 se poate selecta conexiunea MQTT care să fie utilizată pentru acest dispozitiv virtual. Acest meniu va afișa adresa URL a brokerului MQTT și numele de utilizator, pentru a distinge mai mulți utilizatori pe același server. În regiunea marcată de cifra 3 sunt prezentați toți parametrii care pot fi adăugați în acest dispozitiv virtual. Fiecare parametru are o casetă de bifat, în caz că aceasta este bifată, va fi prezentat un câmp adițional pentru acest parametru. Câmpul dat este marcat cu cifra 4, pentru unul dintre parametri. Acest câmp reprezintă topic-ul de pe care va fi preluat parametrul. `${USERNAME}` și `${PARAMETER}` sunt niște variabile speciale care vor fi înlocuite cu numele utilizatorului al serviciului MQTT și denumirea parametrului în limba engleză cu litere mici, acest parametri vor ușura adăugarea de topic-uri în cazurile când ele au un nume standard. Marcat cu cifra 5 este butonul de anulare în cazul în care nu se mai dorește adăugarea acestui dispozitiv, acest buton este diferit de săgeata din bara de sus, aceasta doar va ieși din pagină, însă nu va reseta valorile introduse. Marcat cu cifra 6 este butonul care finisează adăugarea dispozitivului.

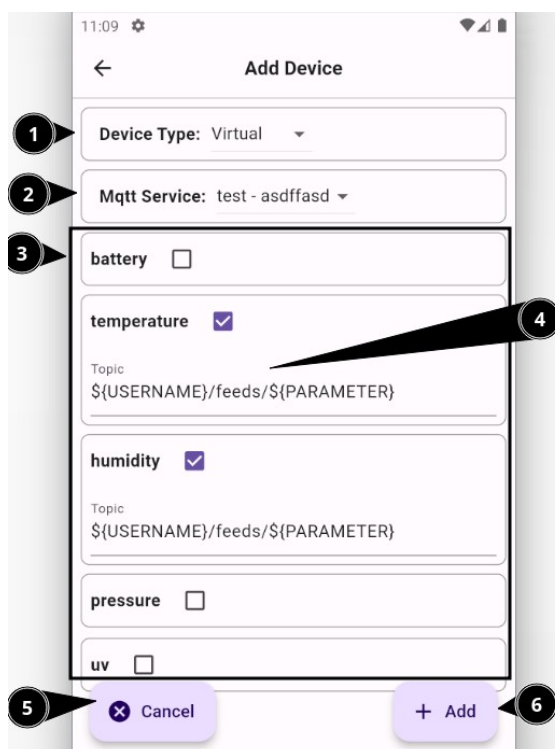


Figura 5.13 - Pagina pentru adăugarea unui dispozitiv, opțiunea MQTT

În figura 5.14 este prezentat modul de a conecta un dispozitiv la aplicație. Trebuie de selectat tipul dispozitivului ca Bluetooth în al doilea câmp. Primul câmp este opțional, el va lua numele dispozitivului dacă este lăsat gol. Apoi trebuie de selectat dispozitivul dorit, este posibil de selectat doar un dispozitiv.

Apoi se apasă pe pe butonul de adăugare din josul paginii. Adăugarea va dura câteva secunde, în această perioadă nu se recomandă apăsarea pe nici un buton. În special pe acela cu săgeata înapoi.

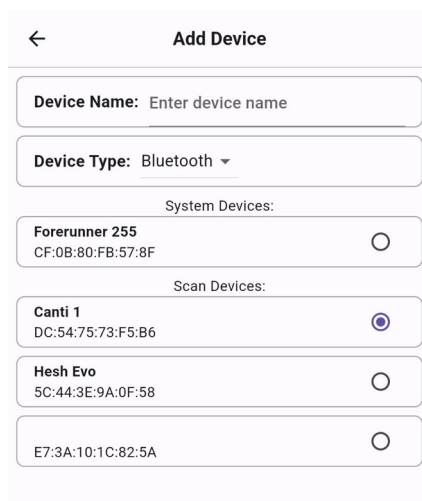


Figura 5.14 - Pagina pentru adăugarea unui dispozitiv, opțiunea Bluetooth

În figura 5.15, avem pagina principală, când este un dispozitiv conectat. Din setări putem selecta modul de afișare a parametrilor, în acest caz ei sunt afișați într-o listă. Atunci când dispozitivul este online în partea de jos dreapta a iconiței sale este un cerculeț verde, dacă dispozitivul este deconectat, atunci cerculețul își schimbă culoarea în roșu.

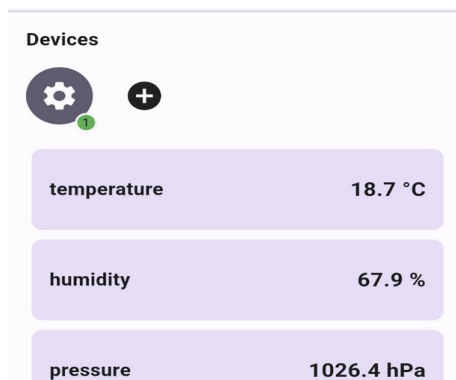


Figura 5.15 - Pagina principală cu un dispozitiv conectat

În figura 5.16 este prezentat cazul când adaptorul Bluetooth este deconectat. În acest caz pe ecran este afișat acest mesaj, și i se propune utilizatorului să apese pe el pentru a activa adaptorul Bluetooth.

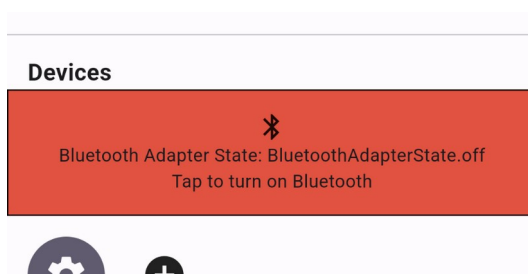


Figura 5.16 - Adaptorul Bluetooth este deconectat

6 ESTIMAREA COSTURILOR PROIECTULUI

Pentru estimarea costurilor proiectului, a fost utilizat instrumentul de gestionare a proiectelor Gnome Planner. În acesta au fost introduse resursele, adică rolurile persoanelor, și salariile acestea. Au fost create sarcinile, care pot fi văzute în anexa A. În cele din urmă programul a calculat costul proiectului.

Pentru a estima costul proiectului a fost nevoie de a identifica salariile pentru fiecare rol în parte. Salariile în domeniul IT pentru Republica Moldova au fost luate de pe acest url: <https://www.paylab.com/md/salaryinfo/information-technolo>. Ca salariu de referință a fost luat jumătate din salariul maxim pentru fiecare rol. Datele extrase de pe această pagină web sunt prezentate în tabelul 6.1.

Tabelul 6.1 - Salariul pe roluri

Titlu rol	Salariu pe lună, mdl	Salariu pe oră, mdl
Programator C++	14028	80
Dezvoltator Android	13411	76
Analist Business IT	11601	66
IT Proiect Manager	12261	70
Scriitor tehnic	7377	42
Tester IT	8813	50
Arhitect IT	20171	115

În cele din urmă a fost estimat că costul total pentru realizarea proiectului ar fi 91704 de lei, iar durata de realizare a fost estimată la 1320 de ore dacă acestea să fie efectuate secvențial. Din suma totală 11200 ar fi utilizați pentru elaborarea sarcinilor și gestionarea proiectului. 11616 ar fi utilizați pentru analiza domeniului. 19320 ar fi utilizați pentru proiectarea sistemului. 46880 ar fi utilizați pentru realizarea sistemului și 2688 ar fi utilizați pentru documentarea sistemului. Estimarea costurilor în întregime poate fi observată în tabelul A.1.

Sarcinile enumerate în tabelul A.1 mai sunt și reprezentate sub formă de diagramă WBS în figura 5.11. În această figură sunt prezentate etapele, sarcinile, costurile și orele necesare pentru realizarea acestora.

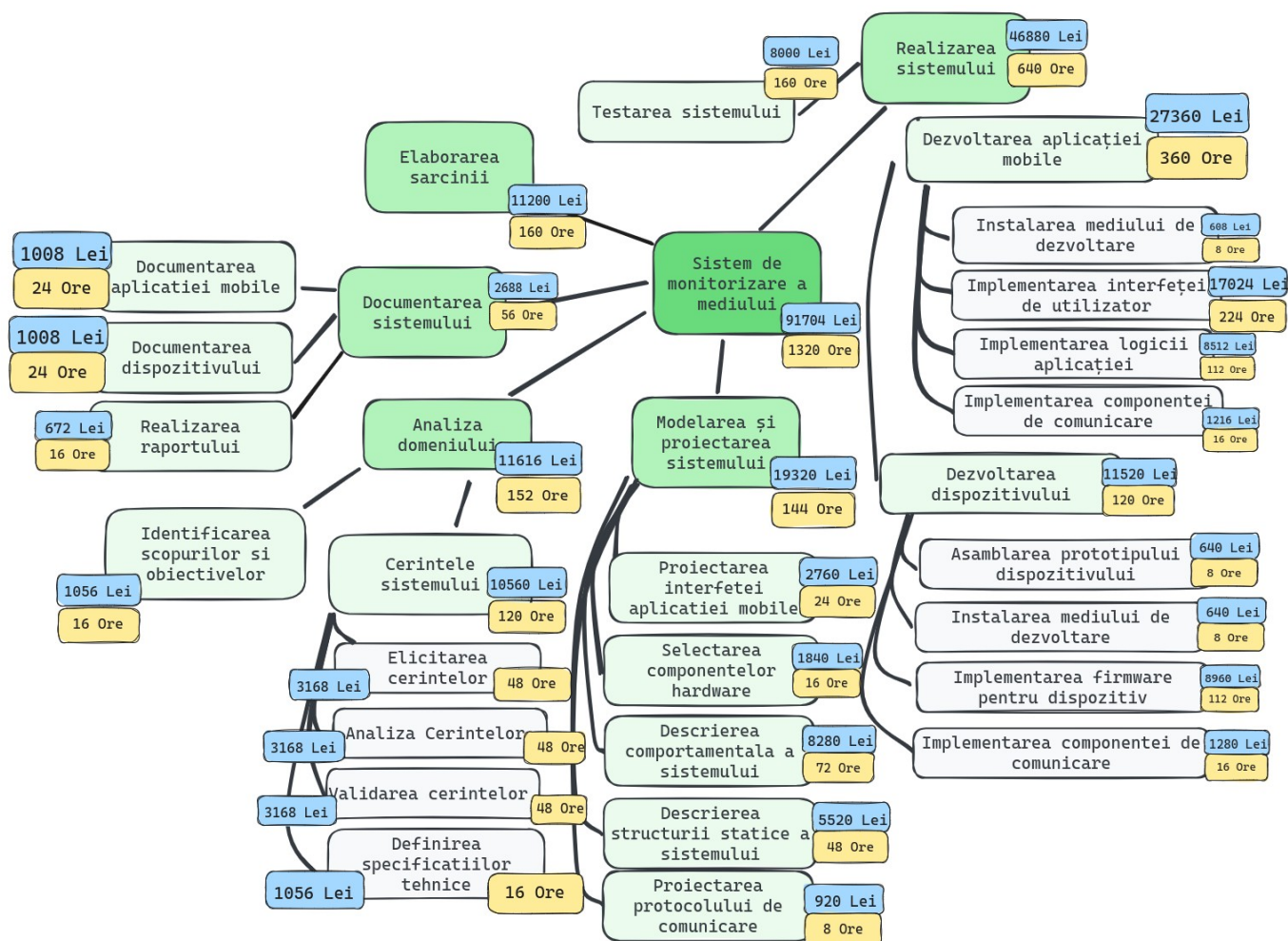


Figura 6.1 - Diagrama WBS

În final, graficul realizării sarcinilor, sub formă de diagramă gantt poate fi vizualizat în figura 5.13. Pe această diagramă sunt prezentate duratele cât se desfășoară sarcinile, iar în dreapta barei albastre este scris rolul care este responsabil de efectuarea sarcinii respective.

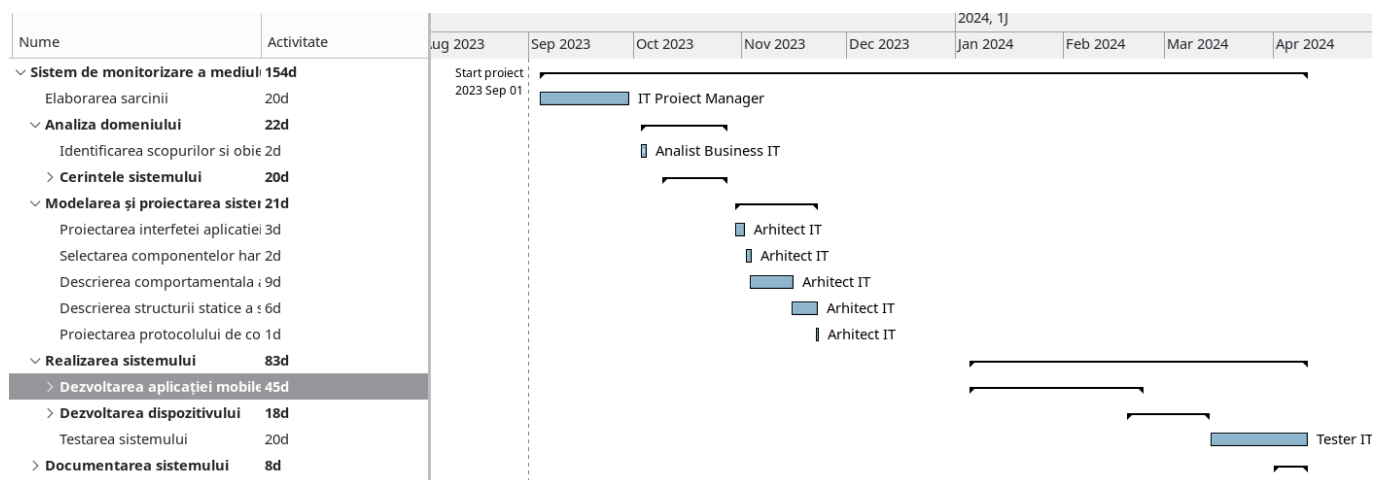


Figura 6.2 - Diagrama Gantt

CONCLUZII

Tehnologiile IoT facilitează monitorizarea mediului pe o scară largă și analiza datelor colectate. Aplicarea lor în acest domeniu poate aduce un impact pozitiv asupra lumii. Pe piață nu există o selecție largă de sisteme dedicate utilizării personale, care să fie portabile, cu o durată mare de viață a bateriei și să măsoare un număr mare de parametri ai mediului. Dacă ar fi posibil produs sistemul dezvoltat în cadrul acestui proiect, cu siguranță că s-ar găsi cel puțin o persoană care să beneficieze utilizând sistemul.

Nu toate cerințele sistemului au fost îndeplinite. În special interoperabilitatea asigurată prin adăugarea comunicării prin protocolul MQTT, aceasta ar putea mări numărul cazurilor de utilizare ale acestui sistem. Dispozitivul va fi util și în afara spațiului personal al utilizatorului.

Majoritatea diagramelor UML proiectate nu au valoare pentru acest proiect, cele mai utile sunt cele de mașină de stări și în special protocolul de comunicare. Eforturile viitoare ce țin de proiectare ar trebui dedicate proiectării PCB-ului dispozitivului și carcasei acestuia.

Inițial s-a estima că dispozitivul și aplicația se vor dezvolta în perioade egale de timp, însă aplicația mobilă a luat cea mai mare parte a timpului. În ciuda numărului mai mare de librării și resurse online. Asta explică de ce există un număr mai mare de roluri pentru dezvoltatori Web/Front-end/UI. O cauză a duratei lungi de dezvoltare este testarea manuală, care necesită asigurarea că interfața de utilizator arată în modul stabilit și se comportă așa cum trebuie la interacțiunea cu aceasta.

Deși sistemul nu este gata de producție, prețul pentru componentele dispozitivului în sumă nu depășesc suma de 400 de lei. Pentru a comercializa un prototip inițial al dispozitivului, s-ar putea de proiectat doar un PCB pe care să se lipească modulele deja existente și testate de producătorii lor. Carcasa poate fi modelată și imprimată 3D, fiind că această metodă este mai accesibilă pentru producerea pe scară mică.

BIBLIOGRAFIE

- [1] D. E. Greenfield, „Environmental Monitoring: A Complete Guide”, Sigma Earth. Data accesării: 26 februarie 2024. [Online]. Disponibil la: <https://sigmaearth.com/environmental-monitoring-a-complete-guide/>
- [2] „What are some of the current trends and challenges in environmental monitoring?” Data accesării: 23 noiembrie 2023. [Online]. Disponibil la: <https://www.linkedin.com/advice/1/what-some-current-trends-challenges-environmental>
- [3] „All about Internet of Things (#IoT)”. Data accesării: 26 februarie 2024. [Online]. Disponibil la: <https://www.linkedin.com/pulse/all-internet-things-iot-ahmed-karam>
- [4] „What Is Information Technology? A Beginner’s Guide to the World of IT”, Rasmussen University. Data accesării: 26 februarie 2024. [Online]. Disponibil la: <https://www.rasmussen.edu/degrees/technology/blog/what-is-information-technology/>
- [5] R. Christopher, *Civilized to Death: The Price of Progress*. Data accesării: 7 aprilie 2024. [Online]. Disponibil la: https://www.amazon.com/Civilized-to-Death-audiobook/dp/B07QGH3TDF/ref=sr_1_1?crid=CONMXH5XS74S&dib=eyJ2IjojMSJ9.6IB6Bz9DlPHDd0kODBNXVwx8vSV934bt9iF8AnLkQHI54wdKsRvEREv6erXxL8Ca3R4uQ1Jwj9P4AqTrayoF1uirKC4n_FNHXiSO6WZLHGCKc093lf0jP4pR3XuO88kWVAnmF4pEjfUAiYAyDFjqP8sUxcphX0CBmb8EVobV7PrdeC2UyqZW0nocGq2VXf4xVamtjKn-s2uxS60ioAhYWCcNm8gjsjaw3dhBPaRh8.XeQ-nYkYzOocYUiSN1_H_26Col8El-luqRg5_QN8pWk&dib_tag=se&keywords=civilized+to+death&qid=1712518477&srefix=civilized+to+%2Caps%2C321&sr=8-1
- [6] R. Fuller *et al.*, „Pollution and health: a progress update”, *Lancet Planet. Health*, vol. 6, nr. 6, pp. e535–e547, iun. 2022, doi: 10.1016/S2542-5196(22)00090-0.
- [7] D. Khovalyg *et al.*, „Critical review of standards for indoor thermal environment and air quality”, *Energy Build.*, vol. 213, p. 109819, apr. 2020, doi: 10.1016/j.enbuild.2020.109819.
- [8] „Atmotube PRO - Wearable and portable air quality monitor”. Data accesării: 23 noiembrie 2023. [Online]. Disponibil la: <https://atmotube.com/atmotube-pro>
- [9] „tempeTM - Garmin Moldova”. Data accesării: 23 noiembrie 2023. [Online]. Disponibil la: <https://garmin.md/produs/010-11092-30/>
- [10] „DT-8820 - 4 in 1 Multifunction Environment Meter”. Data accesării: 21 februarie 2024. [Online]. Disponibil la: <https://www.cem-instruments.com/en/product-id-929>
- [11] „An Overview of the IoT Tech Stack | IoT Glossary”. Data accesării: 23 noiembrie 2023. [Online]. Disponibil la: <https://www.emnify.com/iot-glossary/iot-stack>
- [12] Russell, „FPGA vs. Microcontroller vs. ASIC”, Nandland. Data accesării: 28 februarie 2024. [Online]. Disponibil la: <https://nandland.com/lesson-2-fpga-vs-micro-vs-asic/>
- [13] M. Lab, „Bare-metal and RTOS Based Embedded Systems”, Microcontrollers Lab. Data accesării: 29 februarie 2024. [Online]. Disponibil la: <https://microcontrollerslab.com/difference-bare-metal-and-rtos-based-embedded-systems/>
- [14] „Why I rewrote my Rust keyboard firmware in Zig: consistency, mastery, and fun”. Data accesării: 29 februarie 2024. [Online]. Disponibil la: <https://kevinlynagh.com/rust-zig/>
- [15] „A 2022 Guide to IoT Protocols and Standards”, Particle. Data accesării: 29 februarie 2024. [Online]. Disponibil la: <https://www.particle.io/iot-guides-and-resources/iot-protocols-and-standards/>
- [16] „HG353/2010 cu privire la aprobarea cerințelor minime de securitate și sănătate la locul de muncă”. Data accesării: 20 februarie 2024. [Online]. Disponibil la: https://www.legis.md/cautare/getResults?doc_id=22129&lang=ro
- [17] „Regulamentul sanitar privind normativele de emitere a zgomotului și a vibrației”. Data accesării: 20 februarie 2024. [Online]. Disponibil la: https://gov.md/sites/default/files/document/attachments/intr07_1_18.pdf
- [18] M. Gulin, „Answer to «Body Energy Consumption: calculating watts and kwh given calories?»”, Physics Stack Exchange. Data accesării: 27 februarie 2024. [Online]. Disponibil la: <https://physics.stackexchange.com/a/702472>

- [19] „AADL resource pages”. Data accesării: 1 martie 2024. [Online]. Disponibil la: <http://www.openaadl.org/>
- [20] „Unified Modeling Language, v2.5.1”, *Unified Model. Lang.*.
- [21] „Modelio Open Source - UML and BPMN free modeling tool”. Data accesării: 1 martie 2024. [Online]. Disponibil la: <https://www.modelio.org/index.htm>
- [22] „Ghidul simplu pentru diagrame UML și modelarea bazelor de date”. Data accesării: 1 martie 2024. [Online]. Disponibil la: <https://www.microsoft.com/ro-ro/microsoft-365/business-insights-ideas/resources/guide-to-uml-diagramming-and-database-modeling>
- [23] B. P. Douglass, *Design patterns for embedded systems in C: an embedded software engineering toolkit*, 1st ed. Oxford ; Burlington, MA: Newnes/Elsevier, 2011.
- [24] „SOLID: The First 5 Principles of Object Oriented Design | DigitalOcean”. Data accesării: 3 martie 2024. [Online]. Disponibil la: <https://www.digitalocean.com/community/conceptual-articles/s-o-l-i-d-the-first-five-principles-of-object-oriented-design>
- [25] „Linux and GNU - GNU Project - Free Software Foundation”. Data accesării: 3 martie 2024. [Online]. Disponibil la: <https://www.gnu.org/gnu/linux-and-gnu.en.html>
- [26] „Home - Neovim”. Data accesării: 3 martie 2024. [Online]. Disponibil la: <https://neovim.io/>
- [27] „welcome home : vim online”. Data accesării: 3 martie 2024. [Online]. Disponibil la: <https://www.vim.org/>
- [28] Akin, „akinsho/flutter-tools.nvim”. 2 martie 2024. Data accesării: 3 martie 2024. [Online]. Disponibil la: <https://github.com/akinsho/flutter-tools.nvim>
- [29] „Download Android Studio & App Tools”, Android Developers. Data accesării: 3 martie 2024. [Online]. Disponibil la: <https://developer.android.com/studio>
- [30] „Git”. Data accesării: 3 martie 2024. [Online]. Disponibil la: <https://git-scm.com/>
- [31] „Build software better, together”, GitHub. Data accesării: 3 martie 2024. [Online]. Disponibil la: <https://github.com>
- [32] „Stack Overflow Developer Survey 2023”, Stack Overflow. Data accesării: 23 noiembrie 2023. [Online]. Disponibil la: https://survey.stackoverflow.co/2023/?utm_source=social-share&utm_medium=social&utm_campaign=dev-survey-2023
- [33] „Stack Overflow Developer Survey 2022”, Stack Overflow. Data accesării: 3 martie 2024. [Online]. Disponibil la: https://survey.stackoverflow.co/2022/?utm_source=social-share&utm_medium=social&utm_campaign=dev-survey-2022
- [34] „Home Zig Programming Language”. Data accesării: 21 februarie 2024. [Online]. Disponibil la: <https://ziglang.org/>
- [35] „ZigEmbeddedGroup/microzig”. Zig Embedded Group, 3 martie 2024. Data accesării: 3 martie 2024. [Online]. Disponibil la: <https://github.com/ZigEmbeddedGroup/microzig>
- [36] J. Bertrand, „abstractthat/dactyl-manuform”. 2 martie 2024. Data accesării: 3 martie 2024. [Online]. Disponibil la: <https://github.com/abstractthat/dactyl-manuform>
- [37] „Logitech MX ERGO Advanced Wireless Trackball with Tilt Plate”. Data accesării: 3 martie 2024. [Online]. Disponibil la: <https://www.logitech.com/en-eu/products/mice/mx-ergo-wireless-trackball-mouse.html>

ANEXA A

Estimarea Costurilor

Tabelul A.1 – Estimarea costurilor

WBS	Nume	Start	Sfârșit	Durăță	Cost	Responsabil
1	Sistem de monitorizare a mediului	Sep 4	Apr 19	165d	91704	
1.1	Elaborarea sarcinii	Sep 4	Sep 29	20d	11200	IT Proiect Manager
1.2	Analiza domeniului	Oct 3	Oct 27	19d	11616	
1.2.1	Identificarea scopurilor si obiectivelor	Oct 3	Oct 4	2d	1,056	Analist Business IT
1.2.2	Cerintele sistemului	Oct 9	Oct 27	15d	10560	
1.2.2.1	Elicitarea cerintelor	Oct 9	Oct 16	6d	3,168	Analist Business IT
1.2.2.2	Analiza Cerintelor	Oct 16	Oct 23	6d	3,168	Analist Business IT
1.2.2.3	Validarea cerintelor	Oct 20	Oct 27	6d	3,168	Analist Business IT
1.2.2.4	Definirea specificatiilor tehnice	Oct 26	Oct 27	2d	1,056	Analist Business IT
1.3	Modelarea și proiectarea sistemului	Oct 30	Nov 22	18d	19320	
1.3.1	Proiectarea interfetei aplicatiei mobile	Oct 30	Nov 1	3d	2,760	Arhitect IT
1.3.2	Selectarea componentelor hardware	Nov 2	Nov 3	2d	1,840	Arhitect IT
1.3.3	Descrierea comportamentala a sistemului	Nov 3	Nov 15	9d	8,280	Arhitect IT
1.3.4	Descrierea structurii statice a sistemului	Nov 15	Nov 22	6d	5,520	Arhitect IT
1.3.5	Proiectarea protocolului de comunicare	Nov 22	Nov 22	1d	920	Arhitect IT
1.4	Realizarea sistemului	Jan 5	Apr 10	69d	46880	
1.4.1	Dezvoltarea aplicației mobile	Jan 5	Feb 23	36d	27360	
1.4.1.1	Instalarea mediului de dezvoltare	Jan 5	Jan 5	1d	608	Dezvoltator Android
1.4.1.2	Implementarea interfeței de utilizator	Jan 8	Feb 14	28d	17,024	Dezvoltator Android
1.4.1.3	Implementarea logicii aplicației	Feb 6	Feb 23	14d	8,512	Dezvoltator Android
1.4.1.4	Implementarea componentei de comunicare	Feb 15	Feb 16	2d	1,216	Dezvoltator Android
1.4.2	Dezvoltarea dispozitivului	Feb 19	Mar 13	18d	11520	
1.4.2.1	Asamblarea prototipului dispozitivului	Feb 19	Feb 19	1d	640	Programator C++
1.4.2.2	Instalarea mediului de dezvoltare	Feb 20	Feb 20	1d	640	Programator C++
1.4.2.3	Implementarea firmware pentru dispozitiv	Feb 19	Mar 7	14d	8,960	Programator C++
1.4.2.4	Implementarea componentei de comunicare	Mar 12	Mar 13	2d	1,280	Programator C++
1.4.3	Testarea sistemului	Mar 14	Apr 10	20d	8000	Tester IT
1.5	Documentarea sistemului	Apr 11	Apr 19	7d	2688	
1.5.1	Documentarea aplicatiei mobile	Apr 11	Apr 15	3d	1,008	Scriitor tehnic
1.5.2	Documentarea dispozitivului	Apr 16	Apr 18	3d	1,008	Scriitor tehnic
1.5.3	Realizarea raportului	Apr 18	Apr 19	2d	672	Scriitor tehnic