<Sistem automatizat pentru controlul unui vehicul destinat monitorizării parametrilor de mediu în regiuni greu accesibile>

# Documentul de proiectare

Cuprins

[1. Introducere 1](#_Toc160527836)

[1.1 Scopul documentului 1](#_Toc160527837)

[2. Prezentare generală și abordări de proiectare 2](#_Toc160527838)

[2.1 Prezentare generală 2](#_Toc160527839)

[2.2 Presupuneri/ Constrângeri/ Riscuri 2](#_Toc160527840)

[2.2.1 Presupuneri 2](#_Toc160527841)

[2.2.2 Constrângeri 2](#_Toc160527842)

[2.2.3 Riscuri 3](#_Toc160527843)

[3. Considerațiii de proiectare 4](#_Toc160527844)

[3.1 Obiective și linii directoare (ghiduri) 4](#_Toc160527845)

[3.2 Metode de dezvoltare 4](#_Toc160527846)

[3.3 Strategii de arhitectură 4](#_Toc160527847)

[4. Arhitectura Sistemului și Proiectarea Arhitecturii 6](#_Toc160527848)

[4.1 Vedere logică 6](#_Toc160527849)

[4.2 Arhitectură hardware 6](#_Toc160527850)

[4.3 Arhitectură software 6](#_Toc160527851)

[4.4 Arhitectura informațiilor 7](#_Toc160527852)

[4.5 Arhitectura de comunicații interne 7](#_Toc160527853)

[4.6 Diagrama de arhitectură a sistemului 8](#_Toc160527854)

[5. Proiectarea sistemului 9](#_Toc160527855)

[5.1 Proiectarea bazei de date 9](#_Toc160527856)

[5.1.1 Obiecte de date și structuri de date rezultante 9](#_Toc160527857)

[5.1.2 Fișiere și baze de date 9](#_Toc160527858)

[5.2 Conversii de date 9](#_Toc160527859)

[5.3 Interfețe utilizator 10](#_Toc160527860)

[5.3.1 Intrări 10](#_Toc160527861)

[5.3.2 Ieșiri 10](#_Toc160527862)

[5.4 Proiectarea interfețelor cu utilizatorul 10](#_Toc160527863)

[6. Scenarii de utilizare 11](#_Toc160527864)

[7. Proiectare de detaliu 12](#_Toc160527865)

[7.1 Proiectare hardware de detaliu 12](#_Toc160527866)

[7.2 Proiectare software de deatliu 12](#_Toc160527867)

[7.3 Proiectare detaliată de securitate 13](#_Toc160527868)

[7.4 Proiectare de detaliu pentru performanța sistemului 14](#_Toc160527869)

[7.5 Proiectare detaliată a comunicațiilor interne (între componente) 14](#_Toc160527870)

[8. Controale pentru verificarea integrității sistemului 15](#_Toc160527871)

[Anexa A: Gestiunea modificărilor documentului 16](#_Toc160527872)

[Anexa B: Acronime 17](#_Toc160527873)

[Anexa C Documente la care se face referire 18](#_Toc160527874)

## Introducere

Proiectul intitulat *Sistem automatizat pentru controlul unui vehicul destinat monitorizării parametrilor de mediu în regiuni greu accesibile* urmărește realizarea unui vehicul autonom controlat de o placă Arduino Mega, echipat cu senzori pentru monitorizarea mediului. Acest vehicul este capabil să colecteze date referitoare la temperatură, umiditate, calitatea aerului și poziția geografică, și să transmită aceste informații către un operator uman prin intermediul unei conexiuni Bluetooth. Documentul de față are rolul de a defini în mod detaliat arhitectura sistemului, atât la nivel hardware cât și software, oferind astfel un ghid complet pentru implementarea soluției.

Documentul se adresează echipei de dezvoltare și celor implicați în proiectarea și implementarea sistemului, incluzând managerul de proiect. De asemenea, anumite secțiuni ale documentului pot fi utile beneficiarului sistemului, în special cele legate de interfața de utilizare și scenariile de operare. Sistemul nu implică informații sensibile, dar accesul la funcționalitățile sale va fi restricționat utilizatorilor autorizați.

### Scopul documentului

Scopul prezentului document este de a descrie detaliat modul în care cerințele funcționale și non-funcționale ale sistemului sunt transpuse în specificații tehnice de proiectare, pentru a permite dezvoltarea coerentă și eficientă a vehiculului autonom destinat monitorizării mediului. Documentul acoperă atât arhitectura generală, cât și detaliile necesare implementării componentelor hardware și software, oferind o imagine completă asupra soluției propuse.

## Prezentare generală și abordări de proiectare

Această secțiune descrie principiile și strategiile care vor fi utilizate ca ghiduri în momentul proiectării și implementării sistemului.

### Prezentare generală

Proiectul presupune realizarea unui vehicul autonom care să funcționeze în condiții dificile de teren, colectând date de mediu cu ajutorul unor senzori specializați. Controlul vehiculului și comunicarea datelor se realizează prin intermediul unei conexiuni Bluetooth, facilitând interacțiunea în timp real cu operatorul. Senzorii montați pe vehicul vor transmite date către Arduino Mega, care va procesa informațiile și le va trimite mai departe către un dispozitiv mobil.

### Presupuneri/ Constrângeri/ Riscuri

#### Presupuneri

Se presupune că utilizatorul sistemului are cunoștințe minime în operarea interfeței și că vehiculul va funcționa în zone cu acces limitat, dar în care conexiunea Bluetooth poate fi stabilită în raza specifică de acțiune. Componentele hardware sunt disponibile pe piață, iar alimentarea se va face cu ajutorul unui acumulator care va fi optimizat pentru autonomie extinsă.

#### Constrângeri

Printre constrângerile sistemului se numără spațiul limitat pentru montarea componentelor pe vehicul, resursele limitate de procesare și memorie ale plăcii Arduino Mega, precum și necesitatea optimizării consumului energetic datorită alimentării pe baterie. De asemenea, viteza de deplasare a vehiculului va fi controlată pentru a asigura stabilitatea și siguranța echipamentelor.

#### Riscuri

Riscurile asociate sistemului includ pierderea conexiunii Bluetooth în cazul depășirii razei de acțiune, interferențele electromagnetice care pot afecta acuratețea senzorilor și eșecul alimentării în condiții extreme de teren

## Considerații de proiectare

Instrucțiuni: Descrieți problemele care trebuie abordate sau rezolvate înainte de a încerca să elaborați o soluție de design completă.

### Obiective și linii directoare (ghiduri)

Proiectarea sistemului urmărește atingerea unor obiective bine definite, care ghidează întregul proces de dezvoltare atât din punct de vedere hardware, cât și software. Unul dintre obiectivele fundamentale este obținerea unui sistem fiabil și eficient energetic, capabil să funcționeze autonom pe o perioadă cât mai lungă de timp, în condiții de teren dificile. În acest sens, toate deciziile de design se vor orienta spre optimizarea consumului de energie, inclusiv prin alegerea componentelor hardware cu un consum redus și prin implementarea unor rutine software care să permită utilizarea modurilor de economisire a energiei ale microcontrolerului.

Un alt obiectiv esențial este simplitatea și claritatea interacțiunii dintre utilizator și sistem. Interfața de control, bazată pe conexiune Bluetooth, trebuie să fie intuitivă și să permită operatorului să trimită comenzi clare, precum și să primească rapid datele colectate. În ceea ce privește codul software, o linie directoare importantă este scrierea unui cod modular, structurat, care să permită atât întreținerea ușoară, cât și extensibilitatea sistemului în viitor.

În ceea ce privește alegerea produselor și componentelor, se va opta pentru module hardware bine documentate și testate în proiecte similare, cum ar fi modulul Bluetooth HC-05, senzorul de temperatură și umiditate DHT22, senzorul de calitate a aerului MQ135 și GPS-ul Neo-6M. Aceste componente sunt alese nu doar pentru performanțele lor tehnice, ci și pentru disponibilitatea lor pe piață și comunitatea de suport care oferă biblioteci și exemple de utilizare.

### Metode de dezvoltare

Pentru dezvoltarea sistemului s-a utilizat o metodă bazată pe prototipare iterativă și proiectare structurată. Această abordare permite testarea rapidă a funcționalităților și ajustarea soluțiilor în funcție de rezultate. Codul software este modular, organizat în funcții clare, potrivit resurselor limitate ale plăcii Arduino Mega. S-au folosit diagrame UML simple pentru a vizualiza fluxurile și interacțiunile. Metode mai complexe, precum programarea orientată pe obiecte, au fost evitate din motive de eficiență. Posibile schimbări în design ar putea apărea din dificultăți tehnice legate de integrarea componentelor sau limitări de resurse, caz în care se vor adapta soluțiile prin simplificare sau înlocuirea modulelor problematice.

### Strategii de arhitectură

Arhitectura sistemului a fost proiectată având la bază strategii care să asigure simplitate, eficiență și fiabilitate. S-a optat pentru utilizarea limbajului C/C++ în mediul Arduino IDE, datorită compatibilității cu placa Arduino Mega și a suportului extins pentru modulele și senzorii folosiți. Bibliotecile standard, precum cele pentru senzorii DHT22, MQ135 și modulul Bluetooth HC-05, au fost preferate pentru a reduce timpul de dezvoltare, oferind funcționalități testate și stabile.

Interfața cu utilizatorul se bazează pe schimbul de date text prin Bluetooth, cu comenzi simple și răspunsuri rapide, evitând interfețele grafice complexe care ar necesita resurse suplimentare. La nivel hardware, toate componentele sunt conectate direct la Arduino Mega, folosind protocoale precum I2C și Serial, asigurând o arhitectură centralizată, ușor de controlat și monitorizat.

Pentru gestionarea memoriei, s-au utilizat variabile și structuri de date simple, având în vedere limitările de RAM ale plăcii. Nu s-au implementat baze de date, datele fiind transmise direct către utilizator, iar persistența acestora se realizează extern, pe dispozitivul de control.

Mecanismul de comunicare este asigurat prin Bluetooth, preferat în locul Wi-Fi sau GSM pentru simplitatea integrării și consumul redus de energie. S-a considerat utilizarea unei conexiuni Wi-Fi, dar aceasta a fost respinsă din cauza complexității și a consumului mai ridicat, iar GSM a fost exclus pentru a evita costurile suplimentare.

Detectarea erorilor se face prin validarea datelor primite de la senzori și prin verificarea conexiunii Bluetooth, iar în caz de eroare, sistemul transmite mesaje de alertă către operator. Controlul vehiculului este secvențial, fără concurență, toate acțiunile fiind procesate în ordine, ceea ce simplifică implementarea și evită problemele de sincronizare.

Strategia generală urmărește un echilibru între funcționalitate și simplitate, asigurând o soluție robustă, dar flexibilă pentru eventuale îmbunătățiri viitoare, cum ar fi adăugarea de noi senzori sau integrarea unei memorii externe pentru stocarea locală a datelor.

## Arhitectura Sistemului și Proiectarea Arhitecturii

Arhitectura sistemului este organizată în jurul plăcii Arduino Mega, care gestionează toate componentele esențiale ale vehiculului. Sistemul este împărțit în patru componente majore: modulul de colectare a datelor de mediu, modulul de comunicare Bluetooth, modulul de control al deplasării vehiculului și sursa de alimentare. Fiecare dintre aceste module are responsabilități bine definite, iar datele sunt transmise între ele sub coordonarea directă a plăcii Arduino.

Modulul de colectare a datelor include senzorul DHT22, care măsoară temperatura și umiditatea ambientală, și senzorul MQ135, utilizat pentru determinarea calității aerului. Aceste date sunt preluate de Arduino într-un ciclu continuu și pregătite pentru a fi transmise către utilizator.

Modulul de comunicare este realizat prin intermediul unui modul Bluetooth HC-05, care asigură transmiterea datelor către un dispozitiv de control, cum ar fi un telefon mobil sau un laptop, și primește comenzi de la utilizator pentru controlul vehiculului.

Modulul de control al deplasării vehiculului include driverul L298N, care acționează motoarele DC în funcție de comenzile primite. Arduino interpretează aceste comenzi și reglează viteza și direcția de deplasare. În acest fel, vehiculul se poate mișca înainte, înapoi sau se poate opri, în funcție de instrucțiunile primite prin Bluetooth.

Sursa de alimentare, bazată pe o baterie Li-Ion, furnizează energia necesară întregului sistem, iar Arduino monitorizează starea componentelor pentru a preveni funcționarea defectuoasă în cazul scăderii tensiunii.

Această structură modulară a fost aleasă pentru a permite o dezvoltare clară și controlată a fiecărei funcționalități, asigurând totodată ușurința întreținerii și posibile extinderi ulterioare. Alternative mai complexe, precum integrarea unor sisteme de poziționare sau control distribuit, au fost respinse în favoarea unei soluții simple, fiabile și eficiente energetic.

Fluxul de date între componente este controlat secvențial, utilizând tiparul de proiectare „polling loop”, în care Arduino interoghează periodic senzorii, verifică existența comenzilor noi și acționează asupra motoarelor. Această abordare asigură o sincronizare clară între funcționalități, fiind ideală pentru sisteme embedded cu resurse limitate.

Prin urmare, arhitectura sistemului permite o interacțiune eficientă între componentele hardware și software, oferind un control fiabil al vehiculului și o colectare precisă a parametrilor de mediu, fără a fi necesară integrarea unor module suplimentare, precum GPS.

### Vedere logică

Vederea logică include și aplicația mobilă, care primește datele trimise prin Bluetooth de la Arduino și trimite înapoi comenzi pentru controlul vehiculului. Fluxul complet este: **Senzori → Arduino → Bluetooth → Aplicație mobilă ↔ Comenzi ↔ Arduino → Motoare**. Aplicația afișează datele în timp real și permite utilizatorului să controleze direcția și funcționarea vehiculului.

### Arhitectură hardware

Arhitectura hardware a sistemului este centralizată, având ca element principal placa Arduino Mega 2560, care coordonează toate procesele de colectare a datelor, comunicare și control al deplasării vehiculului. Arduino Mega funcționează la o frecvență de 16 MHz, având 256 KB memorie flash și 8 KB SRAM, suficiente pentru a gestiona toate componentele sistemului. Senzorul DHT22, responsabil pentru măsurarea temperaturii și umidității, și senzorul MQ135, utilizat pentru monitorizarea calității aerului, sunt conectați direct la pinii digitali și analogici ai plăcii. Comunicarea cu aplicația mobilă se face prin modulul Bluetooth HC-05, care este legat la portul serial al Arduino.

Pentru mișcarea vehiculului, semnalele de control sunt transmise de Arduino către driverul L298N, care acționează două motoare DC, asigurând deplasarea. Alimentarea întregului sistem este realizată prin două acumulatori Li-Ion de 3.7V conectați în serie, oferind astfel o tensiune totală de 7.4V, suficientă pentru funcționarea stabilă a plăcii Arduino, a senzorilor și a motoarelor. Componentele sunt montate pe șasiul vehiculului într-o configurație compactă, optimizată pentru stabilitate și accesibilitate.

Sistemul nu depinde de servere externe sau rețele, fiind complet autonom, cu procesare locală și transmiterea directă a datelor prin Bluetooth către aplicația mobilă. Datele nu sunt stocate local, ci transmise în timp real, iar memoria internă a plăcii este utilizată doar temporar pentru procesarea acestora.

### Arhitectură software

Arhitectura software a sistemului se bazează pe interacțiunea dintre codul rulat pe placa Arduino Mega și aplicația mobilă realizată în MIT App Inventor. Codul pentru Arduino este scris în limbajul C/C++ folosind mediul de dezvoltare Arduino IDE, versiunea 1.8.19. Acesta gestionează citirea senzorilor de mediu, comunicarea Bluetooth și controlul motoarelor. Bibliotecile principale utilizate sunt "DHT.h" pentru citirea temperaturii și umidității de la senzorul DHT22, "MQ135.h" pentru monitorizarea calității aerului și "SoftwareSerial.h" pentru comunicarea cu modulul Bluetooth HC-05.

Codul este structurat modular, cu funcții dedicate pentru citirea fiecărui senzor, procesarea datelor, trimiterea acestora prin Bluetooth și interpretarea comenzilor primite de la aplicația mobilă. Toate funcțiile sunt gestionate printr-un ciclu continuu în bucla principală, asigurând colectarea și actualizarea constantă a datelor.

Aplicația mobilă a fost creată în MIT App Inventor, o platformă vizuală de dezvoltare, fără a necesita cunoștințe avansate de programare. Aceasta are rolul de a primi datele transmise prin Bluetooth de la Arduino și de a le afișa într-o interfață grafică simplă, sub formă de text sau valori numerice. Utilizatorul poate trimite comenzi către vehicul prin butoane dedicate din aplicație, cum ar fi înainte, înapoi, stânga, dreapta sau stop.

Comunicarea între Arduino și aplicație se face prin mesaje text transmise prin Bluetooth, folosind un protocol simplu, în care fiecare mesaj este delimitat și identificabil, pentru a facilita interpretarea corectă. Aplicația utilizează componentele BluetoothClient din MIT App Inventor pentru stabilirea conexiunii și trimiterea/primirea mesajelor.

Această soluție software a fost aleasă pentru simplitatea integrării, eficiența în utilizarea resurselor limitate ale plăcii Arduino și ușurința dezvoltării aplicației mobile, fără a necesita utilizarea unor framework-uri complexe sau baze de date. Codul Arduino și blocurile de programare vizuală din MIT App Inventor lucrează împreună pentru a oferi un sistem funcțional, ușor de utilizat și extins, dacă este necesar.

### Arhitectura informațiilor

În cadrul sistemului, informațiile gestionate sunt exclusiv date de mediu, colectate în timp real de la senzorii montați pe vehicul. Acestea includ temperatura, umiditatea și calitatea aerului, exprimate în valori numerice. Datele nu sunt stocate local pe vehicul, ci sunt transmise imediat prin Bluetooth către aplicația mobilă, unde sunt afișate utilizatorului.

Nu sunt gestionate informații cu caracter personal sau sensibil. Toate datele procesate sunt de tip electronic, furnizate automat de către senzorii DHT22 și MQ135. Utilizatorul sistemului nu introduce date manual, ci doar primește informațiile transmise de sistem și poate trimite comenzi de control.

Astfel, sistemul funcționează ca un flux continuu de colectare și transmitere de date numerice, fără stocare permanentă, iar toate informațiile sunt utilizate exclusiv pentru monitorizarea parametrilor de mediu în timp real.

### Arhitectura de comunicații interne

Arhitectura de comunicații interne este locală și simplă, toate componentele fiind conectate direct la Arduino Mega. Senzorii transmit date către Arduino prin conexiuni digitale și analogice, iar modulul Bluetooth HC-05 comunică cu aplicația mobilă prin protocol serial UART. Arduino controlează motoarele prin driverul L298N, folosind semnale digitale. Nu sunt folosite rețele externe, iar volumul de date este mic, suficient pentru comunicația prin Bluetooth.

### Diagrama de arhitectură a sistemului

Senzorii DHT22 și MQ135 sunt conectați direct la placa Arduino Mega, unde datele sunt colectate și procesate. Arduino transmite aceste date prin modulul Bluetooth HC-05 către aplicația mobilă, care le afișează utilizatorului. Tot prin Bluetooth, aplicația trimite comenzi către Arduino, care acționează asupra driverului L298N pentru a controla motoarele vehiculului.

Fluxul sistemului este:  
**Senzori → Arduino Mega → Bluetooth HC-05 → Aplicație mobilă ↔ Comenzi ↔ Arduino Mega → Driver L298N → Motoare DC**

Această diagramă logică arată cum toate componentele colaborează, formând un sistem compact și eficient, fără a depinde de alte rețele sau infrastructuri externe.

.

## Proiectarea sistemului

### Proiectarea bazei de date

Sistemul nu utilizează o bază de date internă, întrucât datele colectate sunt procesate și transmise în timp real, fără a fi stocate local pe vehicul. Nu există un sistem DBMS sau fișiere de stocare non-DBMS asociate în cadrul acestui proiect.

Toate datele – temperatura, umiditatea și calitatea aerului – sunt colectate de la senzori, procesate de Arduino și imediat trimise prin Bluetooth către aplicația mobilă, unde pot fi vizualizate. Aplicația, la rândul ei, nu stochează aceste date, ci le afișează în timp real utilizatorului.

În cazul în care, ulterior, se dorește adăugarea unei funcționalități de salvare a datelor, acest lucru se poate realiza la nivelul aplicației mobile, unde datele pot fi stocate local în fișiere text sau baze de date simple, cum ar fi TinyDB, disponibilă în MIT App Inventor.

Pentru moment, sistemul funcționează fără componente de tip bază de date, toate informațiile fiind volatile și utilizate pe loc.

.

#### Obiecte de date și structuri de date rezultante

Datele utilizate în sistem sunt gestionate prin variabile simple, fără structuri complexe. Temperatura și umiditatea sunt stocate temporar ca valori numerice de tip float, iar calitatea aerului ca valoare de tip int. Aceste valori sunt procesate de Arduino și transformate într-un șir de caractere, care conține toate datele într-un format clar, de exemplu „T:23.5 H:45.0 AQ:150”. Acest șir este transmis prin Bluetooth către aplicația mobilă, fiind principala structură de date utilizată în comunicarea între componente.

#### Fișiere și baze de date

Sistemul nu utilizează fișiere sau baze de date pentru stocarea locală a datelor, deoarece toate informațiile sunt procesate și transmise în timp real. Datele colectate de la senzori sunt stocate temporar în memoria plăcii Arduino, sub formă de variabile, și sunt imediat trimise prin Bluetooth către aplicația mobilă.

##### Baze de date

Sistemul nu utilizează baze de date și nu există fișiere DBMS, deoarece toate datele sunt prelucrate și transmise în timp real, fără a fi stocate. Nu este necesară proiectarea unui sistem de baze de date, întrucât arhitectura propusă nu prevede salvarea informațiilor colectate, ci doar afișarea lor temporară în aplicația mobilă. Toate datele sunt volatile și nu implică gestionarea printr-un sistem DBMS.

##### Fișiere non-DBMS

În cadrul acestui sistem nu sunt utilizate fișiere non-DBMS. Datele colectate sunt procesate direct în memoria Arduino și transmise imediat către aplicația mobilă, fără a fi scrise sau citite din fișiere locale. Nu există fișiere de intrare, ieșire sau temporare implicate în funcționarea sistemului.

Deoarece toate datele sunt gestionate în timp real și nu există necesitatea stocării lor permanente sau temporare pe dispozitiv, nu se definesc structuri de înregistrare, lungimi de fișiere, chei, indici sau politici de backup și recuperare. Sistemul este conceput pentru funcționare continuă, fără manipularea fișierelor.

### Conversii de date

În cadrul sistemului nu sunt necesare conversii complexe de date. Conversia se face doar la nivel de format, de la valori numerice (float pentru temperatură și umiditate, int pentru calitatea aerului) în șiruri de caractere, pentru a fi transmise prin Bluetooth. Aceste conversii sunt realizate direct în codul Arduino, folosind funcții simple, precum String() pentru transformarea valorilor numerice în text.

Nu există documente separate pentru conversii de date, deoarece totul este implementat direct în codul sursă, iar datele nu suferă alte transformări sau conversii suplimentare.

### Interfețe utilizator

Sistemul are o singură clasă de utilizatori, denumită generic operator. Operatorul este persoana care controlează vehiculul și monitorizează datele colectate prin intermediul aplicației mobile. Această clasă de utilizatori are responsabilitatea de a porni vehiculul, de a trimite comenzi de deplasare și de a interpreta valorile transmise de sistem referitoare la temperatură, umiditate și calitatea aerului.

Nivelul de competență necesar pentru utilizatori este minim, fiind necesare doar cunoștințe de bază în utilizarea unei aplicații mobile și în interpretarea valorilor afișate. Interacțiunea cu sistemul se face exclusiv prin aplicația mobilă, utilizând conexiunea Bluetooth.

Numărul total de utilizatori anticipați este de unu, vehiculul fiind controlat de un singur operator la un moment dat. Nu sunt prevăzuți utilizatori simultani și nici utilizatori externi, accesul fiind restricționat celui care operează direct sistemul în teren.

#### Intrări

Intrările în sistem sunt realizate de către utilizator prin aplicația mobilă, folosind interfața grafică simplă dezvoltată în MIT App Inventor. Utilizatorul trimite comenzi către vehicul prin apăsarea unor butoane dedicate pe ecranul aplicației, fiecare buton reprezentând o acțiune specifică: înainte, înapoi, stânga, dreapta și stop.

Fiecare apăsare a unui buton trimite prin Bluetooth un mesaj text simplu, cum ar fi „FWD” pentru deplasare înainte, „BWD” pentru înapoi, „LEFT”, „RIGHT” sau „STOP”. Aceste comenzi sunt recepționate de Arduino, care le procesează și acționează corespunzător asupra motoarelor vehiculului.

Interfața de intrare constă într-un ecran cu butoane, fără introducere manuală de text. Nu există criterii de editare avansate, deoarece toate valorile sunt predefinite. Nu sunt posibile introduceri greșite, iar sistemul este protejat de comenzi nevalide, Arduino ignorând orice mesaj care nu corespunde comenzilor așteptate.

Mesajele trimise sunt de tip alfanumeric scurt, având lungimi între 3 și 5 caractere. Nu sunt utilizate fișiere sau baze de date pentru intrări, iar toate datele sunt transmise direct prin Bluetooth, în timp real.

#### Ieșiri

Ieșirile sistemului sunt afișate în aplicația mobilă și includ valorile curente ale temperaturii, umidității și calității aerului, transmise de Arduino prin Bluetooth. Aceste date apar pe ecran sub formă de text: temperatura în grade Celsius, umiditatea în procente și calitatea aerului ca valoare numerică. Scopul acestor ieșiri este informarea în timp real a operatorului, fără generarea de fișiere sau rapoarte. Datele nu sunt stocate, ci doar vizualizate pe durata funcționării sistemului.

### Proiectarea interfețelor cu utilizatorul

Interfața aplicației mobile are trei zone: afișarea datelor, controlul vehiculului și conexiunea Bluetooth. Datele de temperatură, umiditate și calitatea aerului sunt afișate în timp real. Controlul se face prin butoane simple: Înainte, Înapoi, Stânga, Dreapta și Stop. Conexiunea Bluetooth se realizează printr-un buton dedicat, cu afișarea stării „Conectat” sau „Neconectat”. Interfața este clară și ușor de folosit.

## Scenarii de utilizare

Un scenariu tipic de utilizare începe cu pornirea sistemului de către operator. După alimentarea vehiculului, operatorul deschide aplicația mobilă și apasă butonul de conectare Bluetooth, selectând modulul HC-05. Odată stabilită conexiunea, aplicația afișează datele de mediu în timp real: temperatura, umiditatea și calitatea aerului.

Operatorul observă valorile afișate și, dacă dorește să deplaseze vehiculul, apasă butoanele de control din aplicație, cum ar fi Înainte sau Stânga. Comanda este trimisă către Arduino, care controlează motoarele pentru a executa mișcarea dorită. Pe măsură ce vehiculul se deplasează, senzorii continuă să transmită date actualizate.

Dacă conexiunea Bluetooth se pierde, aplicația notifică utilizatorul, iar acesta trebuie să reconecteze dispozitivul. Sistemul nu reacționează la comenzi invalide și nu reține datele colectate după oprirea alimentării.

Acest flux simplu asigură interacțiunea continuă dintre utilizator și sistem:

alimentare → conectare → afișare date → control vehicul → monitorizare continuă.

## Proiectare de detaliu

Pentru realizarea sistemului, echipa de dezvoltare va integra componentele hardware conform următoarei structuri: senzorul DHT22 va fi conectat la un pin digital al plăcii Arduino Mega, iar senzorul MQ135 la un pin analogic. Modulul Bluetooth HC-05 va fi conectat la pinii TX și RX ai Arduino, utilizând alimentare de 5V. Driverul L298N va fi conectat la pinii digitali pentru controlul direcției motoarelor, alimentat separat de la bateriile de 7.4V (două acumulatori de 3.7V în serie), iar motoarele DC vor fi conectate la ieșirile driverului.

Software-ul va fi scris în Arduino IDE, folosind bibliotecile DHT, MQ135 și SoftwareSerial. Codul va fi organizat în funcții clare pentru citirea senzorilor, transmiterea datelor prin Bluetooth și interpretarea comenzilor de control. Aplicația mobilă va fi creată în MIT App Inventor, cu elemente pentru afișarea datelor, butoane pentru control și conexiune Bluetooth.

Integrarea se realizează prin stabilirea unei conexiuni stabile între componente, testarea comunicației Bluetooth și verificarea răspunsului motoarelor la comenzi. Sistemul trebuie alimentat corect, verificând stabilitatea tensiunilor, iar toate componentele vor fi fixate solid pe șasiul vehiculului.

### Proiectare hardware de detaliu

Sistemul include următoarele componente hardware integrate pe vehicul, toate controlate centralizat de placa Arduino Mega 2560:

1. **Arduino Mega 2560** – microcontroler ATmega2560, frecvență 16 MHz, memorie flash 256 KB, SRAM 8 KB. Are 54 pini digitali și 16 pini analogici. Este unitatea principală care procesează datele și coordonează toate componentele.
2. **Senzor DHT22** – măsoară temperatura și umiditatea. Domeniu: -40°C la +80°C și 0-100% umiditate. Precizie: ±0.5°C / ±2-5%. Alimentare: 3.3V-6V. Conectare la pin digital.
3. **Senzor MQ135** – detectează calitatea aerului (NH3, NOx, alcool, benzen, fum). Alimentare: 5V. Semnal analogic, conectat la pin analogic.
4. **Senzor BMP180** – măsoară presiunea atmosferică și altitudinea. Precizie: ±1 hPa. Comunicare I2C, alimentare: 3.3V-5V.
5. **Senzor ultrasonic HC-SR04** – măsoară distanța față de obstacole. Domeniu: 2-400 cm, precizie ±3 mm. Alimentare: 5V. Conectare: pini digitali (Trigger și Echo).
6. **Senzor de sunet** – detectează intensitatea zgomotului ambiental. Alimentare: 5V. Semnal analogic sau digital.
7. **Senzor de intensitate luminoasă (LDR)** – detectează nivelul de lumină. Alimentare: 5V. Semnal analogic.
8. **Modul Bluetooth HC-05** – comunicare serială UART cu aplicația mobilă. Alimentare: 3.6V-6V. Conectare: TX/RX.
9. **Driver motoare L298N** – controlează două motoare DC. Alimentare: 5V-12V. Semnale de control de la Arduino prin pini digitali.
10. **Motoare DC (2 bucăți)** – alimentare 6-12V, controlate de L298N pentru mișcarea vehiculului.
11. **Buzzer** – element sonor pentru alerte. Alimentare: 5V. Controlat digital de Arduino.
12. **Acumulatori Li-Ion 3.7V (2 bucăți în serie)** – oferă 7.4V pentru alimentarea sistemului. Capacitate: 2000-3000 mAh fiecare.

Arduino colectează date de la toți senzorii, procesează aceste informații și le transmite prin Bluetooth către aplicație. Pe baza comenzilor primite, Arduino controlează motoarele pentru deplasare și buzzer-ul pentru avertizări sonore. Senzorul ultrasonic ajută la evitarea obstacolelor, iar ceilalți senzori contribuie la monitorizarea mediului.

### Proiectare software de detaliu

Sistemul software este alcătuit din patru servicii principale integrate în codul Arduino, fiecare având un rol esențial în funcționarea corectă a vehiculului. Primul serviciu este cel de citire a senzorilor, care colectează în mod continuu date privind temperatura, umiditatea, calitatea aerului, presiunea atmosferică, nivelul de sunet și intensitatea luminoasă. Aceste date sunt stocate temporar în variabile interne și actualizate constant. Al doilea serviciu este responsabil de comunicarea Bluetooth, unde datele colectate sunt formate într-un mesaj text și trimise către aplicația mobilă, iar comenzile primite sunt interpretate pentru a fi executate.

Serviciul de control al motoarelor preia comenzile primite prin Bluetooth și acționează asupra driverului L298N pentru a dirija vehiculul în direcția dorită de utilizator, fie înainte, înapoi sau oprire. Ultimul serviciu, cel de avertizare acustică, monitorizează valorile senzorilor și, în cazul în care anumite praguri sunt depășite, activează buzzer-ul pentru a alerta utilizatorul.

Toate aceste componente software funcționează în strânsă legătură, asigurând o interacțiune continuă și fluidă între hardware și aplicația mobilă. Codul este organizat în funcții separate, clar comentate, pentru ușurința întreținerii și extinderii viitoare. Procesarea datelor este rapidă și optimizată pentru resursele limitate ale plăcii Arduino Mega, iar întregul sistem este conceput să reacționeze în timp real la schimbările de mediu și la comenzile utilizatorului.

### Proiectare detaliată de securitate

Sistemul nu dispune de componente complexe de securitate, având în vedere natura sa locală și limitată. Conexiunea între vehicul și aplicația mobilă se realizează prin Bluetooth, fără acces la rețele externe. Cu toate acestea, se aplică următoarele măsuri de bază:

**Autentificare:** Modulul Bluetooth HC-05 necesită o parolare la prima conectare (de obicei 1234 sau 0000). Aceasta asigură că doar dispozitivele autorizate pot stabili conexiunea.

**Autorizare:** După conectare, doar utilizatorul cu aplicația mobilă poate trimite comenzi către vehicul. Nu sunt acceptate comenzi din alte surse.

**Jurnalizare și auditare:** Sistemul nu păstrează log-uri sau arhive ale datelor sau comenzilor, funcționând în timp real, fără stocare permanentă.

**Criptare:** Datele transmise prin Bluetooth nu sunt criptate, fiind transmise ca text simplu, întrucât riscurile sunt minime datorită razei limitate de acțiune.

**Porturi de rețea:** Sistemul folosește doar conexiunea serială Bluetooth, fără implicarea altor porturi sau rețele IP.

**Detectare și prevenire a intruziunilor:** Nu există mecanisme avansate de detecție, însă orice comandă nevalidă sau neautorizată este ignorată de sistem, prevenind comportamente neprevăzute. Conexiunea Bluetooth poate fi întreruptă manual pentru a opri accesul.

Aceste măsuri sunt suficiente pentru un sistem local, controlat direct și cu un risc scăzut de acces neautorizat.

### Proiectare de detaliu pentru performanța sistemului

Sistemul este proiectat să funcționeze eficient în timp real, având cerințe reduse de capacitate și volum de date. Datele colectate sunt simple și constau în valori numerice de la senzori, transmise la intervale regulate.

**Cerințe și estimări de capacitate:** Sistemul gestionează un volum redus de date, aproximativ 30-50 de caractere per ciclu de transmisie. Transmisiile au loc la fiecare 1-2 secunde, rezultând un trafic de aproximativ 1 KB pe minut. Arduino utilizează sub 50% din memoria disponibilă (256 KB flash, 8 KB SRAM).

**Așteptări de performanță:** Sistemul trebuie să proceseze și să transmită date în timp real, cu o latență sub 1 secundă. Răspunsul la comenzile utilizatorului trebuie să fie instantaneu (sub 500 ms).

**Cerințe de disponibilitate:** Sistemul trebuie să funcționeze continuu pe durata unei sesiuni de operare, de aproximativ 1-2 ore, în funcție de capacitatea bateriilor.

**Proiectare de performanță:** Codul este optimizat pentru procesare rapidă, folosind structuri de date simple și bucle eficiente. Comunicarea Bluetooth este stabilă, la o rată de 9600 baud, suficientă pentru volumul de date.

**Proiectare de fiabilitate:** Componentele hardware sunt alese pentru stabilitate și consum redus. Senzorii și conexiunile sunt testate pentru funcționare constantă, iar sistemul este protejat împotriva valorilor de intrare incorecte.

**Backup, recuperare și arhivare:** Nu există mecanisme de backup sau arhivare, deoarece sistemul nu stochează date. În caz de oprire, la repornire sistemul funcționează normal, fără necesitatea recuperării informațiilor.

**Puncte unice de eșec:** Arduino Mega reprezintă punctul unic de eșec, deoarece toate componentele depind de acesta. În cazul unei defecțiuni, întregul sistem se oprește. Nu este necesară disponibilitate ridicată (clustering), deoarece este un sistem local, de mică anvergură.

Această proiectare asigură performanța necesară pentru scopul sistemului, într-un mod eficient și stabil.

### Proiectare detaliată a comunicațiilor interne (între componente)

Sistemul funcționează cu o arhitectură centralizată, unde toate comunicațiile interne sunt gestionate de Arduino Mega. Nu există servere sau rețele LAN/WAN, toate componentele fiind conectate fizic direct la placă.

**Componente și comunicații interne:**

* **Senzorii DHT22, MQ135, LDR, senzor sunet** transmit date direct către Arduino prin pini digitali sau analogici.
* **BMP180** comunică cu Arduino prin protocol I2C.
* **Senzorul ultrasonic HC-SR04** folosește doi pini digitali (Trigger și Echo).
* **Modulul Bluetooth HC-05** este conectat la portul serial al Arduino prin TX/RX.
* **Driverul L298N** primește semnale PWM și digitale de la Arduino pentru controlul motoarelor.
* **Buzzer-ul** este controlat printr-un pin digital.

**Formatul datelor:**  
Datele transmise între componente sunt de tip numeric (float sau int), iar între Arduino și Bluetooth sunt convertite în șiruri de caractere text (ex: "T:23.5 H:45 AQ:150").

**Specificații de sincronizare:**  
Arduino funcționează pe un ciclu continuu de citire – procesare – transmitere, fără cerințe avansate de sincronizare. Comunicarea pe bus-ul I2C este gestionată de Arduino, fără conflicte.

**Conectivitate și distanțe:**  
Toate componentele sunt montate pe același șasiu, la distanțe de câțiva centimetri, cu fire directe. Nu există variații semnificative de semnal sau pierderi pe distanță.

**Topologie:**  
Topologia este punct-la-punct, fiecare componentă fiind conectată direct la placa Arduino, fără intermediere.

Această configurație simplificată asigură comunicarea rapidă și fiabilă între toate elementele sistemului.

## Controale pentru verificarea integrității sistemului

Sistemul este unul local, simplu, fără baze de date sau stocare de informații critice, astfel controalele pentru verificarea integrității sunt aplicate la nivel operațional, adaptate la specificul său.

**Securitate internă:** Accesul la sistem este restricționat prin conexiunea Bluetooth, care necesită împerechere și parolă (ex: 1234). Doar utilizatorul autorizat, cu aplicația mobilă conectată, poate trimite comenzi și vizualiza datele.

**Proceduri de audit:** Nu există stocare de date și nici generare de rapoarte, deci nu se aplică proceduri de audit sau perioade de reținere. Sistemul funcționează exclusiv în timp real.

**Piste de auditare:** Nu sunt implementate mecanisme de auditare a accesului sau recuperării datelor, întrucât datele nu sunt salvate, ci doar afișate temporar.

**Validarea câmpurilor de date:** Datele primite de la senzori sunt validate în programul Arduino prin verificarea dacă sunt în intervale normale. De exemplu, temperatura trebuie să fie între -40°C și +80°C, umiditatea între 0-100%.

**Verificare date:** Nu există procese de adăugare, ștergere sau actualizare a datelor, sistemul doar le colectează și transmite. Integritatea este asigurată prin procesare directă și validare în timp real.

**Identificare audit:** Sistemul nu colectează informații despre utilizator sau despre terminalul de control. Nu sunt urmărite date precum ora sau identitatea utilizatorului, toate operațiunile fiind locale și directe.

Astfel, controlul asupra integrității sistemului este asigurat prin simplitatea arhitecturii și prin validarea continuă a datelor și comenzilor, fără stocare sau manipulare avansată a acestora.

Anexa A: Gestiunea modificărilor documentului

Instrucțiuni: Furnizați informații despre modul în care dezvoltarea și distribuția documentului va fi controlată și urmărită. Utilizați tabelul de mai jos pentru a furniza numărul de versiune, data versiunii, autorul/deținătorul versiunii și o scurtă descriere a motivului pentru crearea versiunii revizuite.

Tabel 1 – Înregistrarea modificărilor asupreaa documentului curent

| versiune | Data | Autorul/Deținătorul | Descriere |
| --- | --- | --- | --- |
| <X.X> | <ZZ/LL/AAAA> | <nume autor> | <Descrierea modificării> |
| <X.X> | <ZZ/LL/AAAA> | <nume autor> | <Descrierea modificării> |
| <X.X> | <ZZ/LL/AAAA> | <nume autor> | <Descrierea modificării> |

Anexa B: Acronime

*Instrucțiuni: Furnizați o listă de acronime și traduceri literale asociate utilizate în cadrul documentului. Enumerați acronimele în ordine alfabetică folosind un format tabular, așa cum este ilustrat mai jos.*

Tabel 2 - Acronime

| Acronim | Forma completă |
| --- | --- |
| <Acronim> | <Forma completă> |
| <Acronim> | <Forma completă> |
| <Acronim> | <Forma completă> |

Anexa C Documente la care se face referire

*Instrucțiuni: Sintetizați relația acestui document cu alte documente relevante. Furnizați informații de identificare pentru toate documentele folosite pentru a ajunge la și/sau referite în acest document (de exemplu, documente conexe și/sau asociate, documente prealabile, documentație tehnică relevantă, etc.).*

Tabel 3 – Documente la care se facce referire

| Nume document | Locație sau URL | Dată emitere document |
| --- | --- | --- |
| < Nume document > | <Locație sau URL> | <ZZ/LL/AAAA> |
| < Nume document > | <Locație sau URL> | <ZZ/LL/AAAA> |
| < Nume document > | <Locație sau URL> | <ZZ/LL/AAAA> |